University of Zurich[UZH]

Burkhard Stiller, Chao Feng, Daria Schumm, Jan von der Assen, Katharina O. E. Müller, Nazim Nezhadsistani, Thomas Grübl, Weijie Niu (Edts.)

Communication Systems XVIII

TECHNICAL REPORT – No. IFI-2025.02

June 2025

University of Zurich
Department of Informatics (IFI)
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland

ifi

Introduction

The Department of Informatics (IFI) of the University of Zurich, Switzerland works on research and teaching in the area of computer networks and communication systems. Communication systems include a wide range of topics and drive many research and development activities. Therefore, during the spring term FS 2024 a new instance of the Communication Systems seminar has been prepared and students as well as supervisors worked on this topic.

The areas of communication systems include, among others, wired and wireless network technologies, various network protocols, network management, Quality-of-Service (QoS) provisioning, mobility, security aspects, peer-to-peer systems, multimedia communication, and manifold applications, determining important parts of future networks. Therefore, this year's seminar addressed such areas in more depth. The understanding and clear identification of problems in technical and organizational terms have been prepared, and challenges as well as weaknesses of existing approaches have been addressed. All talks in this seminar provide a systematic approach to judge dedicated pieces of systems or proposals and their suitability.

Content

This new edition of the seminar entitled "Communication Systems XVIII" discusses a number of selected topics in the area of computer networks and communication systems. The seminar begins with Talk 1, which explores the evolution toward passwordless authentication, providing an overview of standards such as UAF and FIDO2 that enhance both security and usability by eliminating reliance on traditional passwords. Talk 2 shifts focus to transport protocols for the modern web, analyzing how technologies like QUIC and MASQUE improve latency, multiplexing, and performance tradeoffs. Talk 3 transitions to blockchain applications beyond cryptocurrency, discussing real-world use cases including digital identity, supply chain verification, and electronic voting, and emphasizing the benefits of decentralization and immutability. In Talk 4, the focus turns to machine unlearning in large language models (LLMs), examining emerging techniques for removing specific training data to meet privacy requirements while maintaining overall model performance. Talk 5 delves into the cryptographic underpinnings of blockchain systems, covering essential methods such as hashing, digital signatures, and zero-knowledge proofs that ensure security and data integrity. Talk 6 addresses the

pressing need for post-quantum cryptography in secure messaging, introducing quantum-resistant algorithms designed to withstand future adversaries equipped with quantum computing power. In Talk 7, the discussion broadens to AI agent systems, highlighting their communication protocols, coordination strategies, and applications in automation and digital ecosystems. Talk 8 then examines the role of large language models in fact-checking, presenting strategies for assessing factual reliability, including external evidence grounding and explainability techniques. Finally, Talk 9 investigates the integration of machine learning in securing 5G networks, showcasing techniques like federated and reinforcement learning to detect threats, with a case study focused on DDoS mitigation using a decentralized learning architecture.

Seminar Operation

Based on well-developed experiences of former seminars, held in different academic environments, all interested students worked on an initially offered set of papers and book chapters. Those relate to the topic titles as presented in the Table of Contents below. They prepared a written essay as a clearly focused presentation, an evaluation, and a summary of those topics. Each of these essays is included in this technical report as a separate section and allows for an overview of important areas of concern, technology architectures and functionality, sometimes business models in operation, and problems encountered.

In addition, every group of students prepared a slide presentation of approximately 45 minutes to present its findings and summaries to the audience of students attending the seminar and other interested students, research assistants, and professors. Following a general question and answer phase, a student-led discussion debated open issues and critical statements with the audience.

Local IFI support for preparing talks, reports, and their preparation by students had been granted by Chao Feng, Daria Schumm, Jan von der Assen, Katharina O. E. Müller, Nazim Nezhadsistani, Thomas Grübl, Weijie Niu, and Burkhard Stiller. In particular, many thanks are addressed to Katharina O. E. Müller for her strong commitment on getting this technical report ready and quickly published. A larger number of pre-presentation discussions have provided valuable insights in the emerging and moving field of communication systems, both for all groups of students and supervisors. Many thanks to all people contributing to the success of this event, which has happened in a lively group of highly motivated and technically qualified students and people.

Zürich, June 2025

Contents

1 Beyond Passwords – An Overview of Passwordless Authentication Standards and Mechanisms **1**

Sven Greuter, Andri Spescha

2 Exploring New Transport Protocols for the Modern Web **23**

Mete Polat

3 Blockchains beyond Cryptocurrency – Practical Use Cases **47**

Efe Saman, Cagla Ataoglu

4 An Overview of Machine Unlearning for Large Language Models **73**

Jamo Sharif

5 Overview of Crypthographic Techniques in Blockchain Applications **99**

Florian Mattmueller, Jana Joy Anja Muheim

6 Securing the Future — Post Quantum Cryptography in Messenger Systems **123**

Shirley Feng Li Lau, Linn Anna Spitz

7 Overview of the AI Agent Systems and Its Protocols **147**

Erik Avtandilyan, Lukas Sager

8 Fact-Checking with Large Language Models **171**

Fabien Morgan, Yijie Tong

9 Machine Learning in 5G Security – An Overview **195**

Ajeong Shin, Dora Silva, Nasim Nezhadsistani

Chapter 1

Beyond Passwords – An Overview of Passwordless Authentication Standards and Mechanisms

Sven Greuter, Andri Spescha

With password-based authentication becoming increasingly insecure and computers growing more powerful, passwordless authentication has emerged as a viable and secure alternative. This paper examines the passwordless authentication standards while also addressing the security risks associated with both password-based and passwordless approaches. It provides an overview of technologies currently in use as well as those under development. The paper discusses established passwordless standards such as UAF, U2F, and FIDO2, exploring how they are implemented and how they address the shortcomings of traditional authentication methods. In addition, it analyzes the security advantages offered by these systems and how they enhance overall authentication security. Finally, current research and emerging trends in passwordless authentication are reviewed, with a focus on future technologies and evolving standards.

Contents

1.1 Background 3
1.1.1 Authentication and Authorization 3
1.1.2 The Five Authentication Factors 3
1.1.3 Asymmetric Cryptography 3
1.2 Password Authentication 5
1.2.1 Universal Authentication 5
1.2.2 2FA and MFA 5
1.2.3 Hardware Token 6
1.2.4 Single Sign-On 6
1.2.5 OTP Tokens 6
1.2.6 Summarizing Password Authentication 7
1.3 Passwordless Authentication 7
1.3.1 FIDO (Fast IDentity Online) Alliance 7
1.3.2 Overview of FIDO Standards 7
1.3.3 FIDO U2F 8
1.3.4 FIDO UAF 8
1.3.5 FIDO2 9
1.3.6 FIDO2 Registration and Authentication Flow 11
1.3.7 Passkeys 13
1.3.8 Authenticators 13
1.4 Security of Passwordless Authentication 14
1.4.1 Issues with Current Password-Based Methods 15
1.4.2 23andMe Data Breach (2023) 15
1.4.3 Passwordless Advantages and Drawbacks 15
1.4.4 LastPass and Malicious Chrome Extensions 16
1.5 Current Research in Passwordless Authentication 16
1.5.1 Credential Exchange Protocol (CXP) 16
1.5.2 Combining Device-Bound and Synced Passkeys 16
1.5.3 Bluetooth-Based Two-Factor Authentication 17
1.5.4 Behavior-Based Authentication 17
1.6 Summary 17

1.1 Background

This section provides a brief introduction to the background of the topic. First, it defines authentication and outlines the five common factors of authentication as a foundation. Next, it introduces the concept of asymmetric cryptography, which is essential for understanding passwordless authentication, with a particular emphasis on public key infrastructures.

1.1.1 Authentication and Authorization

An authenticated user does not imply that the user is authorized to access every resource or perform every action [25]. Therefore, it is essential to clearly distinguish between authentication and authorization.

OWASP's Authentication Cheat Sheet [24] defines authentication as a process "[...] of verifying that an individual, entity, or website is who or what it claims to be by determining the validity of one or more authenticators (e.g., passwords, fingerprints, or security tokens) that are used to back up this claim". Most common method involve the use of usernames and passwords. This verification process serves as the initial step in every protection procedure [27].

NIST [21] defines authorization as a process "[...] of verifying that a requested action or service is approved for a specific entity". This process grants access to a resource to specific users. Authorization is frequently associated and used interchangeably with access control and client privilege, both with the goal to restrict access of users. Not every user is supposed to execute every action or access every resource. Furthermore, authorization must always come after authentication, because users should first prove their identities, and then an organization's administrator should allow them access to the requested service [27].

1.1.2 The Five Authentication Factors

Passwords and usernames are just one method of authentication, falling under the knowledge factor, which is one of the five common authentication factors. This paper also considers other factors that are possession-based and biometric-based. Table 1.1 lists the five authentication factors [26].

1.1.3 Asymmetric Cryptography

Asymmetric cryptography is one of the key technologies that enables passwordless authentication [27]. This section will briefly go over the basic concept of asymmetric cryptography, as it will play a key role later on.

Table 1.1: Authentication Factors [26].

Factor	***Example***
Something you know	Password
Something you have	OTP/U2F Token
Something you are	Biometrics
Somewhere you are	Source IP Address
Something you do	Behavioral profiling

Asymmetric cryptography relies on the use of a public-private key pair. The key pair is generated through mathematical algorithms, such as RSA or elliptic curve methods, which produce a public and a private key that are computationally linked but infeasible to derive from one another [27]. The public key can then be openly exposed to the public, while the private key must be kept private by the person who generated the keys. These key pairs are then used for encryption or signing.

1.1.3.1 Asymmetric Encryption

In asymmetric encryption, a message is encrypted using the recipient's public key, as seen in Figure 1.1. Only the recipient's corresponding private key can decrypt the message. This ensures that only the recipient is able to decrypt the message and read it. As a result, asymmetric encryption is widely used for securing communications across untrusted networks, such as the Internet [33].

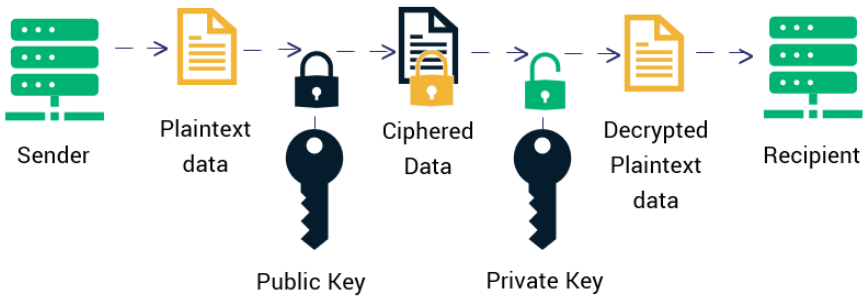


Figure 1.1: Overview of Encryption Using a Public-Private Key Pair [33].

1.1.3.2 Digital Signing

In digital signing, the signer's private key is used to generate a digital signature, and the corresponding public key can then be used to verify it, as seen in Figure 1.2. This ensures that the message originates from the sender and the recipient can verify this. Digital signing is one of the key aspects that make passwordless authentication possible, as it facilitates the verification of the passkey owner's identity [27].

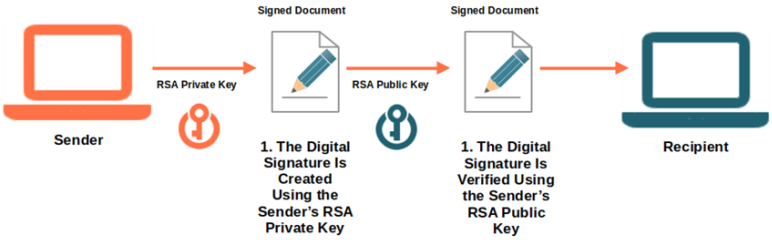


Figure 1.2: Overview of Signing Using a Public-Private Key Pair [29].

1.2 Password Authentication

In this section, an overview of selected password authentication methods is provided, aiming to contextualize their historical significance and highlight the ongoing shift toward passwordless authentication. Additionally, password-based authentication methods are examined in conjunction with their extensions to incorporate other authentication factors. The FIDO U2F standard, even though reliant on passwords, is described in Section 1.3, in order to offer a more intuitive overview of all the FIDO standards.

1.2.1 Universal Authentication

Using usernames and passwords is the simplest and most widely spread authentication implementation. Due to its simplicity, it is easy to use for both users and developers. One aspect is that an in-depth knowledge of cryptography or complex security protocols is not required in order to implement or use passwords. It falls under the something you know factor, requiring the user to be able to remember a password or passphrase in order to log in [27, 23, 40].

Simplicity and ease of deployment are some benefits, however, this simplicity is accompanied by certain drawbacks [27]. For example, users are prone to reuse their passwords or use weak passwords [1]. One reason is the common knowledge, the inherent nature of the knowledge factor, that allows passwords to be shared with others or exploited by adversaries [24].

1.2.2 2FA and MFA

Combining multiple authentication factors, which the users have to provide during log-in, can improve security [26]. The distinction between 2FA (Two Factor Authentication) and MFA (Multi-factor Authentication) is that 2FA is restricted to only one second factor, whereas MFA can use multiple additional factors. While for 2FA or MFA, any other factor of the five authentication factors can be used, most commonly seen are examples of SMSs (Short Message Service), something you have, or biometrics, something you are [27].

While MFA improves security, it cannot be guaranteed. MFA's main purpose is to defend against brute-force attacks, credential stuffing, and password spraying [26]. However, there are many attacks, also in the wild, that comprise some MFA authentications. This can either be done by compromising the implementation, for example by intercepting the SMS with the OTP token, or by phishing the human element [15, 20, 2].

Other downsides include usability concerns for users and implementation disadvantages for developers. One concern is that users might get locked out of their accounts. Implementing MFA might add relationships and external dependencies for developers, which should be taken into account [26, 15].

1.2.3 Hardware Token

In addition to the password, the users have to authenticate themselves with an ownership factor [26, 27]. This is mainly a variant of 2FA, nevertheless worth highlighting on its own, due to the relation to the FIDO standards in subsequent chapters [18]. The tokens for authentication are on a separate physical device. Hence, the adversary needs physical access to the hardware device in order to exploit it. However, a single point of failure might be introduced if the implementation requires a backend server [26].

Smart cards and YubiKeys are some common examples of implementations of hardware tokens. Smart cards are mainly used for operating system authentication. If the card is stolen, it is not usable without a PIN (personal identification number). YubiKeys, which are discussed in more detail in Section 1.3, can be used as hardware tokens as well [26, 37].

1.2.4 Single Sign-On

Single sign-on (SSO) authenticates users with a single action of authentication. This permits access to all related, but independent software systems or applications to the authorized user, without the need to authenticate to each one of them. The service provider application and the identity provider have a trust arrangement on which this service's authentication method is built [28, 27].

SSO can reduce the risks that come with storing passwords locally, but it also comes with new challenges. It reduced the amount of passwords that are stored and the risk for administrators to manage users centrally [28, 27].

1.2.5 OTP Tokens

One-Time Password (OTP) tokens belong to the possession-based authentication factor. An on-demand OTP token can be delivered to the user by email or SMS [26,

27]. Most common are Time-based One Time Password (TOTP) tokens, which can be either hardware or software based [26].

Attacks on OTP authentication include phishing attempts, among other attacks, on the implementation. NIST specifies that OTP tokens, even though they might be short-lived, are not phishing-resistant [22]. Attacks on the implementation consist of exploiting the problem of creating truly random and unpredictable OTPs. An adversary could try to get the seed or try to predict the next token [27, 15].

1.2.6 Summarizing Password Authentication

In this section, a selected overview of password authentication mechanisms was provided. While passwords are easy to manage, they have security issues for the user and developers [27]. With the ubiquity of passwords, multiple attacks were developed to circumvent the authentication or obtain the password. In recent years, one could observe a shift toward moving from password-oriented authentication to passwordless authentication.

1.3 Passwordless Authentication

This section provides an introduction to the FIDO standards and passkeys. First, an overview of the three main FIDO standards is introduced. Then, the authentication mechanism of WebAuthn for UAF and FIDO2 is highlighted. These two standards are also mostly associated with passwordless authentication.

1.3.1 FIDO (Fast IDentity Online) Alliance

The FIDO Alliance provides a set of popular MFA open standards that rely on public-private key cryptography. The alliance consists of a collaboration of multiple countries, such as the USA, China, Europe, and companies, such as Apple, Mastercard, Microsoft, Samsung, and Yubico. They provide specifications with the goal of open, scalable, and interoperable mechanisms in order to decrease the reliance on passwords but still support a wide range of websites, software, and devices. Products and Software that implement up to the standard can be certified by the FIDO Alliance to guarantee interoperability, which is critical for worldwide adoption [11, 39, 4, 15].

1.3.2 Overview of FIDO Standards

There are three main standards: U2F, UAF, and FIDO2. U2F and UAF were introduced first. A few years later, FIDO2 came as an extension of U2F and

combining the benefits and use cases of U2F and UAF [2]. U2F provides a strong cryptographic second factor, but it is not just 2FA because the dependence on the password is reduced [13]. UAF allows for both multi-factor and passwordless authentication [19]. FIDO2 is an evolution of 2FA, offering the same and improved level of security [36]. Both UAF and FIDO2 offer passwordless authentication. U2F and FIDO2 are mainly known for web-based authentication, whereas UAF is mostly known for authentication of mobile applications [2].

1.3.3 FIDO U2F

U2F specifies the presence of a hardware token as an advanced framework for 2FA [2]. It was originally proposed to resolve the security issues of OTP, such as the vulnerabilities of SMS or social engineering, as discussed in Section 1.2.5 [2]. By enabling this strong second factor, the relying party's dependence on passwords is reduced. It can be reduced to a simple 4-digit pin or biometrics [38]. The hardware token is a physical device, for example, a cell phone or USB dongle [15].

Enabling and authenticating with U2F is similar to any 2FA mechanism, which still has a password-based aspect. The users provide username and password in the same way as they usually would with password-based authentication mechanisms, but then they additionally present the physical device either via USB or NFC (Near Field Communication). Thus, the physical device feels like another additional factor to the user [32, 15]. Due to the fact that the users still have to provide the knowledge factor, the password, U2F cannot be regarded as a fully passwordless authentication mechanism [2].

1.3.4 FIDO UAF

The UAF protocol offers both passwordless and multi-factor authentication. It was intended to overcome the issues of multiple passwords. The intended goal is to define a secure, passwordless, authentication scheme [19, 2].

Once users register their device, they simply repeat the local authentication action in order to authenticate. The local authentication action enables the passwordless aspect by utilizing biometric or entering a PIN. Thus, after registering the device, the user's password is no longer needed when authenticating from that device. Therefore, the UAF protocol combines two or more strong authentication factors. For example, when using biometrics on that registered device, the user is authenticating with "something you are" and "something you have". This is one of the goals specified for this protocol [19, 2].

The architecture, illustrated in Figure 1.3, specified in the UAF protocol, consists of five high-level components. The Relying Party (RP) is the application the users want to authenticate against. The User Agent, which is either a browser or a mobile application. The FIDO Client (or UAF client) implements the client side

logic of the protocol and is the interface between the relying party and the user agent on the device. The Authenticator Specific Module (ASM) is the abstraction layer that provides an uniform API. And the FIDO Authenticator is responsible for generating and storing keys associated to a Relying Party. The authenticator needs a specific type (e.g., biometric, PIN), has to provide cryptographic capabilities, and its provenance, which guarantees to the Relying party that the user being authenticated is the user that originally registered with the site [19, 10].

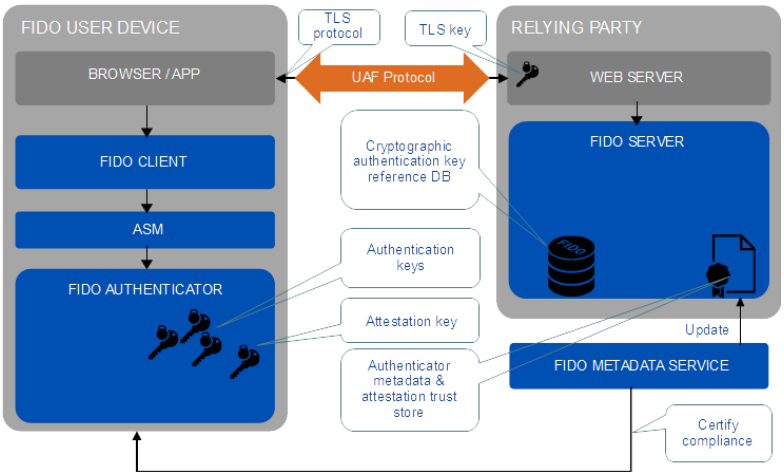


Figure 1.3: UAF Architecture as Illustrated in [19].

The UAF standard cannot yet be considered passwordless, due to the fact that the user still needs to authenticate as usual with the authentication method used so far, which may still be a password-based method, and can add only after that the device that uses the UAF protocol. After registration or authentication the usual way, the user adds the trusted client with the authenticator. Only when a device has been added, the users simply repeat the local authentication action whenever they want to authenticate against that relying party again. Thus, it cannot be considered a passwordless authentication method if it still relies on a password for an initial authentication [32].

In summary, the UAF protocol is a step towards a standardized and passwordless protocol [19]. It still relies on a password-based factor for the initial registration of the physical device; the users can authenticate with their device without having to rely on their password [2].

1.3.5 FIDO2

The FIDO2 standard was introduced a few years after U2F and UAF as an extension of U2F, but combining the benefits and applications of U2F and UAF [2, 36]. Thus, FIDO2 can be used for both 2FA and passwordless authentication [2]. It offers expanded authentication options, including a strong single factor [36].

FIDO2 provides passwordless authentication where a client is authorized to access the authenticator device. This client is bound to the authenticator such that the

authenticator only accepts messages sent by this trusted client. The server does not use passwords at all [4].

The architecture is a simplified version of the UAF architecture with only three components: The Relying Party, the client, and the roaming authenticator. The Relying Party is again the site that is authenticated against. The client is the browser. In FIDO2, due to the WebAuthn protocol, we focus mainly on web-based authentication [2]. The architecture is illustrated in Figure 1.4 as described by [3].

FIDO2 consists of the W3C's (World Wide Web Consortium) Web Authentication standard, WebAuthn, and CTAP2 (Client to Authenticator Protocol) [2, 16]. WebAuthn and APIs are used such that the FIDO Alliance's goal to support a wide range of software, services, and devices might be achieved [15, 2]. The CTAP2 describes the communication between the roaming authenticator and the client on a registered device [16, 7]. This is also contained in Figure 1.4. Figure 1.5 contains both WebAuthn, as the communication between Server and Client, and CTAP2 as the communication between Client and Authenticator.

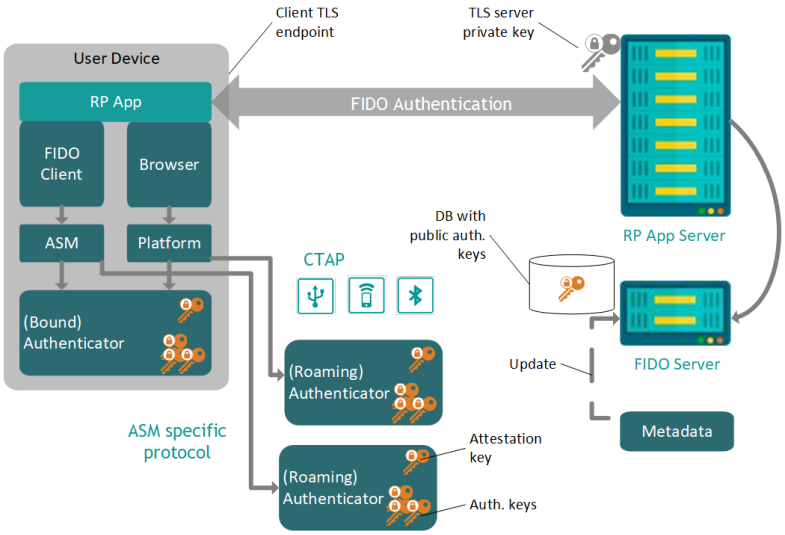


Figure 1.4: FIDO2 Architecture Illustrated in [3, 16].

The goal of the Flow is to register or authenticate a user, (the client), to a server (the relying party). [4] illustrates this flow as seen in Figure 1.5. The relying party sends a challenge to the client. The client forwards this challenge to the authenticator. The authenticator requires a user interaction, called "evidence of user interaction", which can be a user gesture like touching, entering a PIN, biometrics, or other. The response to the challenge is sent back to the client, and the client, adding the original challenge, forwards it to the server. The server then verifies the response [3, 4, 5].

Thus, it is noted that the FIDO2 protocol is similar to the previous two protocols. However, this one simplifies the previous protocols and combines their benefits [2].

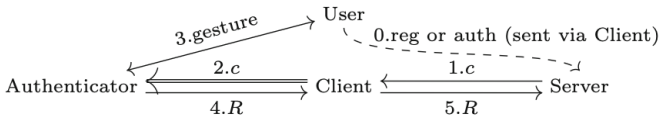


Figure 1.5: FIDO2 Flow Illustrated in [4].

1.3.6 FIDO2 Registration and Authentication Flow

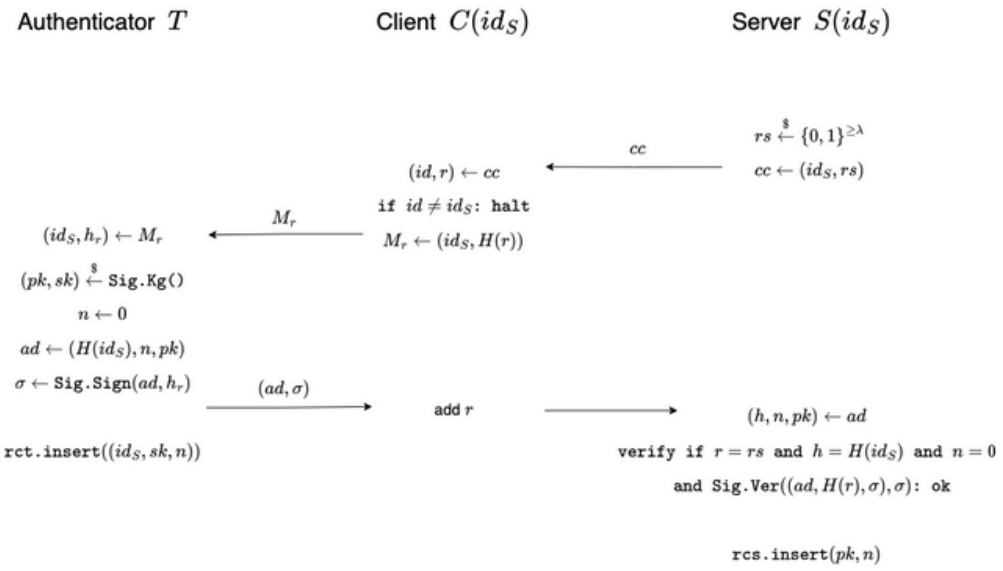


Figure 1.6: Simplified Registration Flow [4, 5].

The registration flow, based on a challenge-response mechanism and composed of WebAuthn and CTAP2, as illustrated in Figure 1.6, begins with a challenge issued by the server, i.e., the relying party, and a response is issued by the client and authenticator [3, 16, 2, 4, 5]. In this first step, the server creates a random string of λ bits or more, stored in rs, which is the challenge. The server's ID and the challenge are packed into cc, which is then sent to the client. The client, who knows the ID of the server (id_S), checks upon unpacking the cc received, if that message really originates from the expected server. This is a simple check to make sure to only accept challenges from the expected server to which the user wants to authenticate against. If the ID is as expected, the client hashes the challenge, $H(r)$, and sends this together with the server ID, id_S, to the authenticator [4, 5, 16].

The authenticator unpacks the server ID and the hashed challenge h_r. Because the client is registered against the server, the authenticator needs to generate a public-private key pair (pk, sk) and initializes the signature counter to zero, $n = 0$, because this is the first one. The attestation signature ad contains the hash of the server ID, $H(id_S)$, the signature counter n, and the public key pk, which is eventually sent to the server. The server needs the $h(id_S)$ to verify that the authenticator responded to the challenge issued by this server. The signature counter is sent in order for the server to check the order and to prevent replay attacks.

And the pk is sent such that the server knows the public key to verify signatures of that authenticator. Besides the ad, the authenticator sends the signature σ over the ad and the hashed challenge h_r. The ad is signed such that it cannot be tampered with on the way to the server, and the hash of the server ID is signed such that the signature can be verified by the correct server. The authenticator saves the server's ID, the private key, and the signature for subsequent authentication challenges. The client adds the original challenge r [4, 5, 16].

The server now validates the received response. It first checks if the received response challenge r is its issued challenge rs. And if the received hash of the server ID h is the hash of its own ID $H(id_S)$. And if the message counter n is zero, indicating a registration attempt. And lastly, if the signature is correct. This is checked by verifying that ad and the hash of the challenge $H(r)$ corresponds to the expected signature σ. $H(r)$ is needed because the authenticator signed the hash of the challenge. If all these things are correct, the client and authenticator are correctly validated and registered for this server, and the server saves the public key pk and signature counter n for subsequent authentication requests [4, 5, 16].

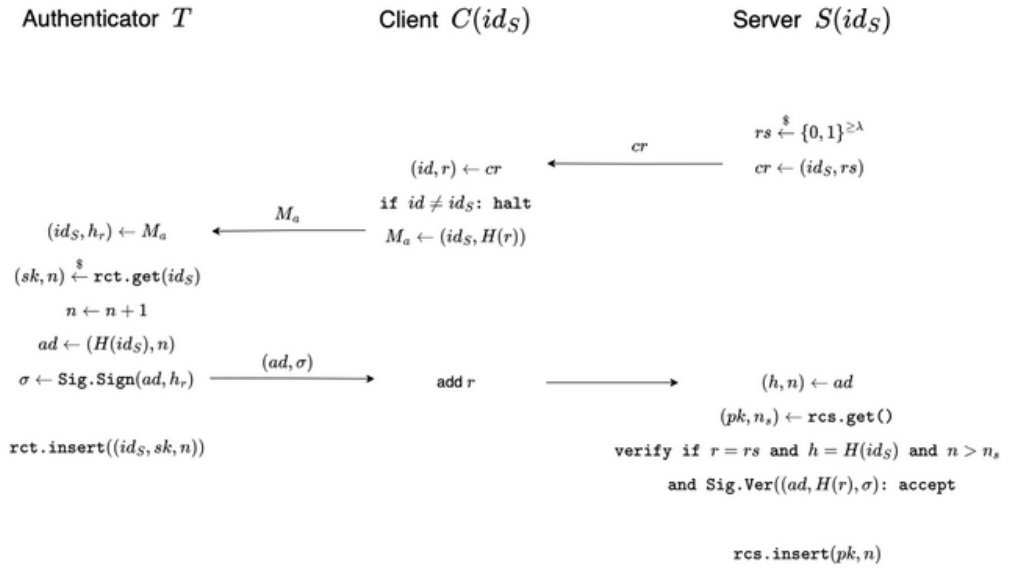


Figure 1.7: Simplified Authentication Flow [4, 5].

The authentication flow, illustrated in Figure 1.7, is very similar to the registration flow. The difference is in the authenticator, which instead of generating a new public-private key pair, now fetches the private key sk and the signature counter n it has saved. The message counter is then increased by one in order to make sure that every request has its own signature number, such that replay attacks might be avoided. This updated signature counter n is then saved in the authenticator for subsequent authentication requests. The signature is calculated in the same way as in the registration flow. The server checks the validity of the signature and attribute values the same way, except that it checks if the signature counter received n is higher than the signature counter the server has saved n_s. Furthermore, the server also saves the updated received signature counter upon successful validation [4, 5, 16].

To conclude, one can observe that the registration and authentication flows are similar and are based on a challenge-response authentication scheme [4, 2]. For simplicity, the user interaction is left out here in these illustrations [3]. Regardless, given that scheme, it can be observed that this registration and authentication mechanism can be used as a passwordless scheme [4].

1.3.7 Passkeys

Passkeys are the data that is needed to log into different services; they include data such as the private key needed for signing, the account name for that service, and details about that service. This can sometimes include even more data; this depends on the service, however [8].

These passkeys can be stored in many different ways to help with accessibility, but this can come at the cost of overall security [14]. We can differentiate between four different ways of storing these passkeys, as displayed in Figure 1.8 [8].

In the single-user context, it can be differentiated between device-bound and synced passkeys. With device-bound passkeys, the passkey itself remains at all times locally on the device and is never sent or stored anywhere else. This is the most secure way to store passkeys [8].

However, synced passkeys can be used on multiple devices and can be stored either cloud-synced or on specific devices. This will be further discussed in Section 1.3.8. This increased accessibility introduces additional security risks, as passkeys are no longer stored locally but are instead uploaded to a cloud or other devices [8, 14].

In the multi-user context, there are shared passkeys. These are usually cloud-based and shared with different users [8]. This exposes the risk that if one of these users gets compromised, all other users may get compromised as well [8, 14].

There are also exported passkeys. These are usually stored in CSV files and are not encrypted [40]. This is an unreliable and insecure way of storing passkeys and is usually only used in backups and transfers of passkeys [8].

1.3.8 Authenticators

Authenticators are the component which actually store the passkey and perform the cryptographic challenges needed for authentication. When logging in with a passkey, the authenticator is responsible for signing the challenge sent by the server using the private key [4].

There are different types of authenticators, each with its own use cases and security advantages and concerns. Listed below are a few examples of the most widely used ones [40]:

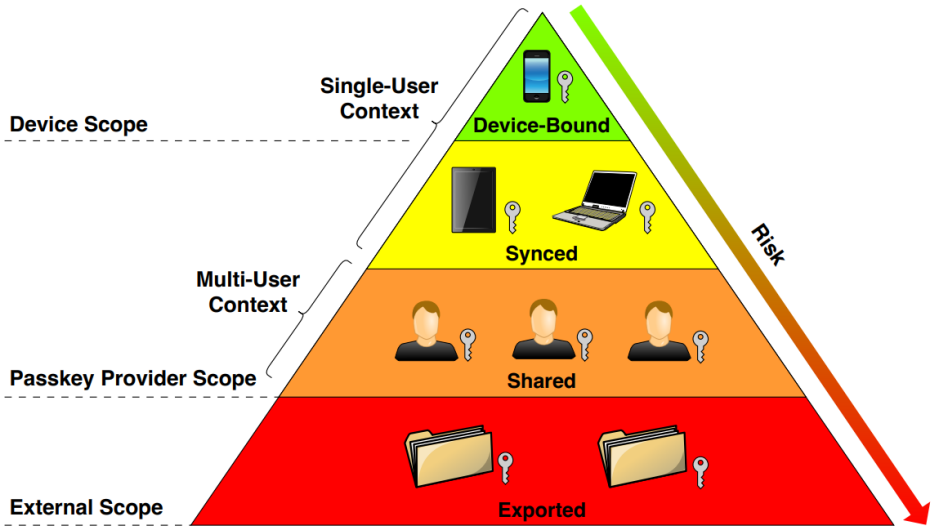


Figure 1.8: The Four Different Ways of Storing Passkeys [8].

- **USB-based Hardware Keys:** Devices like YubiKeys are USB keys and are inserted when authentication is needed. They store the private key securely and often require user authentication, such as a fingerprint, to complete authentication.

- **NFC (Near Field Communication) Smart Cards:** These are physical cards that communicate with devices using NFC. They are often used for enterprise or governmental use, since they require a specific device to read the NFC Card.

- **Inbuilt Biometrics:** These are integrated into modern devices and use secure hardware like the TPM (Trusted Platform Module) or Secure Enclave. Authentication involves some sort of password or biometric verification, such as fingerprint or facial recognition.

- **Cloud-Synced Authenticators:** Cloud-based services like Apple Keychain or Google Password Manager store passkeys in the cloud and allow you to use them on multiple devices. While this increases convenience and availability, it introduces risks related to cloud storage security [8].

1.4 Security of Passwordless Authentication

Security is one of the main motivations behind the push for passwordless authentication. Traditional password-based systems have shown significant weaknesses, both in theory and in real-world breaches. This section discusses why and where current systems fall short and how passwordless methods aim to improve them.

1.4.1 Issues with Current Password-Based Methods

Passwords are inconvenient for most users. They are hard to remember, and users often write them down, reuse them across platforms, or choose weak ones to avoid forgetting them. This creates a major security problem, as reused or easily guessed passwords can be exploited [27]. Attacks such as credential stuffing take advantage of this behavior by using leaked credentials from one service to log into others [9]. Another concern is that password databases and password managers are common targets for hackers, and once breached, attackers may gain access to millions of credentials at once. Even two-factor authentication methods like SMS or email-based one-time passwords are vulnerable to man-in-the-middle attacks and interception [2, 15, 20]. Social engineering also remains a major threat in traditional systems since users can be tricked into giving up their credentials even when two-factor authentication is enabled [30, 27]. Usability also starts to suffer with two-factor authentication taking up time, and frequent password resetting requests annoying users. This causes users to try and avoid these methods or go around them, further causing security issues [30].

1.4.2 23andMe Data Breach (2023)

A major incident that highlights the dangers of password reuse is the 2023 data breach at 23andMe. Attackers used credential stuffing techniques to access around 6.9 million user accounts. This was possible because many users reused the same credentials across different platforms. Once inside, the attackers were able to access sensitive personal and even genetic data. In the aftermath, 23andMe attributed the breach to poor password hygiene among users and implemented two-factor authentication only after the incident had already caused damage [9].

1.4.3 Passwordless Advantages and Drawbacks

Passwordless authentication attempts to resolve many of the above-mentioned security issues. It removes the need for users to create or remember passwords, eliminating the risks of password reuse and weak credentials. Because users no longer store or manage passwords, there's less chance of them being tricked into revealing anything, reducing the effectiveness of phishing and social engineering attacks [27, 30, 4]. Additionally, many passwordless systems don't rely on central storage, making large-scale database breaches less likely. Authentication mechanisms that rely on asymmetric cryptography are also far more resistant to brute-force attacks than traditional password-based logins [27, 40].

Still, passwordless methods aren't without challenges. When passkeys are synced via cloud services or exported as unencrypted backups, they can become exposed if the cloud provider is compromised or if the file is not securely handled. The security of passwordless systems also heavily depends on the security of the user's

device [8]. If a device is mismanaged with few security options enabled, infected with malware, or if it gets stolen, the user's credentials may still be at risk. There is also a psychological barrier: Studies show that many users perceive passwordless systems as less secure, which could slow down adoption [30, 40]. Additionally, if a user loses their authentication device and hasn't set up a secure backup or recovery option, regaining access can be difficult.

1.4.4 LastPass and Malicious Chrome Extensions

Even solutions designed to manage credentials securely can be vulnerable. In 2022, LastPass suffered a significant breach where attackers gained access to the development environment and were later able to retrieve encrypted vaults containing user credentials. Though the vaults were encrypted, attackers could still attempt offline brute-force attacks [34, 6]. Another area of concern is malicious browser extensions, which can mimic trusted password managers and trick users into storing or autofilling credentials into fake interfaces [31, 35]. These kinds of attacks could apply to both traditional passwords and passwordless credentials like passkeys.

1.5 Current Research in Passwordless Authentication

As passwordless authentication becomes more widely adopted, new technologies are emerging to improve security and usability. This section presents an overview of some of the most recent developments in passwordless authentication.

1.5.1 Credential Exchange Protocol (CXP)

One of the current problems with authenticators and passkeys is the lock-in effect, where users are tied to a specific vendor. The Credential Exchange Protocol (CXP), in development by the FIDO Alliance, aims to address this issue by allowing secure and standardized transfer of credentials between providers. This removes the need for insecure formats like CSV files and gives users greater flexibility in managing their passkeys. This is still under heavy development and not yet fully released [17].

1.5.2 Combining Device-Bound and Synced Passkeys

The Fido Alliance is also looking at combining device-bound and cloud-synced passkeys in order to increase, as visible in Figure 1.9. With this method, the passkey is only usable when both a device-bound passkey and a cloud-synced passkey are available. This not only improves security but also improves recovery options in case a device-bound passkey is lost [12].

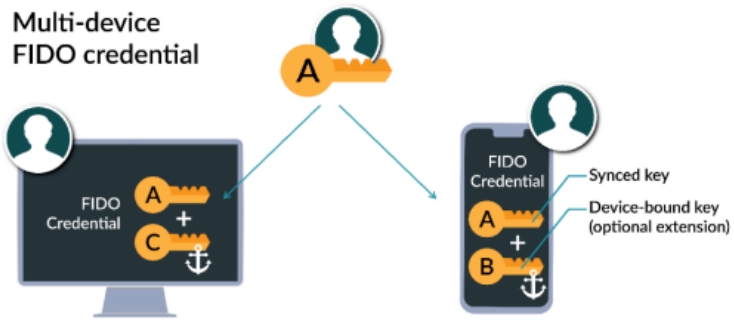


Figure 1.9: An Overview of Combining Different Passkey Storage Methods [12].

1.5.3 Bluetooth-Based Two-Factor Authentication

Another emerging trend involves smartphones being used as authenticators for Bluetooth-based two-factor authentication. In this idea, the smartphone would pose as a roaming authenticator, while the user's computer acts as the device requesting authentication. The authentication process occurs via Bluetooth, allowing for a seamless and secure user experience [12, 27]. This basically mirrors the same way of authenticating as NFC-based passwordless authentication.

1.5.4 Behavior-Based Authentication

Behavior-based authentication is currently only a research-based technology that analyzes a user's behavior patterns, such as typing, Wi-Fi connections, mouse movements, and other habits, to validate a user's identity. Although still in early development, this approach could be valuable for high-security environments. However, it also raises privacy concerns and must be carefully implemented to avoid intrusive user experiences [40].

1.6 Summary

This paper demonstrates that passwordless authentication, particularly by exploring the implementation of FIDO UAF and FIDO2, provides viable alternatives to traditional password-based systems. Passwordless systems deal with many of the flaws and security issues of current systems. FIDO U2F introduces hardware tokens for enhanced 2FA and FIDO UAF focuses on passwordless authentication through local actions like biometrics or PINs. FIDO2, by combining WebAuthn and CTAP2, builds on those protocols, combining their strengths into a unified, secure, and user-friendly authentication standard, that is both resistant to common attacks and flexible enough to work across many different devices and platforms.

While passwordless authentication has significant improvements in security and usability, it also introduces new challenges. The reliance on users keeping their device secure, the need for effective backup and recovery strategies, and the complexity of cross-device and cloud-based passkey management all represent areas where further refinement and user education are needed. However, security breaches, as exemplified in the paper, show the limitations of password-based systems and the need to transition to more secure methods like passwordless authentication.

Ultimately, this paper argues that passwordless authentication is not only a viable successor to password-based systems but a necessary and feasible evolution. With continued development around multi-device integration and alternative verification methods like behavioral authentication, passwordless solutions are well on track to becoming the new standard in authentication.

References

[1] Tolga Acar, Mira Belenkiy, and Alptekin Küpçü. "Single password authentication". In: *Computer Networks* 57.13 (2013), pp. 2597–2614. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2013.05.007. URL: https://www.sciencedirect.com/science/article/pii/S1389128613001667.

[2] Anna Angelogianni, Ilias Politis, and Christos Xenakis. "How many FIDO protocols are needed? Analysing the technology, security and compliance". en. In: *ACM Computing Surveys* 56.8 (Aug. 2024), pp. 1–51. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3654661. URL: https://dl.acm.org/doi/10.1145/3654661 (visited on 03/20/2025).

[3] Davit Baghdasaryan et al. *FIDO Security Reference*. Accessed: 2025-04-21. URL: %5Curl%7Bhttps://fidoalliance.org/specs/fido-v2.0-id-20180227/fido-security-ref-v2.0-id-20180227.html%7D.

[4] Manuel Barbosa et al. "Provable Security Analysis of FIDO2". en. In: *Advances in Cryptology - CRYPTO 2021*. Ed. by Tal Malkin and Chris Peikert. Vol. 12827. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, pp. 125–156. DOI: 10.1007/978-3-030-84252-9_5. URL: https://link.springer.com/10.1007/978-3-030-84252-9_5 (visited on 03/20/2025).

[5] Nina Bindel, Cas Cremers, and Mang Zhao. "FIDO2, CTAP 2.1, and WebAuthn 2: Provable Security and Post-Quantum Instantiation". In: *2023 IEEE Symposium on Security and Privacy (SP)*. San Francisco, CA, USA: IEEE, May 2023, pp. 1471–1490. DOI: 10.1109/SP46215.2023.10179454. URL: https://ieeexplore.ieee.org/document/10179454/ (visited on 03/20/2025).

[6] Matt Binder. *Password managers are under threat in 2025. What the LastPass breach taught us.* Accessed: 2025-05-16. URL: %5Curl%7Bhttps://mashable.com/article/password-manager-breaches-lastpass-lessons-learned%7D.

[7] John Bradley et al. *Client to Authenticator Protocol (CTAP)*. Accessed: 2025-04-21. URL: %5Curl%7Bhttps://fidoalliance.org/specs/fido-v2.1-ps-20210615/fido-client-to-authenticator-protocol-v2.1-ps-errata-20220621.html%7D.

[8] Andre Büttner and Nils Gruschka. *Device-Bound vs. Synced Credentials: A Comparative Evaluation of Passkey Authentication*. arXiv:2501.07380 [cs]. Jan. 2025. DOI: 10.48550/arXiv.2501.07380. URL: http://arxiv.org/abs/2501.07380 (visited on 03/20/2025).

[9] Mack DeGeurin. *Hackers got nearly 7 million people's data from 23andMe*. Accessed: 2025-05-16. URL: %5Curl%7Bhttps://www.theguardian.com/technology/2024/feb/15/23andme-hack-data-genetic-data-selling-response%7D.

[10] Haonan Feng et al. "A Formal Analysis of the FIDO UAF Protocol". en. In: *Proceedings 2021 Network and Distributed System Security Symposium*. Virtual: Internet Society, 2021. DOI: 10.14722/ndss.2021.24363. URL: https://www.ndss-symposium.org/wp-content/uploads/ndss2021_4A-2_24363_paper.pdf (visited on 03/20/2025).

[11] FIDO Alliance. *Alliance Overview*. Accessed: 2025-04-10. URL: %5Curl%7Bhttps://fidoalliance.org/overview/%7D.

[12] FIDO Alliance. *How FIDO Addresses a Full Range of Use Cases*. Accessed: 2025-04-29. URL: %5Curl%7Bhttps://fidoalliance.org/white-paper-multi-device-fido-credentials%7D.

[13] FIDO Alliance. *Universal 2nd Factor (U2F) Overview*. Accessed: 2025-04-11. URL: %5Curl%7Bhttps://fidoalliance.org/specs/u2f-specs-master/fido-u2f-overview.html%7D.

[14] Raul Gonzalez, Eric Y. Chen, and Collin Jackson. *Automated Password Extraction Attack on Modern Password Managers*. arXiv:1309.1416 [cs]. Sept. 2013. DOI: 10.48550/arXiv.1309.1416. URL: http://arxiv.org/abs/1309.1416 (visited on 03/20/2025).

[15] Roger A. Grimes. *Hacking multifactor authentication*. eng. Indianapolis, IN: John Wiley & Sons, Inc, 2021.

[16] Eric Klieme et al. "FIDOnuous: A FIDO2/WebAuthn Extension to Support Continuous Web Authentication". In: *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. Guangzhou, China: IEEE, Dec. 2020, pp. 1857–1867. DOI: 10.1109/TrustCom50675.2020.00254. URL: https://ieeexplore.ieee.org/document/9343231/ (visited on 03/20/2025).

[17] René Léveillé et al. *Credential Exchange Specifications*. Accessed: 2025-04-29. URL: %5Curl%7Bhttps://fidoalliance.org/specs/cx/cxf-v1.0-rd-20250313.html%7D.

[18] Shanshan Li et al. "A Secure Two-Factor Authentication Scheme From Password-Protected Hardware Tokens". In: *IEEE Transactions on Information Forensics and Security* 17 (2022), pp. 3525–3538. ISSN: 1556-6013, 1556-6021. DOI: 10.1109/TIFS.2022.3209886. URL: https://ieeexplore.ieee.org/document/9903479/ (visited on 03/20/2025).

[19] Salah Machani et al. *FIDO UAF Architectural Overview*. Accessed: 2025-04-11. URL: %5Curl%7Bhttps://fidoalliance.org/specs/fido-uaf-v1.1-id-20170202/fido-uaf-overview-v1.1-id-20170202.html%7D.

[20] Ali Hameed Yassir Mohammed, Rudzidatul Akmam Dziyauddin, and Liza Abdul Latiff. *Current Multi-factor of Authentication: Approaches, Requirements, Attacks and Challenges*. en. 2023. DOI: 10.14569/IJACSA.2023.0140119. (Visited on 03/20/2025).

[21] NIST. *Glossary: authorization*. Accessed: 2025-04-26. URL: %5Curl%7Bhttps://csrc.nist.gov/glossary/term/authorization%7D.

[22] NIST. *SP 800-63B: 3. Authenticator and Verifier Requirements*. Accessed: 2025-04-10. URL: %5Curl%7Bhttps://pages.nist.gov/800-63-4/sp800-63b/authenticators/%5C#singlefactorOTP%7D.

[23] Obi Ogbanufe and Dan J. Kim. "Comparing fingerprint-based biometrics authentication versus traditional authentication methods for e-payment". en. In: *Decision Support Systems* 106 (Feb. 2018), pp. 1–14. ISSN: 01679236. DOI: 10.1016/j.dss.2017.11.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167923617302154 (visited on 03/20/2025).

[24] OWASP. *Authentication Cheat Sheet*. Accessed: 2025-04-08. URL: %5Curl%7Bhttps://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html%7D.

[25] OWASP. *Authorization Cheat Sheet*. Accessed: 2025-04-26. URL: %5Curl%7Bhttps://cheatsheetseries.owasp.org/cheatsheets/Authorization_Cheat_Sheet.html%7D.

[26] OWASP. *Multifactor Authentication Cheat Sheet*. Accessed: 2025-04-08. URL: %5Curl%7Bhttps://cheatsheetseries.owasp.org/cheatsheets/Multifactor_Authentication_Cheat_Sheet.html%7D.

[27] Viral Parmar et al. "A Comprehensive Study on Passwordless Authentication". In: *2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*. Erode, India: IEEE, Apr. 2022, pp. 1266–1275. DOI: 10.1109/ICSCDS53736.2022.9760934. URL: https://ieeexplore.ieee.org/document/9760934/ (visited on 03/20/2025).

[28] V. Radha and D. Hitha Reddy. "A Survey on Single Sign-On Techniques". en. In: *Procedia Technology* 4 (2012), pp. 134–139. ISSN: 22120173. DOI: 10.1016/j.protcy.2012.05.019. URL: https://linkinghub.elsevier.com/retrieve/pii/S2212017312002988 (visited on 03/20/2025).

[29] Savvy Security. *What Is the RSA Algorithm? A Look at RSA Encryption*. Accessed: 2025-04-29. 2020. URL: https://cheapsslsecurity.com/blog/what-is-the-rsa-algorithm-a-look-at-rsa-encryption/.

[30] Sharath Chandra Thurupati. "The Usability Paradigm in Passwordless Authentication : A Comparative Study of YubiKey and Passkey Implementations". In: *International Journal of Scientific Research in Computer Science, Engineering and Information Technology* 10.6 (Nov. 2024), pp. 12–21. ISSN: 2456-3307. DOI: 10.32628/CSEIT24106152. URL: https://ijsrcseit.com/index.php/home/article/view/CSEIT24106152 (visited on 02/26/2025).

[31] SquareX. *Polymorphic Extensions: The Sneaky Extension That Can Impersonate Any Browser Extension*. Accessed: 2025-05-16. URL: %5Curl%7Bhttps://labs.sqrx.com/polymorphic-extensions-dd2310006e04%7D.

[32] Sampath Srinivas et al. *Universal 2nd Factor (U2F) Overview - FIDO-U2F-COMPLETE-v1.2-ps-20170411.pdf*. Accessed: 2025-04-12. URL: %5Curl%7Bhttps://fidoalliance.org/specs/fido-u2f-v1.2-ps-20170411/FIDO-U2F-COMPLETE-v1.2-ps-20170411.pdf%7D.

[33] The SSL Store. *ECDSA vs RSA: Everything You Need to Know*. Accessed: 2025-04-29. 2020. URL: %5Curl%7Bhttps://sectigostore.com/blog/ecdsa-vs-rsa-everything-you-need-to-know/%7D.

[34] Karim Toubba. *12-22-2022: Notice of Security Incident*. Accessed: 2025-05-16. URL: %5Curl%7Bhttps://blog.lastpass.com/posts/notice-of-recent-security-incident%7D.

[35] Bill Toulas. *Malicious Chrome extensions can spoof password managers in new attack*. Accessed: 2025-05-16. URL: %5Curl%7Bhttps://www.bleepingcomputer.com/news/security/malicious-chrome-extensions-can-spoof-password-managers-in-new-attack/%7D.

[36] Yubico. *FIDO2 passwordless authentication*. Accessed: 2025-04-11. URL: %5Curl%7Bhttps://www.yubico.com/authentication-standards/fido2/%7D.

[37] Yubico. *How the YubiKey works*. Accessed: 2025-04-08. URL: %5Curl%7Bhttps://www.yubico.com/products/how-the-yubikey-works/%7D.

[38] Yubico. *What is FIDO U2F?* Accessed: 2025-04-12. URL: %5Curl%7Bhttps://www.yubico.com/authentication-standards/fido-u2f-standard/%7D.

[39] Yubico. *What is the FIDO Alliance?* Accessed: 2025-04-10. URL: %5Curl%7Bhttps://www.yubico.com/resources/glossary/fido-alliance/%7D.

[40] Mohd Imran Md Yusop et al. "Advancing Passwordless Authentication: A Systematic Review of Methods, Challenges, and Future Directions for Secure User Identity". In: *IEEE Access* 13 (2025), pp. 13919–13943. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2025.3528960. URL: https://ieeexplore.ieee.org/document/10839395/ (visited on 02/25/2025).

Chapter 2

Exploring New Transport Protocols for the Modern Web

Mete Polat

The limitations of traditional Internet transport protocols, particularly Transmission Control Protocol (TCP) and User Datagram Protocol (UDP), have been increasingly exposed by the evolving requirements of modern web applications. In this paper, recent advances in next-generation transport protocols—including Quick UDP Internet Connections (QUIC) and Multipath TCP (MPTCP)—as well as related application-layer protocols such as Hypertext Transfer Protocol Version 3 (HTTP/3), Multiplexed Application Substrate over QUIC Encryption (MASQUE), and WebTransport, are reviewed. The technical foundations, performance analysis, and deployment experiences of these protocols are examined based on findings from existing literature and empirical studies. It is shown that protocols such as QUIC and HTTP/3 can reduce latency, improve reliability, and enhance user experience, especially under conditions of high latency or network loss. MPTCP and MASQUE are described as providing additional benefits in multipath and tunneling scenarios, although challenges related to middlebox compatibility and deployment persist. Performance trade-offs are discussed, including the potential for increased overhead in high-speed networks. Finally, the ongoing evolution of these protocols and their impact on web performance, security, and adaptability are highlighted, with an outlook provided on future research and deployment directions.

Contents

2.1 Introduction and Motivation 25
2.2 Background 25
2.2.1 Evolution of Transport Protocols 26
2.2.2 Hypertext Transfer Protocol Version 1.1 and 2 26
2.2.3 Previous Attempts 27
2.3 Modern Transport Layer and Application Layer Protocols . . . 27
2.3.1 Transport Layer Security 1.3 (TLS 1.3) 27
2.3.2 Multipath TCP (MPTCP) 29
2.3.3 Quick UDP Internet Connections (QUIC) 29
2.3.4 Hypertext Transfer Protocol Version 3 (HTTP/3) . . . 30
2.3.5 Multiplexed Application Substrate over QUIC Encryption 30
2.3.6 WebTransport 31
2.4 Evaluation and Discussion 31
2.4.1 MPTCP Performance and Adoption 31
2.4.2 QUIC Performance 34
2.4.3 QUIC Adoption 36
2.4.4 MASQUE Performance and Adoption 37
2.4.5 WebTransport Performance 38
2.4.6 Summary 39
2.5 Summary and Conclusion 39
2.5.1 Key Findings 39
2.5.2 Challenges and Ongoing Work 41
2.5.3 Future Outlook 42

2.1 Introduction and Motivation

The rapid growth of internet usage and feature-rich web applications has created a demand for faster, more reliable, and more secure transport protocols. Traditional transport protocols such as the Transmission Control Protocol (TCP) and User Datagram Protocol (UDP), while foundational, are beginning to struggle with the needs of modern web traffic [14, 5]. Issues such as high latency (especially over long distances or mobile networks), frequent network switching on mobile devices, and increased security and privacy requirements expose the limitations of traditional protocols [19]. TCP provides reliable and ordered delivery, but incurs connection setup delays due to the three-way handshake and can suffer head-of-line blocking when multiple streams share one connection [8]. UDP offers low-overhead, connectionless communication, but without built-in reliability or congestion control [23]. These limitations make it challenging to meet today's performance expectations for web browsing, video streaming, real-time applications, and other latency-sensitive services.

Researchers and engineers have recognized the need for more efficient, secure, and adaptive transport solutions to overcome these challenges [19, 21, 34]. This has led to the development of new transport protocols like Quick UDP Internet Connections (QUIC) [14] and Multipath TCP (MPTCP) [10], as well as application-layer protocols and improvements such as Hypertext Transfer Protocol Version 3 (HTTP/3) [5], Multiplexed Application Substrate over QUIC Encryption (MASQUE) [26], WebTransport [31], and Transport Layer Security 1.3 (TLS 1.3) [24]. These next-generation protocols aim to reduce latency (e.g., via zero-round-trip handshakes) [19, 34], improve throughput (e.g., by using multiple network paths or parallel streams) [32, 17, 6], and enforce security by default (encryption of transport headers and payloads) [19, 34]. The following sections introduce these emerging protocols, discuss how they work, and discuss their benefits and trade-offs in modern web infrastructure.

2.2 Background

This section provides the necessary background for understanding the motivations and context behind modern transport protocol development. First, it reviews the evolution of Internet transport protocols, highlighting the strengths and limitations of foundational protocols like TCP and UDP. Next, it examines how application protocols such as HTTP/1.1 and HTTP/2 have adapted to these transport-layer constraints. Finally, it surveys previous attempts to address these limitations through new protocol designs and enhancements. Together, these sections lay the groundwork for understanding the challenges that led to recent advances in Internet transport protocols.

2.2.1 Evolution of Transport Protocols

The history of Internet transport protocols dates back to the Advanced Research Projects Agency Network (ARPANET), which used the Network Control Program (NCP) before the Transmission Control Protocol over the Internet Protocol (TCP/IP) was standardized in the 1980s [16]. TCP and UDP have been the workhorses of Internet transport for decades. The strength of TCP lies in the reliable in-order delivery with control mechanisms, making it suitable for most applications that require accuracy [8]. However, TCP was designed in an era with different requirements; its strict ordering causes head-of-line (HoL) blocking and its connection setup and teardown add latency [16]. Head-of-line blocking occurs when the loss or delay of a single packet prevents the delivery of all subsequent data, even if those later packets have already arrived. In TCP, data must be delivered in order. Therefore, any lost packet forces the receiver to wait for retransmission before it can process later packets, causing potentially significant delays in high-latency or lossy networks [21, 16]. UDP, on the other hand, provides a minimal message-passing service with no reliability guarantees or congestion control, which gives it flexibility and low latency at the cost of potential packet loss [23]. These characteristics mean UDP is often used for real-time media where occasional loss is acceptable, whereas TCP is used for web page loads, file transfers, and other tasks needing reliability. As the Internet evolved, the shortcomings of using TCP and UDP "as-is" for all purposes became evident, spurring research into alternatives [16].

2.2.2 Hypertext Transfer Protocol Version 1.1 and 2

At the application layer, the HTTP protocol has also evolved to mitigate transport limitations. HTTP/1.1 introduced persistent connections and pipelining, but due to the lack of multiplexing, browsers often opened multiple TCP connections to fetch resources in parallel. This was inefficient for congestion control and could overwhelm networks [9]. Google's experimental SPDY protocol, which was introduced around 2009, demonstrated that multiplexing multiple HTTP streams within one connection and compressing request headers could greatly improve page load times [2]. SPDY became the basis for HTTP/2, which was then standardized in 2015. HTTP/2 uses a single TCP connection to carry many parallel streams with binary framing and HPACK header compression. This solves the problem of multiple TCP connections. However, HTTP/2's multiplexing occurs at the application layer and is still ultimately constrained by TCP's behavior. In particular, if a TCP packet is lost, all HTTP/2 streams in that connection are blocked until the packet is retransmitted, since TCP delivers data in order. Thus, a packet loss causes all streams to experience a stall, even if only one is lost [3]. This residual head-of-line blocking at the transport layer limits HTTP/2's effectiveness.

2.2.3 Previous Attempts

Over the years, various new transport protocols were proposed to address TCP's rigidity. The Stream Control Transmission Protocol (SCTP) is one notable example. SCTP was designed with features like multi-streaming (independent ordering per stream to avoid head-of-line blocking between streams) and multi-homing (a connection, called an association in SCTP, can span multiple network interface addresses for redundancy) [28]. SCTP employs a four-way handshake (with a cookie mechanism to protect against spoofed connection requests) and can seamlessly fail over to an alternate path if one network path drops [28]. Despite these advantages and being a standardized protocol, SCTP saw limited adoption in the web ecosystem; middleboxes and NATs often did not support SCTP, and it was not enabled by default in most client operating systems or browsers [32, 19]. Another line of work attempted to improve TCP itself via new congestion control algorithms like Cubic or Bottleneck Bandwidth and Round-trip propagation time (BBR), and extensions like TCP Fast Open, but these still operate within TCP's fundamental design [6, 34]. SPDY (and subsequently HTTP/2) demonstrated that application-layer improvements have limited effect if the underlying transport layer remains unchanged [21]. These experiences set the stage for more radical rethinking of transport protocols, leading to the development of QUIC and MPTCP as well as related efforts in encryption and tunneling protocols [19, 32]. Below, the key approaches that have emerged are outlined.

2.3 Modern Transport Layer and Application Layer Protocols

This section surveys the most significant recent developments at both the transport and application layers of the Internet protocol stack. Table 2.1 summarizes the key properties, advantages, and deployment status of each protocol discussed in this section, providing a comparative overview before the individual protocols are explained in detail. The discussion begins with the major security and performance improvements of TLS 1.3, then examines transport protocol innovations like MPTCP and QUIC. Building on these, the section explores new application protocols such as HTTP/3, as well as advanced frameworks like MASQUE and WebTransport, which extend flexibility and performance for modern web use cases. Together, these protocols represent the state of the art in secure, efficient, and adaptable Internet communication.

2.3.1 Transport Layer Security 1.3 (TLS 1.3)

TLS 1.3 is not a transport protocol by itself, but rather the latest version of the ubiquitous security protocol that underpins HTTPS (and by extension, HTTP/2, HTTP/3, QUIC encryption, etc.). TLS 1.3, finalized in 2018, is a major overhaul

Table 2.1: Summary of Modern Transport Layer and Application Layer Protocols.

Protocol	**Layer**	**Key Features**	**Deployment Status**
TLS 1.3	Session / Presentation	1-RTT (and 0-RTT) handshake, stronger encryption, improved privacy, reduced latency	Widely deployed (default for modern HTTPS)
MPTCP	Transport	Multipath connectivity, seamless failover, bandwidth aggregation, backward-compatible with TCP	Niche (Apple/iOS, Linux), limited elsewhere
QUIC	Transport	Multiplexing, 0/1-RTT handshake, head-of-line blocking elimination, built-in encryption, connection migration	Broad (Google, Facebook, Cloudflare, browsers)
HTTP/3	Application	Runs atop QUIC, eliminates transport HoL blocking, faster setup, encrypted by default	Rapid adoption (browsers, CDNs, large sites)
MASQUE	Application	Tunneling (VPN, proxy) over QUIC, multiplexed streams and datagrams, minimal overhead	Emerging (Cloudflare WARP, draft RFCs)
WebTransport	Application	Multiplexed bidirectional streams/datagrams in browser JS, low-latency, runs over HTTP/3	Early adoption

of TLS with an emphasis on both security and performance improvements. It removes outdated or insecure cryptographic algorithms and modes (such as static RSA key exchange and old ciphers), and simplifies the protocol state machine to avoid many historical vulnerabilities. Critically, TLS 1.3 reduces handshake latency compared to TLS 1.2: a full handshake in TLS 1.3 requires only one round-trip between client and server (1-RTT) versus two in TLS 1.2, and it also supports zero-round-trip (0-RTT) mode for repeat connections, where a client can send data immediately without waiting at all [24]. This means faster connection setup for secure sessions. TLS 1.3 also encrypts more of the handshake itself, providing better privacy, since observers cannot see as much metadata during setup [24]. In summary, TLS 1.3 improves user privacy and reduces handshake latency [24], making it an important complement to protocols like QUIC and HTTP/3 which mandate encryption. Because TLS 1.3 is often embedded in these new transport protocols, its improvements directly translate into faster and safer transport-layer connections. For example, QUIC integrates TLS 1.3 directly into its handshake.

2.3.2 Multipath TCP (MPTCP)

MPTCP is an extension of TCP that enables a single transport connection to use multiple network paths simultaneously [10]. The goal of MPTCP is to improve reliability and aggregate bandwidth by pooling resources across multiple interfaces (e.g. a smartphone's Wi-Fi and cellular data). To an application, an MPTCP connection still looks like a normal single TCP socket [10]—no modifications are required at the application layer to benefit from multipath capabilities. Under the hood, however, MPTCP can establish additional TCP subflows (additional TCP connections) between the same two hosts over different network interfaces or IP addresses, and coordinate data transfer over all subflows. By doing so, MPTCP can seamlessly move traffic to an alternate path if one path fails, and potentially use all paths in parallel for greater throughput. This makes it attractive for mobile devices that frequently switch networks or for scenarios where using both Wi-Fi and cellular together can boost performance or provide failover continuity [10].

2.3.3 Quick UDP Internet Connections (QUIC)

QUIC is a new transport protocol initially developed by Google and later standardized by the IETF [14]. QUIC operates over UDP but provides a rich set of transport features in user space. It was designed to tackle longstanding TCP issues like slow handshake setup, lack of parallel streams, and poor support for mobility. QUIC is a message-oriented, multiplexed, and secure transport protocol: it allows many concurrent streams of data within a single connection, with independent flow control for each stream [14]. It integrates encryption (built-in TLS 1.3 for confidentiality and integrity) at the transport layer by default, so all payload and most header information is encrypted [14]. QUIC's handshake is streamlined to reduce latency; it performs the cryptographic handshake in parallel with the

transport handshake and can even send data in the first round-trip (0-RTT data) when reconnecting to a server it has talked to before [14]. Another notable feature is support for connection migration: a QUIC connection is identified by a stable connection ID rather than the 4-tuple of IP and port, which means if a device's network changes (e.g. switching from Wi-Fi to LTE), the QUIC connection can continue seamlessly on the new network path without reconnecting [14]. These properties make QUIC particularly suited to modern web use cases and have led major web companies to embrace it, as discussed later.

2.3.4 Hypertext Transfer Protocol Version 3 (HTTP/3)

HTTP/3 is the latest version of the HTTP protocol and operates over QUIC, in contrast to HTTP/2, which runs atop TCP [5]. By leveraging QUIC as its transport, HTTP/3 eliminates head-of-line blocking at the transport layer, since each QUIC stream is delivered independently. This architecture also enables faster connection establishment and mandates encryption for all traffic by default. HTTP/3 preserves the core semantics of HTTP/2—including streams, headers, and methods—but redefines their encoding to suit QUIC's features. Effectively, HTTP/3 replaces the TCP+TLS stack of HTTP/2 with QUIC, providing an application protocol better suited to modern network conditions [5].

A key motivation for HTTP/3 was to further reduce web page load times and improve latency, especially under adverse network conditions. Because QUIC delivers reliability and flow control per stream and provides shared congestion control across all streams, HTTP/3 can avoid the multi-stream head-of-line blocking problem inherent to HTTP/2 [5]. In practical terms, packet loss in one HTTP/3 stream does not impact the progress of others, whereas, in HTTP/2, all streams on a connection are stalled by a single lost TCP packet.

2.3.5 Multiplexed Application Substrate over QUIC Encryption

Multiplexed Application Substrate over QUIC Encryption (MASQUE) is a framework and set of protocols that enable tunneling and proxying various types of traffic within an HTTP/3 (QUIC) connection. In essence, MASQUE allows an HTTP/3 client and server (often, a user and a proxy) to negotiate the ability to exchange arbitrary datagrams or even IP packets, all encapsulated inside a single QUIC connection [26]. This approach can be used to build VPNs or general-purpose proxies that are indistinguishable from regular HTTPS traffic, since to the network everything looks like QUIC-encrypted data between the client and proxy. For example, MASQUE defines mechanisms like HTTP Datagrams and extended CONNECT methods so that a client can request the proxying of UDP traffic to some target, and then send/receive those UDP packets as QUIC DATAGRAM frames inside the secure connection [26]. By multiplexing stream-based HTTP requests and datagram-based proxy traffic together, MASQUE can efficiently use one underlying transport connection for many purposes concurrently.

A key benefit is improved privacy and obfuscation—e.g. someone observing the network cannot easily distinguish if the user is loading a webpage or tunneling IP traffic—as well as improved performance by reducing overhead (one handshake for many tunneled flows).

2.3.6 WebTransport

WebTransport is an emerging protocol framework that enables web applications (JavaScript in browsers) to use a dedicated transport-like channel to servers, with support for multiple streams and datagrams, all accessible via a JavaScript API. In effect, WebTransport gives web developers an alternative to the traditional WebSocket or AJAX mechanisms for sending data: it allows bidirectional, multiplexed, and possibly unreliable communication over HTTP/3 [31]. The WebTransport protocol is implemented on top of HTTP/3 (and QUIC); a client opens a special WebTransport session to a server (over an HTTP/3 connection), after which it can open any number of unidirectional or bidirectional streams and send datagram packets, all within that session [31]. WebTransport is designed to be "safe" under the Web security model (it obeys the same-origin policy, etc.) while giving capabilities similar to a raw UDP/TCP socket, but with the convenience of running through HTTPS/QUIC. This is very useful for applications like real-time gaming, live media feeds, or file sharing directly from the browser, where the request-response pattern of HTTP is too limiting. By standardizing this capability, WebTransport avoids the need for hacks like abusing WebSockets or multiple HTTP requests for real-time data, and it leverages QUIC's benefits (low latency and multiplexing) underneath [31, 33]. As of 2025, WebTransport drafts are in progress at the IETF and implementations are being tested in browsers.

2.4 Evaluation and Discussion

The following sections provide a detailed comparative evaluation of the most significant next-generation transport and application protocols. The performance differences between QUIC (HTTP/3) and TCP (HTTP/2) are first analyzed under various network conditions, followed by an examination of large-scale deployment experiences. Subsequent subsections review empirical evidence on the adoption and effectiveness of MPTCP, MASQUE, and WebTransport in real-world scenarios. Table 2.2 summarizes the principal findings of the evaluation, highlighting where each protocol delivers notable benefits, the scenarios in which these are most evident, and key deployment challenges.

2.4.1 MPTCP Performance and Adoption

Multipath TCP (MPTCP) has been proposed for over a decade to harness multiple network interfaces concurrently (e.g. Wi-Fi + cellular) for greater performance and

Table 2.2: Key Findings from Protocol Evaluation and Adoption.

Protocol	**Main Benefits**	**Where Benefits Are Most Pronounced**	**Key Challenges**
MPTCP	Aggregated bandwidth, seamless failover, session continuity	Multi-interface scenarios (mobile, vehicular, Wi-Fi+cellular)	Middlebox/NAT compatibility, limited real-world deployment
QUIC and HTTP/3	Reduced latency (0/1-RTT), avoids HoL blocking, fast connection migration, improved user experience	High-latency, lossy, mobile or unreliable networks; tail latency reduction	CPU overhead on fast/reliable links; requires UDP support; implementation tuning
MASQUE	Efficient tunneling/proxying, minimal overhead, privacy (traffic obfuscation), scalable	VPN/proxy scenarios, lossy last-mile networks	Proxy configuration, congestion management, early deployment stage
WebTransport	Multiplexed real-time communication in browsers, low latency, no HoL blocking	Browser-based apps, lossy networks, concurrent streams	Early standardization, browser/server support, API maturity

reliability. Recent empirical studies provide insight into both, the performance of MPTCP in real-world scenarios (mobile, vehicular, etc.) and its adoption in the wild.

2.4.1.1 Performance Benefits in Mobile and Vehicular Use

A wealth of research confirms that when MPTCP is enabled, it can improve throughput, latency, and robustness by utilizing multiple paths. Modern smartphones are often multi-homed (e.g. Wi-Fi + cellular). MPTCP allows simultaneous use of both links, which has been shown to increase aggregate bandwidth and provide seamless failover. For example, Apple's use of MPTCP in Siri is aimed at reliability: if Wi-Fi drops during a Siri request, the connection transparently continues over cellular. Korea Telecom has leveraged MPTCP in its network (in partnership with Samsung) to deliver gigabit-class speeds by bonding LTE and Wi-Fi connections [27], something a single interface could not achieve alone. Academic studies similarly demonstrate that MPTCP can aggregate bandwidth across interfaces. One vehicular network experiment showed that using MPTCP with a car's LTE and 802.11p Wi-Fi links yielded higher total throughput than either link alone, and allowed smooth handover when the car moved out of Wi-Fi range [13]. The MPTCP session seamlessly switched to the cellular link when the Wi-Fi dropped, avoiding a connection break. This illustrates MPTCP's potential in vehicular or mobile scenarios to preserve session continuity and quality as network availability changes. Another benefit observed is improved resilience: by sending duplicate or split traffic across paths, MPTCP can tolerate one path's packet loss or high latency by compensating with the other path. Researchers have found MPTCP generally achieves higher aggregated throughput and better resource utilization than single-path TCP in heterogeneous network conditions [27]. These gains are most pronounced when network paths have complementary characteristics (e.g. one may have lower latency, the other higher throughput). Empirical QoE studies (before 2020) on mobile web browsing over MPTCP showed faster page load times in scenarios where one interface on its own was insufficient, though energy consumption is a consideration (using two radios can drain battery faster, which is an active research topic for MPTCP scheduling). Overall, the real-world evidence suggests that in multi-interface environments—from smartphones to connected cars—MPTCP can significantly boost performance and reliability by pooling network resources.

2.4.1.2 Adoption and Real-World usage

Although standardized in 2013, MPTCP's deployment was historically limited. New measurements post-2020 indicate a cautiously growing uptake. A longitudinal study scanned the Internet and found a steady rise in MPTCP-capable hosts: by late 2021, over 13000 IPv4 addresses responded to MPTCP handshakes (approximately doubling from the year prior) and around 1000 IPv6 addresses (a 40× increase, though from a small base) [27]. This growth correlates with Apple's

adoption—Apple iOS devices use MPTCP for Siri, Apple Music, Maps, and Wi-Fi Assist, and in early 2022 Apple enabled MPTCPv1 on iOS, rapidly increasing the number of MPTCPv1-capable servers (hundreds of Apple servers upgraded to v1) [27]. Indeed, the scans showed Apple dominates MPTCPv1 deployment in the current Internet [27]. Outside of Apple's ecosystem, support is still sparse but rising: the same study found ∼13k hosts with MPTCPv0 (mainly Linux-based servers or proxies) and only ∼100 hosts with MPTCPv1 as of 2022 [27]. In terms of traffic, MPTCP remains a tiny fraction—analysis of backbone traces showed MPTCP traffic volume increased 20× from 2014 to 2021, but still under 0.5% of bytes (only 0.044% of TCP traffic in one 2021 trace) [1]. In other words, MPTCP is growing in use but has not yet achieved mass adoption; it's mostly employed in niche or controlled environments (e.g. Apple's services, research testbeds, or specific carrier offerings). A complicating factor for wider adoption is middleboxes: studies uncovered that many hosts appear MPTCP-aware due to middleboxes mirroring TCP options (false positives), and some middleboxes interfere with MPTCP's subflow establishment [27, 1]. This suggests deployment challenges—network devices not recognizing MPTCP can block or disrupt it—which partly explains the slow rollout despite MPTCP's benefits.

2.4.1.3 Current Status and Challenges

With MPTCP now part of Linux (since kernel 5.6 in 2020) [29] and available via APIs on iOS, its adoption may accelerate. However, the fragmented uptake (dominance in Apple's closed ecosystem but not widely used elsewhere) shows that incentives and compatibility issues remain. Some carriers and vendors hesitate to deploy MPTCP because of middlebox compatibility and the complexity of managing multipath connections. Additionally, the rise of QUIC has introduced an alternative path to multipath capabilities (the IETF is exploring Multipath QUIC as a future extension, which could potentially sidestep some TCP-level issues). For now, MPTCP's real-world usage is concentrated in specific domains that need multi-homing support. The empirical data in the previous section can guide the discussion: MPTCP clearly works and improves performance in the scenarios where it's enabled, but its overall impact on Internet traffic is still limited by deployment barriers.

2.4.2 QUIC Performance

Recent empirical studies offer a nuanced view of QUIC (HTTP/3) performance compared to TCP (HTTP/2), showing significant gains in some scenarios but minimal or even negative impact in others.

2.4.2.1 Lossy or High-Latency Networks

QUIC often outperforms TCP under adverse conditions. One systematic evaluation found HTTP/3 can greatly outperform HTTP/2 in noisy links—achieving up to ∼88% faster transfers under high packet loss and latency, thanks to QUIC's UDP multiplexing and connection migration features [20]. In extreme loss scenarios, QUIC maintained robust throughput (improving performance by ∼81% in one test) whereas TCP throughput collapsed [20]. Similarly, an empirical study of Dynamic Adaptive Streaming over HTTP (DASH) showed QUIC/HTTP3 delivering better Quality of Experience (QoE) than TCP/HTTP2. With certain low-latency adaptive streaming algorithms, HTTP/3 achieved higher throughput and faster media segment downloads, outperforming HTTP/2 especially on lossy networks [7]. These results indicate QUIC's design (eliminating head-of-line blocking and fast 1-RTT handshakes) yields lower latency and better resilience on unreliable paths. Real-world data aligns with this: Facebook reported a 20% reduction in tail latency for requests after enabling QUIC [15], and Cloudflare observed ∼116 ms faster TLS handshake times at the 95th percentile with HTTP/3 enabled (a ∼17% improvement in high-latency cases) [30].

2.4.2.2 Well-Provisioned or High-Speed Networks

In contrast, the advantages of QUIC can fade on very fast, reliable links—and may even reverse. A 2024 study found that over high-bandwidth, low-delay Internet paths, the QUIC+HTTP/3 stack suffered up to 45% lower throughput than TCP+HTTP/2 [36]. Moreover, this performance gap widened as bandwidth increased [36]. The degradation was observed across multiple browsers (Chrome, Firefox, etc.) and affected applications like video streaming (reducing video bitrate by ∼9–10%) [36]. Root-cause analysis traced the slowdown to QUIC's implementation overhead—in particular, high CPU cost of user-space packet processing and frequent ACK handling [36]. In essence, on gigabit networks with minimal loss, current QUIC stacks can become CPU-bound, negating throughput benefits. These findings temper the "QUIC is always faster" narrative, showing that TCP can outperform QUIC in ideal network conditions (where TCP's inefficiencies like head-of-line blocking are irrelevant and QUIC's overhead dominates).

2.4.2.3 Impact of Implementation and Congestion Control

Multiple studies stress that QUIC's real-world performance depends heavily on how it's deployed and tuned. A large-scale 2021 comparison of production QUIC vs. TCP (testing Google, Facebook, and Cloudflare's live services) concluded that congestion control choices often outweigh protocol differences. For instance, Google's BBR congestion control versus Reno could swing results more than QUIC vs TCP itself [35]. That study confirmed QUIC's 1-RTT handshake gives a modest win for small payloads (e.g. slight faster completion for sub-1MB objects), but for larger transfers the handshake advantage was negligible [35]. Notably, they found QUIC's

widely praised elimination of head-of-line blocking had little observable impact on page load times in practice [35] —likely because modern HTTP/2 uses already optimized multiplexing and most web flows aren't heavily stalled by a single lost packet. In scenarios with 1% packet loss, differences came down to congestion algorithms: for example, Cloudflare's QUIC initially underperformed HTTP/2 until tuning issues were identified [35]. The key implication is that many performance "wins" or regressions attributed to QUIC are in fact due to implementation quirks, default parameters, and congestion control algorithm behavior [35]. In summary, QUIC can be faster than TCP, but achieving that consistently requires careful engineering (e.g. CPU optimizations, loss recovery tuning) to avoid undermining QUIC's theoretical advantages.

2.4.3 QUIC Adoption

Major technology companies and CDNs have widely deployed QUIC (HTTP/3) in production, yielding valuable data on its real-world benefits.

2.4.3.1 Facebook/Meta's Adoption

Facebook began migrating its traffic to IETF QUIC/HTTP3 around 2020 and by late 2020 over 75% of Facebook's Internet traffic was using QUIC [15]. This large-scale rollout led to concrete performance gains. In the Facebook app, enabling QUIC for API calls reduced request error rates by ∼6% and cut tail request latency by ∼20% compared to HTTP/2 [15]. Response header sizes also dropped ∼5% due to HPACK vs QPACK differences [15]. These improvements translated into a smoother user experience. For example, Facebook's video streaming saw "transformative" improvements after QUIC deployment: the mean time between rebuffering events increased by up to 22%, video stall frequency dropped 20%, and video error rates fell ∼8% in aggregate [15]. Notably, QUIC had an outsized impact for users on slower or more error-prone networks (e.g. mobile networks in emerging markets), consistent with QUIC's strength in unreliable conditions [15]. Facebook's engineers did encounter some regressions (e.g. app heuristics tuned for TCP had to be readjusted for QUIC's faster speed) [15], and they invested in optimizing their QUIC implementation (mvfst) for CPU efficiency before enabling QUIC for all content [15]. By 2021, Facebook had extended QUIC to Instagram and their web frontends, reporting similar or better performance gains on those platforms [15]. In summary, Facebook's real-world deployment confirms several QUIC benefits (lower latency, fewer errors, better video QoE), especially under conditions where TCP previously struggled.

2.4.3.2 Cloudflare and Web-Wide Adoption

Cloudflare, as a major CDN, enabled HTTP/3 across its edge network early and has tracked adoption closely. By late 2023, about 20% of global web requests seen

by Cloudflare used HTTP/3 [4], while roughly 47% were HTTP/2 and the rest HTTP/1.x. Some countries have even higher HTTP/3 usage—up to ∼30–35% of traffic—although no country exceeded 35% HTTP/3 as of 2023 [4]. This indicates substantial uptake of QUIC in just a few years since standardization. Cloudflare's measurements also highlight performance benefits of HTTP/3 for TLS connection setup and handshake latency. In a controlled A/B test, identical content served with HTTP/3 had significantly faster connection times than HTTP/2: at the 95th percentile, enabling HTTP/3 reduced connect latency from 695 ms to 579 ms (a ∼17% improvement) [30]. Median connection time also improved (130 ms vs 174 ms) [30]. Cloudflare attributes this to QUIC's zero-RTT handshake and loss-tolerant setup, which allows clients to start sending data sooner [30]. The faster setup, combined with QUIC's ability to avoid head-of-line blocking, helps improve user-perceived page load times in real networks with nontrivial round-trip delays [30]. Beyond performance, Cloudflare notes increased confidence in HTTP/3: 2022 was a "milestone year" for HTTP/3 deployment, and by 2024 all major browsers have it enabled by default [22, 30]. In practice, adopting HTTP/3 is often as simple as toggling it on (Cloudflare offers it free to all customers), but the impact can be significant for end-user performance on suboptimal networks.

2.4.3.3 Other Adoptions

Google was the pioneer of QUIC, deploying a precursor (gQUIC) for Google Search, YouTube, and other services in the mid-2010s. Google's early experiments reportedly saw modest page load time improvements (on the order of a few percent mean page load time reduction) for high-traffic sites by using QUIC [16], especially on mobile and high-latency connections. Since 2020, Google has moved to IETF QUIC and enabled HTTP/3 in Chrome (as of version 87+). Although Google hasn't published new performance figures recently, QUIC remains fundamental to their protocols (e.g. HTTP/3 and HTTP/3-based VPN/proxy protocols). Akamai, another large CDN, was an early adopter as well—they deployed Google's QUIC in 2016 and have since offered IETF QUIC support to customers [36]. Overall, the large-scale deployments by Google, Facebook, Cloudflare, and others demonstrate that QUIC/HTTP3 is not only widely supported but also generally beneficial. The degree of benefit, however, varies by context: sites with lots of small objects or users on high-latency mobile links see the most gain, whereas sites on very fast networks might see more modest differences or need to optimize QUIC's resource usage.

2.4.4 MASQUE Performance and Adoption

Empirical evaluations have shown that MASQUE introduces minimal bandwidth overhead—typically in the range of 3–5% when using QUIC datagrams as the encapsulation method [18]. For instance, it was measured that encapsulating UDP

packets via HTTP/3 DATAGRAM frames results in a mean overhead of approximately 3%, with this overhead decreasing as packet sizes increase [18]. When reliable delivery is required (using HTTP/3 streams), the overhead can be higher due to retransmissions and additional framing; however, even in these cases, MASQUE can significantly reduce end-to-end latency under lossy last-mile conditions by recovering lost packets locally at the proxy [18].

Notably, MASQUE acts as a performance-enhancing proxy in lossy network scenarios. In controlled experiments with simulated last-hop packet loss rates of 2–5%, reliable stream-based MASQUE tunnels reduced file transfer completion times by up to 80% compared to direct QUIC connections, since local retransmissions over the short client-proxy link quickly recovered losses before they affected the end-to-end flow [18]. However, this can mask congestion signals from the origin server, necessitating careful proxy configuration to avoid bufferbloat. Real-world deployment evidence confirms MASQUE's scalability and reliability: Cloudflare reports successful global operation of MASQUE in production as the core of their WARP VPN service, now handling millions of connections per day. The protocol is designed for high concurrency, with single MASQUE tunnels efficiently multiplexing hundreds of flows, and modern implementations (e.g., in Rust or C++) achieving line-rate throughput on commodity hardware [11, 12].

Additional studies in cellular network emulation environments have validated that MASQUE-based QUIC proxies can maintain throughput and reliability across LTE scenarios [25]. However, performance bottlenecks may arise in high-throughput scenarios due to the computational limitations of real-time simulators such as ns-3, which can introduce additional delay beyond protocol or network constraints [25]. In these environments, while the per-packet header overhead remains modest and protocol-level tuning (including congestion control and scheduling) is important, careful attention must also be paid to simulator resource constraints to avoid artificial bottlenecks [25].

2.4.5 WebTransport Performance

Empirical analysis shows that, under current browser implementations, WebSockets outperform WebTransport in both connection setup and message response times. In experiments using Chrome, establishing 50 new WebSocket or WebTransport connections typically took around 5 seconds, with WebTransport slightly slower. Message response times increased linearly with the number of connections, but WebTransport's latency was consistently higher than WebSockets in the browser, and both were slower than a Node.js WebSocket client [33].

When adding artificial network latency (100 ms) or packet loss (7%), both protocols experienced similar increases in connection and response times, with no observed advantage for WebTransport. Notably, WebTransport failed to complete tests at higher packet loss rates, contrary to expectations that QUIC would outperform TCP in degraded conditions [33].

Tests of WebTransport's delivery mechanisms showed that single-stream and datagram modes reduced connection setup times compared to multi-stream, with datagram delivery maintaining low latency up to about 150 simultaneous messages, beyond which message loss occurred. Chrome allowed up to 6,000 concurrent WebTransport connections, compared to 250 for WebSockets, but WebTransport setup required a larger payload and more packets. Once established, message payloads were similar across both protocols [33].

WebTransport is still in a draft stage, with relatively few mature implementations. Its theoretical advantages—such as support for multi-streaming and datagram-based delivery—have yet to translate into clear empirical performance gains in typical browser-based scenarios. As implementations mature, further research is needed to clarify the circumstances under which WebTransport can reliably outperform WebSockets, particularly in adverse network conditions or at scale [33]

2.4.6 Summary

Table 2.3 highlights the most significant quantitative results from the evaluation of each protocol, summarizing where these technologies offer the greatest performance benefits (or, in the case of QUIC and HTTP/3, important trade-offs) under real-world conditions.

2.5 Summary and Conclusion

The following sections provide a summary of the key findings regarding the evaluation and real-world impact of modern transport and application protocols, including QUIC, HTTP/3, MPTCP, MASQUE, and WebTransport. The subsequent discussion addresses the principal challenges and deployment barriers facing these protocols, followed by an outlook on promising directions for future research and development.

2.5.1 Key Findings

This paper has surveyed the evolution, design, and real-world performance of new transport protocols and application protocols for the modern web, focusing on QUIC, MPTCP, HTTP/3, MASQUE, and WebTransport. The evidence shows that next-generation transport protocols deliver concrete benefits over legacy approaches, particularly in environments characterized by high latency, frequent loss, or multipath connectivity. QUIC and HTTP/3 reduce connection establishment delays through 1-RTT or 0-RTT handshakes and eliminate head-of-line blocking at the transport layer, translating into improved web performance on lossy or mobile networks [20, 15, 30]. Large-scale deployments by leading technology companies confirm these benefits, with notable improvements in user-perceived

Table 2.3: Summary of Most Significant Quantitative Results for Modern Transport Protocols and Application Protocols.

Protocol	**Key Numeric Result**	**Summary Context**
QUIC and HTTP/3 (High-Latency Networks)	Up to 88% faster transfers (20% lower tail latency) [20]	Under high loss or latency, QUIC/HTTP3 transfers are up to 88% faster than HTTP/2/TCP; production use at Facebook cut tail latency by 20%.
QUIC and HTTP/3 (High-Speed Networks)	Up to 45% lower throughput (negative result) [36]	On high-speed, reliable links, QUIC/HTTP3 can be up to 45% slower than TCP/HTTP2 due to CPU overhead (e.g., for large video streaming).
MPTCP	20× increase in global use since 2014 [1]	MPTCP traffic volume has increased 20-fold from 2014–2021, though it remains under 0.5% of total TCP bytes; enables seamless failover in Apple/Samsung deployments.
MASQUE	Up to 80% faster file transfers [18]	In lossy last-mile networks, MASQUE tunnels can reduce file transfer times by up to 80% compared to direct QUIC connections, with only 3–5% bandwidth overhead.
WebTransport	5,000–14,000 ms connection setup (browser); fails at high loss [33]	WebTransport setup takes 5–14 s per 50 connections (slower than WebSockets); under 7%+ packet loss, performance is comparable or worse, and tests may fail to complete. No measured latency advantage over WebSockets under degraded conditions.

latency, error rates, and video streaming quality [15, 4]. MPTCP, meanwhile, enables seamless aggregation of bandwidth and robust failover in multi-interface scenarios, as demonstrated in both research testbeds and production systems like Apple's ecosystem [27, 13]. The emergence of MASQUE and WebTransport further extends the reach of QUIC-based architectures, enabling efficient tunneling, obfuscation, and real-time communication natively in browsers with minimal overhead for MASQUE and some overhead for WebTransport [11, 33].

Importantly, empirical studies reveal that the real-world gains of these protocols are context-dependent. While HTTP/3 over QUIC outperforms HTTP/2 over TCP under adverse conditions such as high latency or lossy networks, its benefits are less pronounced—and can even reverse—on fast, reliable links due to implementation overhead [36, 35]. Similarly, MPTCP's impact is greatest where multiple heterogeneous network paths are available, but its adoption remains limited by middlebox interference and slow deployment in existing networks. Performance outcomes are also strongly influenced by implementation choices, congestion control algorithms, and careful tuning [27].

2.5.2 Challenges and Ongoing Work

Despite the substantial progress, several challenges persist. Deployment barriers—such as middlebox compatibility for MPTCP, varying UDP support for QUIC, and the complexities of protocol negotiation—continue to slow universal adoption [1, 16]. Implementation overhead, especially CPU utilization in high-speed environments, remains a bottleneck for QUIC-based stacks [36]. Moreover, real-world performance still depends heavily on the maturity of protocol libraries, tuning of congestion control, and operating system support.

For MPTCP, middlebox traversal and the lack of consistent support outside of tightly controlled environments (notably Apple's platforms) constrain its broader use. It appears that as of 2025, MPTCP is only natively available in the Linux and macOS kernels, but not in the Windows kernel. QUIC, although widely adopted in browsers and CDNs, faces ongoing scrutiny regarding optimal resource management, fairness in shared networks, and security against novel attack vectors. The deployment of MASQUE and WebTransport is in early stages, and widespread interoperability and monitoring tooling are still emerging.

Research continues on extending these protocols. Multipath QUIC, for example, seeks to combine QUIC's flexibility with MPTCP's multi-homing [32], while the evolution of MASQUE and WebTransport will be shaped by browser support and emerging real-time use cases. The trade-offs between performance, security, and deployability remain active research areas.

2.5.3 Future Outlook

The transport layer of the Internet is undergoing a significant transformation. QUIC and HTTP/3 are now default in major browsers and CDNs, marking a generational shift away from TCP's legacy limitations. MASQUE and WebTransport point toward a future in which secure, multiplexed, and highly adaptive transport is a built-in feature of the web platform itself, not an add-on. As protocol implementations mature and operating system, browser, and network device support continues to improve, the proportion of Internet traffic carried by these next-generation protocols will likely increase rapidly.

Key research directions include further optimizing protocol stacks for performance and resource efficiency, addressing deployment and compatibility issues (particularly for multipath technologies), and continuing to monitor security implications as these protocols become ubiquitous. The anticipated standardization of Multipath QUIC, broader adoption of MASQUE proxies, and the evolution of WebTransport APIs will further shape the landscape. Ultimately, the adoption of these protocols represents an ongoing, community-driven process, balancing innovation with the practical realities of global Internet deployment.

In summary, the modern web is already reaping the benefits of new transport protocols, but continued research, careful engineering, and cross-ecosystem collaboration are essential to fully realize their potential for secure, reliable, and performant Internet communication.

References

[1] Florian Aschenbrenner et al. "From Single Lane to Highways: Analyzing the Adoption of Multipath TCP in the Internet". In: *2021 IFIP Networking Conference (IFIP Networking)*. June 2021, pp. 1–9. DOI: 10.23919/IFIPNetworking52078.2021.9472785.

[2] Mike Belshe and Roberto Peon. *SPDY Protocol*. Internet-Draft. Work in Progress. Mar. 2012.

[3] Mike Belshe, Roberto Peon, and Martin Thomson. *Hypertext Transfer Protocol Version 2 (HTTP/2)*. RFC 7540. May 2015. DOI: 10.17487/RFC7540. URL: https://www.rfc-editor.org/info/rfc7540.

[4] David Belson. *Cloudflare 2023 Year in Review*. Accessed: 2025-05-23. 2023. URL: %5Curl%7Bhttps://blog.cloudflare.com/radar-2023-year-in-review/%7D.

[5] Mike Bishop. *HTTP/3*. RFC 9114. June 2022. DOI: 10.17487/RFC9114. URL: https://www.rfc-editor.org/info/rfc9114.

[6] Luomeng Chao et al. "A Brief Review of Multipath TCP for Vehicular Networks". In: *Sensors* 21.2793 (2021). DOI: 10.3390/s21082793.

[7] Sindhu Chellappa and Radim Bartos. "Is QUIC Quicker with HTTP/3? An Empirical Analysis of Quality of Experience with DASH Video Streaming". In: *2022 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. Dec. 2022, pp. 237–242. DOI: 10.1109/ANTS56424.2022.10227765.

[8] Wesley Eddy. *Transmission Control Protocol (TCP)*. RFC 9293. Aug. 2022. DOI: 10.17487/RFC9293. URL: https://www.rfc-editor.org/info/rfc9293.

[9] Roy T. Fielding and Julian Reschke. *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing*. RFC 7230. June 2014. DOI: 10.17487/RFC7230. URL: https://www.rfc-editor.org/info/rfc7230.

[10] Alan Ford et al. *TCP Extensions for Multipath Operation with Multiple Addresses*. RFC 6824. Jan. 2013. DOI: 10.17487/RFC6824. URL: https://www.rfc-editor.org/info/rfc6824.

[11] Mari Galicer. *Donning a MASQUE: building a new protocol into Cloudflare WARP*. Accessed: 2025-05-23. 2023. URL: %5Curl%7Bhttps://blog.cloudflare.com/masque-building-a-new-protocol-into-cloudflare-warp/%7D.

[12] Dan Hall. *Zero Trust WARP: tunneling with a MASQUE*. Accessed: 2025-05-23. 2024. URL: %5Curl%7Bhttps://blog.cloudflare.com/zero-trust-warp-with-a-masque/%7D.

[13] Vadym Hapanchak and Antonio Costa. "A Cross-layer Approach for MPTCP Path Management in Heterogeneous Vehicular Networks". In: *Journal of Communications Software and Systems* 19.1 (Mar. 2023), pp. 103–113. DOI: 10.24138/jcomss-2022-0177. URL: %5Curl%7Bhttps://doi.org/10.24138/jcomss-2022-0177%7D.

[14] Jana Iyengar and Martin Thomson. *QUIC: A UDP-Based Multiplexed and Secure Transport*. RFC 9000. May 2021. DOI: 10.17487/RFC9000. URL: https://www.rfc-editor.org/info/rfc9000.

[15] Matt Joras and Yang Chi. *How Facebook is bringing QUIC to billions*. Accessed: 2025-05-23. 2020. URL: %5Curl%7Bhttps://engineering.fb.com/2020/10/21/networking-traffic/how-facebook-is-bringing-quic-to-billions%7D.

[16] Arash Molavi Kakhki et al. "Taking a long look at QUIC: an approach for rigorous evaluation of rapidly evolving transport protocols". In: *Commun. ACM* 62.7 (June 2019), pp. 86–94. ISSN: 0001-0782. DOI: 10.1145/3330336. URL: %5Curl%7Bhttps://doi.org/10.1145/3330336%7D.

[17] Duckwoo Kim et al. "FlexCP: A Scalable Multipath TCP Proxy for Cellular Networks". In: *Proc. ACM Netw.* 1.CoNEXT3 (Nov. 2023). DOI: 10.1145/3629143. URL: https://doi.org/10.1145/3629143.

[18] Mirja Kühlewind et al. "Evaluation of QUIC-based MASQUE proxying". In: *Proceedings of the 2021 Workshop on Evolution, Performance and Interoperability of QUIC*. EPIQ '21. Virtual Event, Germany: Association for Computing Machinery, 2021, pp. 29–34. DOI: 10.1145/3488660.3493806. URL: %5Curl%7Bhttps://doi.org/10.1145/3488660.3493806%7D.

[19] Adam Langley et al. "The QUIC Transport Protocol: Design and Internet-Scale Deployment". In: *Proceedings of the Conference of the ACM Special*

Interest Group on Data Communication. SIGCOMM 17. Los Angeles, CA, USA: Association for Computing Machinery, 2017, pp. 183–196. DOI: 10.1145/3098822.3098842. URL: https://doi.org/10.1145/3098822.3098842.

[20] Fan Liu et al. "Performance Comparison of HTTP/3 and HTTP/2: Proxy vs. Non-Proxy Environments". In: *arXiv preprint arXiv:2409.16267* (2024).

[21] Peter Megyesi, Zsolt Kraemer, and Sandor Molnar. "How quick is QUIC?" In: *2016 IEEE International Conference on Communications (ICC)*. 2016, pp. 1–6. DOI: https://doi.org/10.1109/ICC.2016.7510788.

[22] Mark Nottingham. *The State of HTTP in 2022*. Accessed: 2025-05-23. 2022. URL: %5Curl%7Bhttps://blog.cloudflare.com/the-state-of-http-in-2022/%7D.

[23] J. Postel. *User Datagram Protocol*. RFC 768. Aug. 1980. DOI: 10.17487/RFC0768. URL: https://www.rfc-editor.org/info/rfc768.

[24] Eric Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.3*. RFC 8446. Aug. 2018. DOI: 10.17487/RFC8446. URL: https://www.rfc-editor.org/info/rfc8446.

[25] Donat Scharnitzky et al. "Real-time Emulation of MASQUE-based QUIC Proxying in LTE Networks using ns-3". In: *Proceedings of the 2023 2nd Asia Conference on Algorithms, Computing and Machine Learning*. CACML '23. Shanghai, China: Association for Computing Machinery, 2023, pp. 581–586. DOI: 10.1145/3590003.3590995. URL: %5Curl%7Bhttps://doi.org/10.1145/3590003.3590995%7D.

[26] David Schinazi. *The MASQUE Proxy*. Internet-Draft draft-schinazi-masque-proxy-05. Work in Progress. Internet Engineering Task Force, Feb. 2025. 7 pp. URL: https://datatracker.ietf.org/doc/draft-schinazi-masque-proxy/05/.

[27] Tanya Shreedhar et al. *A Longitudinal View at the Adoption of Multipath TCP*. 2022. arXiv: 2205.12138 [cs.NI]. URL: %5Curl%7Bhttps://arxiv.org/abs/2205.12138%7D.

[28] Randall R. Stewart, Michael Tüxen, and Karen Nielsen. *Stream Control Transmission Protocol*. RFC 9260. June 2022. DOI: 10.17487/RFC9260.

[29] The kernel development community. *Multipath TCP (MPTCP)*. Accessed: 2025-05-23. URL: %5Curl%7Bhttps://docs.kernel.org/networking/mptcp.html%7D.

[30] David Tuber. *Network Performance Update: Security Week 2024*. Accessed: 2025-05-23. 2024. URL: %5Curl%7Bhttps://blog.cloudflare.com/network-performance-update-security-week-2024%7D.

[31] Victor Vasiliev. *The WebTransport Protocol Framework*. Internet-Draft draft-ietf-webtrans-overview-09. Work in Progress. Internet Engineering Task Force, Feb. 2025. 13 pp. URL: https://datatracker.ietf.org/doc/draft-ietf-webtrans-overview/09/.

[32] Tobias Viernickel et al. "Multipath QUIC: A Deployable Multipath Transport Protocol". In: *2018 IEEE International Conference on Communications (ICC)*. May 2018, pp. 1–7. DOI: 10.1109/ICC.2018.8422951.

[33] Declan Williamson and Ruairi O'Reilly. "WebTransport and WebSockets: An Empirical Analysis of Connection Time, Message Response, and Payload

Efficiency". In: *2023 34th Irish Signals and Systems Conference (ISSC)*. June 2023, pp. 1–6. DOI: 10.1109/ISSC59246.2023.10162060.

[34] Konrad Wolsing et al. "A performance perspective on web optimized protocol stacks: TCP+TLS+HTTP/2 vs. QUIC". In: *Proceedings of the 2019 Applied Networking Research Workshop*. ANRW '19. Montreal, Quebec, Canada: Association for Computing Machinery, 2019, pp. 1–7. DOI: 10.1145/3340301.3341123. URL: https://doi.org/10.1145/3340301.3341123.

[35] Alexander Yu and Theophilus A. Benson. "Dissecting Performance of Production QUIC". In: *Proceedings of the Web Conference 2021*. WWW '21. Ljubljana, Slovenia: Association for Computing Machinery, 2021, pp. 1157–1168. DOI: 10.1145/3442381.3450103. URL: %5Curl%7Bhttps://doi-org.ezproxy.uzh.ch/10.1145/3442381.3450103%7D.

[36] Xumiao Zhang et al. "QUIC is not Quick Enough over Fast Internet". In: *Proceedings of the ACM Web Conference 2024*. WWW '24. Singapore, Singapore: Association for Computing Machinery, 2024, pp. 2713–2722. DOI: 10.1145/3589334.3645323. URL: %5Curl%7Bhttps://doi.org/10.1145/3589334.3645323%7D.

Chapter 3

Blockchains beyond Cryptocurrency – Practical Use Cases

Efe Saman, Cagla Ataoglu

Blockchain is a transaction and data management technology that prioritizes decentralization and immutability. It was initially developed as a distributed ledger for cryptocurrency transactions but nowadays, its application has spread into many other fields and industries. This report looks into some use cases of blockchain beyond cryptocurrency and evaluates the benefits and drawbacks of using blockchain in such scenarios. Namely, it investigates supply chain management, healthcare, identity verification, voting systems, intellectual property protection, and banking and finance as areas where blockchain has practical uses. The report considers how blockchain technology can be applied to the field in general and examines specific existing applications.

Contents

3.1 Introduction 49
3.1.1 What is Blockchain 49
3.1.2 Key Features 50
3.1.3 Benefits 51
3.1.4 Challenges 51
3.2 Applications 51
3.2.1 Supply Chain Management 51
3.2.2 Healthcare 53
3.2.3 Identity Verification 57
3.2.4 Voting Systems 59
3.2.5 Intellectual Property Protection 62
3.2.6 Banking and Finance 63
3.3 Conclusion 66

3.1 Introduction

3.1.1 What is Blockchain

Blockchain is a distributed ledger technology, which means a collection that is distributed between many machines [52]. The ledger consists of transactions of tokens on blockchains and are recorded in blocks that are chained together, which makes the blockchain name. Each block has a unique hash value that gets generated from the transactions on the block. Every block references the previous block on the chain via its hash value [2, 64]. Blockchain gained significant traction with its application in cryptocurrencies, the most important cryptocurrency being Bitcoin. Blockchain was first proposed with Bitcoin, as a peer-to-peer electronic currency technology [40]. The peer-to-peer, immutable, secure, and transparent nature of blockchains is a great tool for cryptocurrencies. Through the peer-to-peer network, the blockchain operates without the need for intermediary services, such as banks [8].

In blockchains, the transactions initiated by users are grouped into blocks. These blocks are validated by network participants, whom are referred to as nodes, through consensus mechanisms such as Proof of Work (PoW) or Proof of Stake (PoS). This validation (consensus) allows for the distributed network to always have an agreement on the blocks. When a block is added to the blockchain, it gets the hash of the previous block to reference it, and gets appended to the end of the chain [32]. Figures 3.1 and 3.2 demonstrate an overview of the consensus mechanism and the chain of blocks.

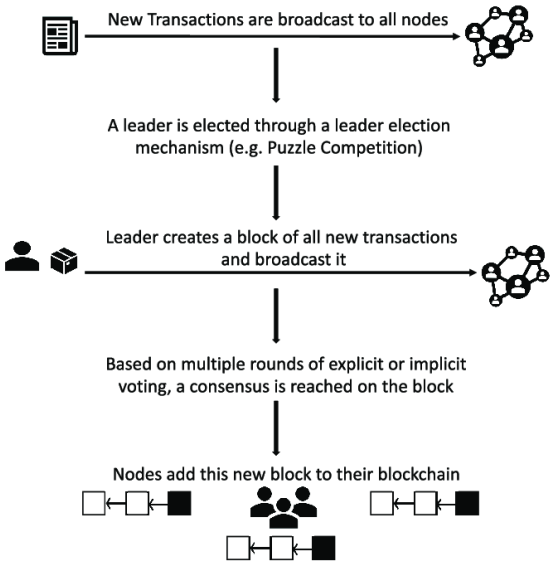


Figure 3.1: Blockchain consensus scenario highlighting the steps taken [45].

The real world applications of blockchains are not limited to cryptocurrencies. This report will cover several practical use cases of blockchain technology beyond cryptocurrencies, focusing on supply chain management, healthcare, finance, the Internet of Things (IoT), and identity management.

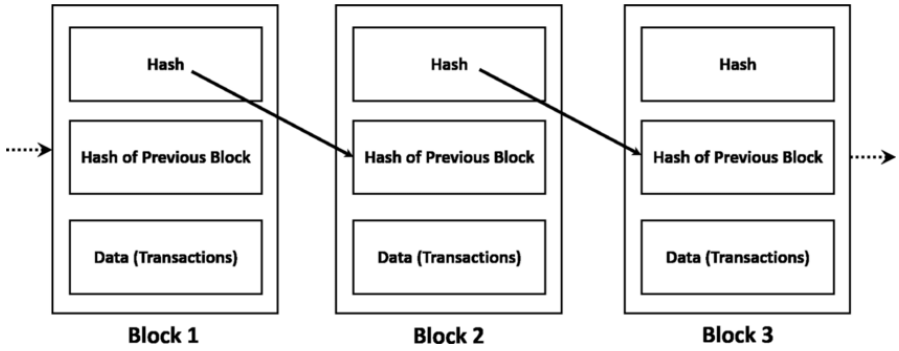


Figure 3.2: The structure of chain of blocks with header and data [45].

3.1.2 Key Features

Some key features of blockchain that showcase its effectiveness and usability are immutability, decentralization, consensus mechanisms, distributed ledger, smart contracts, and cryptography.

Once data is recorded on a blockchain, it cannot be altered or deleted, which makes the blockchain immutable. Each block has a cryptographic hash that is generated through the data it contains, and every block references the previous block through these hashes. This chain structure makes altering the data of any block almost impossible, as it would invalidate the hash of a block, hence the consequent blocks in the chain [64, 61]. In addition, the nodes on the blockchain communicate on a peer-to-peer network. The blocks in the blockchain are maintained through this decentralized network. This eliminates the single point of failure [61, 13] and gives the blockchain its decentralized structure. Transactions are validated through consensus mechanisms such as Proof of Work (PoW) or Proof of Stake (PoS). The consensus mechanisms ensure that all nodes agree on the current state of the blocks (the data) in the blockchain through these mechanisms. This maintains consistency and prevents fraud [61].

The blockchain acts as a distributed ledger. The data recorded in the blocks are recorded in every node in the blockchain network. The blockchain is replicated across all nodes, which enhances resilience, and provides transparency. If any node fails or become compromised, other nodes can still access the data [13]. Smart contracts in blockchains are programmable contracts (code blocks), which are executed when certain criteria are met on the network. These criteria are based on predefined rules, and the blockchain handles the execution of the smart contracts without the need for intermediary services. Ethereum pioneered the integration of smart contracts [65]. It is important to use proper cryptography algorithms to ensure data security on the blockchain. Even though the blocks are distributed in the decentralized network, it is important that ownership verification and transaction approvals can be made. Users possess a pair of cryptographic keys: a public key, which represents their identity on the network, and a private key, which authorizes transactions, which provide security [40].

3.1.3 Benefits

The blockchain is inherently transparent and allows for traceability of transactions. The blockchain nodes have the same distributed ledger, so all participants have access to the same data, providing transparency. Additionally, the clear history of transactions and ownership provide traceability [13]. The cryptographic algorithms used to encrypt data on the blockchain and the immutability (the fact that no data can be altered) provides proof of ownership,and enhances trust [61]. Moreover; cryptographic algorithms, consensus protocols, and immutability provide strong data integrity and security. It is very difficult for attackers to alter historical records [61]. Blockchains can also improve the efficiency of some workflows with its smart contracts'. They speed up automatable processes by eliminating intermediaries, reduces manual intervention, and lowers operational costs [65]. Direct peer-to-peer interactions reduce the reliance on traditional intermediaries. Hence, cutting costs and improving efficiency [13].

3.1.4 Challenges

Despite the benefits, blockchain faces challenges. Migration to blockchain systems often requires fundamental changes to existing systems, which introduces many technical challenges [13]. Interfacing blockchain networks with legacy IT infrastructure is technically challenging and often requires the development of new standards and protocols [65]. In some jurisdictions, the regulations for the blockchain system are not clearly defined. This prohibits the development of systems utilizing blockchain [13]. Moreover, some blockchain platforms perform very poorly when processing large volumes of data, mainly due to their consensus mechanisms, limiting their use in large-scale applications [61]. Consensus protocols such as Proof of Work are highly energy-intensive, contributing to significant environmental impacts [61].

3.2 Applications

3.2.1 Supply Chain Management

Supply chain management (SCM) is a field that comprises of all planning and management of supply and demand within and across companies and also includes logistics management [16]. The transparency and traceability properties of blockchain play a key role in product tracking and inventory management, as well as counterfeit identification and fraud prevention [44]. Having a tamper-proof record of all transactions provides actors from all across the supply chain with reliable, secure, and auditable information.

In a supply chain, a chain of custody refers to "the time-ordered registry of parties who take physical custody of a product as it moves through the supply chain

network". Keeping a secure record of the movement of objects in a supply chain is essential in use cases such as identifying the origin of a product, enforcing certifications and improving transparency [6]. Blockchain technology offers a method to achieve a transparent and traceable chain of custody by representing physical goods as tokens and constructing smart contracts to automate points of transaction. In this way, the usage of blockchain in a SCM also establishes greater end customer satisfaction because suppliers, manufacturers, distributors and many other involved parties can be verified, and the customer is provided with more trustworthy guarantees of the goods' origins.

Thanks to its decentralized architecture, blockchain is inherently collaborative – this fits in well with supply chain management as there is a need to eliminate the process of determining a controlling party or a mediator, which requires a lot of trust. This trust needs to be established by all parties, such as the supplier, manufacturer, and buyer [59]. This centralized approach is problematic because if the trusted party is compromised, the entire supply chain could be put in jeopardy. Moreover, this allows for all parties to blame one another in case of dispute. In blockchain systems, consensus should be fulfilled by all nodes in the system, which mitigates the problem of establishing trusting parties and resolving conflicts in uncertain scenarios.

3.2.1.1 IBM Food Trust

Some industries require more sensitive and careful supply chain management, and the food industry is one of them. Both consumers and producers are expressing greater concern in wellness, health, and sustainability in the food and beverage industry, and it's to all parties' benefit that food items are traceable and protected against fraud [49]. This is where blockchain can provide great benefit in terms of transparency, efficiency, and security.

One real world example of blockchain in the food industry is IBM Food Trust, a platform that uses blockchain technology to track food products up and down supply chains. The IBM Food Trust network has been used by major corporations such as Walmart and Nestle [20], and the pilots with these companies have yielded impressive results. For example, an analysis of the time it takes to trace the origin of a pack of mangoes from Walmart found out that IBM Food Trust's technology reduced this time period from about 7 days to 2.2 seconds [38]. This is just one of IBM Food Trust's many use cases, which also includes protecting against food fraud, tracking sources of food-borne disease, sharing information with partners in a secure way, storing batch numbers and expiry dates, more efficient transport of goods, and more [36].

The power behind IBM Food Trust comes from a distributed ledger technology called Hyperledger Fabric. Hyperledger Fabric is an open source platform from the Linux Foundation that is commonly used in enterprise level blockchain platforms. The platform is highly customizable and provides robust security, privacy, and scalability [24]. There are several properties of Hyperledger Fabric that distinguish

it from other blockchain platforms and make it a suitable choice for IBM Food Trust and other similar enterprise applications.

Permissioned network: Other well-known blockchains such as Ethereum are built on a permissionless networks, which means that the data stored on the blockchain can be accessed by all participants. In the case of Hyperledger Fabric, however, all participants in the network must be known and approved and members must be authorized properly before joining the network. This is implemented by a virtual blockchain network that sits on top of the physical one, having its own access rules to minimize exposure of data. These features of the Hyperledger Fabric allow companies to ensure stronger privacy where needed [24, 31].

Channels: Hyperledger Fabric allows private transactions to take place between specific members of its network. These transactions take place in subnets called "Channels". Channels have their own separate ledgers, so transactions are not exposed to those outside of the channel [29].

Chaincode: Chaincode is the name given to smart contracts in Hyperledger Fabric. Chaincode programs can be written in Go, Java, or node.js and their purpose is to handle business logic. The organizations in the channel need to define the conditions that have to be met so that the chaincode executes and the transaction takes place [30].

Consensus mechanisms: Consensus mechanisms are pluggable in Hyperledger Fabric, which means that the algorithm used to reach consensus can be configured by the user to match the application's needs. Practical Byzantine Fault Tolerance, RAFT, and Solo are some algorithms that are most commonly used in Hyperledger Fabric [24].

Figure 3.3 shows an overview of how organizations, channels, and contracts interact in Hyperledger Fabric. Vehicle and Insurance are two private channels and an organization can participate in multiple channels. Two contracts named 'car' and 'insurance' are defined, and the endorsement policy is defined by the members of the channel the contract is in. For example, the car contract requires both participating organizations, ORG1 and ORG2, to sign the transactions before they are executed, while the insurance channel only requires ORG3 to sign.

3.2.2 Healthcare

Healthcare is an industry which encompasses many sub-fields such as public health, medical services, health insurance, medical equipment production, the pharmaceutical industry. Health data is considered one of the most sensitive types of data out there, so it could really benefit from blockchain as a way to ensure security. Current electronic health record and electronic medical record systems handle the

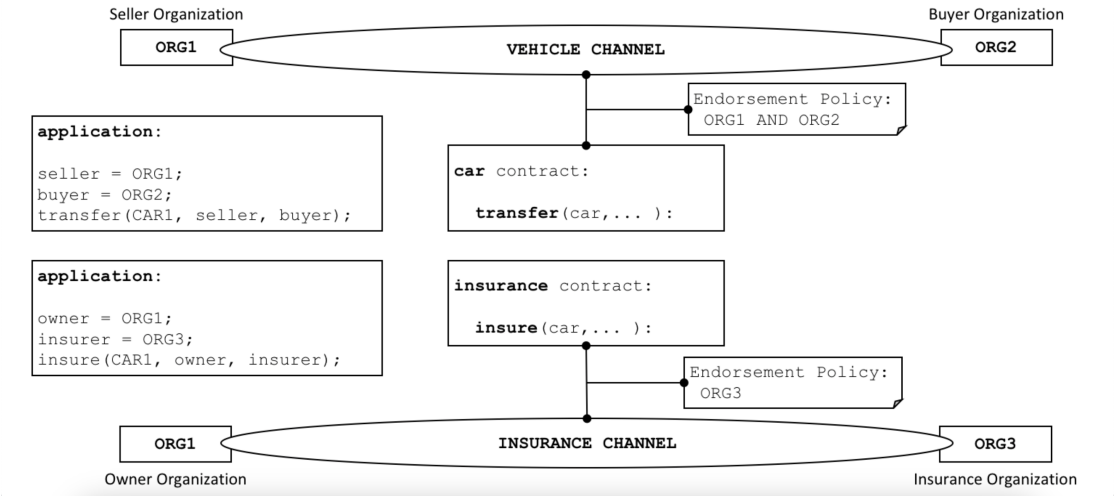


Figure 3.3: Diagram showing how channels, organizations, and contracts interact in Hyperledger Fabric [30].

security of medical records, data ownership and integrity, but not without issues. Blockchain can be used to overcome these challenges [48].

In healthcare, there are different ways technology is used to process and store medical information. The storage of patient medical records is crucial, as data breaches and leaks in healthcare are very dangerous [48]. Consequently, security and data integrity are very important. In addition, the fight against counterfeit medications is an important area in which technology is used. More than 700,000 people die from fake drugs, even with the current technologies used in healthcare [33]. Hence, transparency in medical information, data traceability, and integrity are crucial which blockchain offers. Let's delve into real applications of blockchain.

3.2.2.1 Modum.io

Modum.io is a Zurich-based startup that uses blockchain to provide monitoring and analytics tools for pharmaceutical supply chains. The company's main objective is to ensure that medical products are stored and transported safely. The moves of healthcare products are stored in a system that utilizes blockchain to ensure data integrity. Figure 3.4 highlights the use of blockchain in the architecture of the whole system.

The blockchain provides immutable, chronological records of medicine produced and transported. This allows for secure and traceable untampered data of medicine that has been shipped or transferred, with details such as container temperature. Modum.io uses the Ethereum network to utilize the benefits of the blockchain. The smart contracts written in the Ethereum network eliminate the need for intermediary services, that would normally track the medicine for good distribution practice (GDP) compliance and temperature checks. These smart contracts are executed automatically by the blockchain when there is a new shipment, new trade

or new medicine. The results are then recorded on the Ethereum blockchain and are immutable afterwards [10].

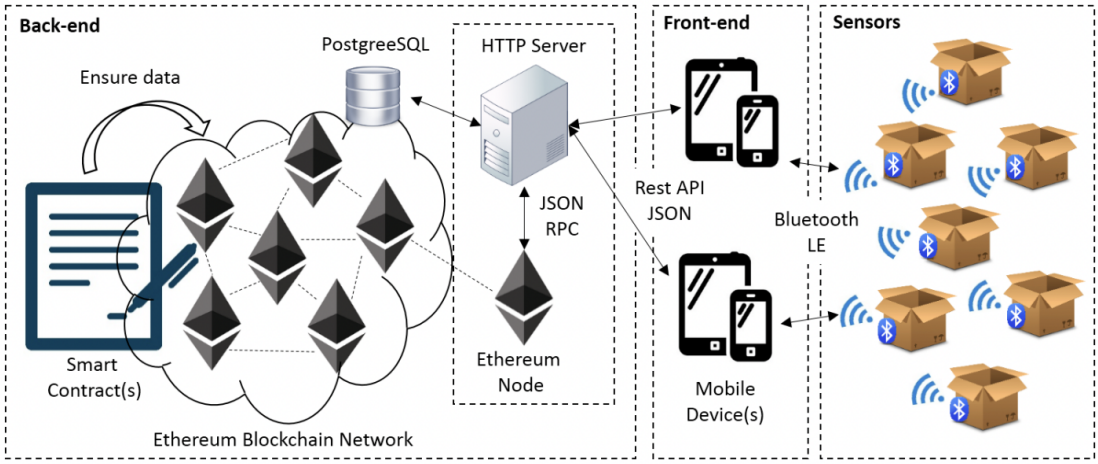


Figure 3.4: The architecture of Modum.io [10].

3.2.2.2 Gcoin

Gcoin is a blockchain system that aims to solve the problem of counterfeit drugs by utilizing blockchain. Gcoin records transactions of drugs, from production until it reaches patients. The blockchain has levels of authority that grant different sets of privileges to different groups. Figure 3.5. highlights these roles and how they interact with the blockchain. It can be seen that only large manufacturers and government agencies can mine, which is the process of creating new drugs on the blockchain. These drugs can then be traced back on the blockchain, as blockchains are immutable and transparent.

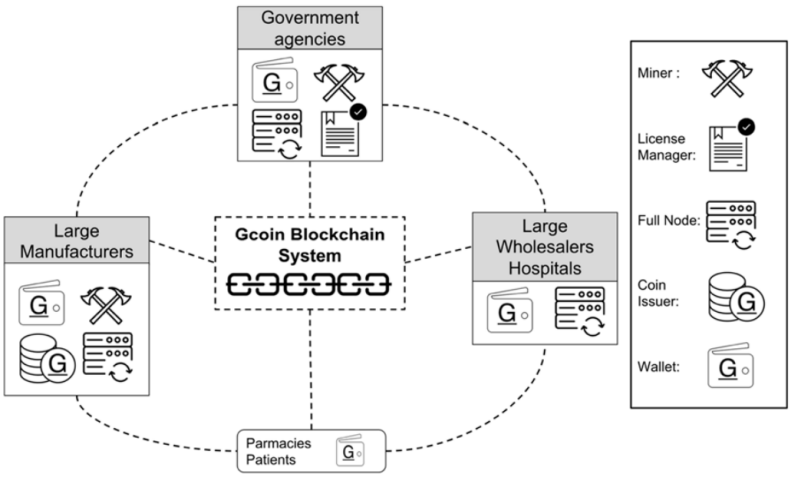


Figure 3.5: Roles of users on Gcoin [57].

Figure 3.6 shows the translation of the real life transportation/trading process of drugs (offline) in the blockchain. The process starts with the manufacturer

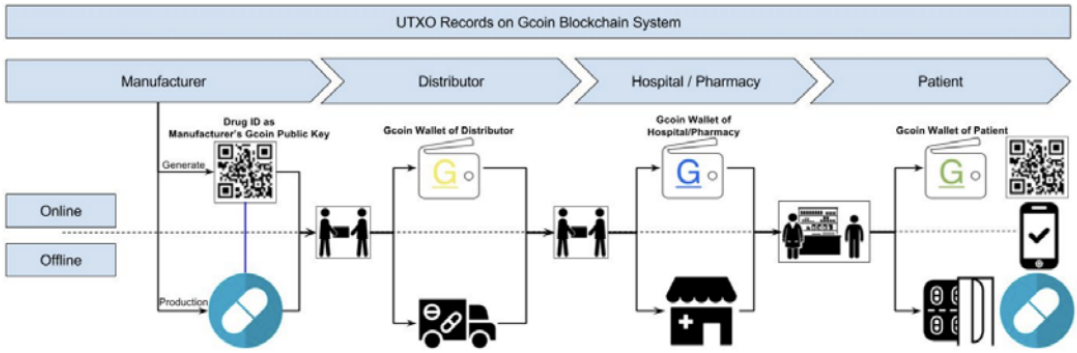


Figure 3.6: How Gcoin works [57].

producing the drug, and creating the drug on the blockchain. Then, moves on with the transactions between parties on the blockchain until it reaches the patient. However, one challenge of blockchains is that they can have poor performance because of consensus mechanisms. However, [57] concludes that the "processing ability of the Gcoin blockchain is more than sufficient.".

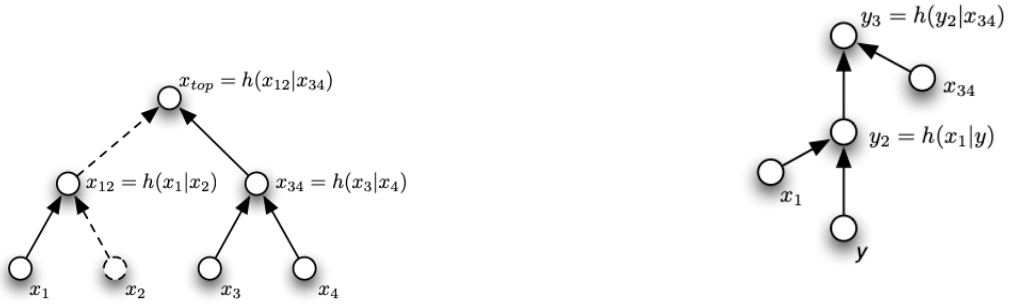
By using blockchain, counterfeit product detection in the pharmaceutical industry becomes much more secure and reliable. This would especially be useful in the export of drugs and vaccines into less developed countries, where counterfeit drugs are much more widespread and the existing systems in place are less trustworthy. Since blockchain is a decentralized system, the need to trust central authorities, intermediaries and third parties is significantly reduced, which simplifies international transactions and transports.

3.2.2.3 Blockchain-based Healthcare in Estonia

Estonia has implemented a system with the cooperation of a company called Guardtime to store the health data of its citizens. The system is built on top of the proprietary Keyless Signature Infrastructure (KSI), which stores over a million citizens' health records. The blockchain (KSI), only functions as a validator for the health data that is stored in a separate storage system. The blockchain tracks the creations or updates of health data, but not the data itself. The KSI technology is unique in not using a public key interface (PKI) to encode transactions and store them as a list in blocks [11, 27].

KSI uses hash trees, and does not use any cryptographic keys. Rather than storing the whole transactions on the blocks, KSI generates a signature directly from the data using a one-way, collision free hashing algorithm and aggregates these hashes in a hash tree. Then, to validate if a data is indeed in the blockchain, KSI confirms if the data exists in the true hash tree. This significantly reduces the transaction speed, as adding a new transaction becomes a lot faster, but the system becomes a lot more scalable, as it does not slow down with more transactions. The aggregated hash trees make the operations a lot faster. Figure 3.7a showcases the structure

of a hash tree, and figure 3.7b demonstrates how given a data y, the hash function can be used to climb up to the root of a hash tree. KSI concludes that if $x_{top} == y_3$ then y is valid [11].



(a) Hash tree structure [11].

(b) Validating data through hash tree [11].

Figure 3.7: Illustrations of how hash trees function in KSI.

The KSI blockchain provides tamper-proof, secure, transparent and scalable solution to validating health data. However, the health records themselves are still stored in a separate system, that can be vulnerable to cyber-attacks. Given Estonia stores over a million citizens' records on the blockchain, it demonstrates the most used utilization of blockchain in government healthcare services [27]. It also relieves a common drawback of using blockchain in commonly used systems which is backwards compatibility and the cost of adapting the existing system. By keeping the data in the currently used database and only using the blockchain for verification, this example shows how customizable blockchain technology can be.

3.2.3 Identity Verification

Identity verification today is a very vital part of an individual's life, as it emerges in many contexts such as opening bank accounts, voting, job applications, and international travel. However, identity related data is also a very vulnerable target and is involved in major crimes such as identity theft and fraud. Identification and certification documents are highly centralized, and are often issued by governmental organizations or private companies. This introduces a single point of failure and creates devastating results in the case of data leaks. For example, in 2023, a data breach concerning a Bangladeshi government website exposed the personal information of millions of citizens [3].

An alternative to these centralized identity management systems is decentralized identity, which can be achieved using blockchain technologies. The need for centralized identity in today's society arises from the fact that every individual needs to keep track of and deal with an increasingly high number of credentials, which includes hundreds of login credentials, certifications, and other authentication details [5]. Decentralized identification was also standardized by the World Wide Web

Consortium (W3C) in 2022 [51], establishing the fact that decentralized identity is not just a hot topic, it's also being recognized by global authorities.

The discussion around decentralized identity brought about a popular new approach: Self-Sovereign Identity (SSI), which is a term sometimes used interchangeably with decentralized identity. The main idea of SSI is that the user or the individual has full control over their identity. Within the concept of SSI, users can generate and present their verifiable credentials without reliance on any authority or third party [17]. Self sovereign identity should be decentralized, should be available to any authorized entity, and the owner of the identity should have full control over it. These requirements overlap well with the properties of blockchain, hence blockchain technology is very suitable for implementing self sovereign identity [23].

There are additional benefits to using blockchain for identity management and verification. One example is that it can improve authentication efficiency via technologies such as single sign on (SSO) [18]. Decentralized identity can also be used to provide identification to under-served communities such as undocumented people and refugees [54].

3.2.3.1 Truvera

Truvera by Dock Labs is a decentralized identity management platform that uses blockchain to store and verify credentials. Its core idea involves three parties, as shown in Figure 3.8: the issuer, the holder, and the verifier. The issuer can be a university for example, and the university can issue the holder a credential, such as a diploma. The holder now has full control of this diploma and can present it to a verifier, for example an employer. In centralized systems, the verifier has to reach out to the issuer to verify the document, whereas with systems such as Truvera the verifier does not have to come into contact with the issuer at all, the credential verification process can be handled between the holder and the verifier. This makes the process much more efficient and also puts the holder in charge of of what information they share about themselves [55].

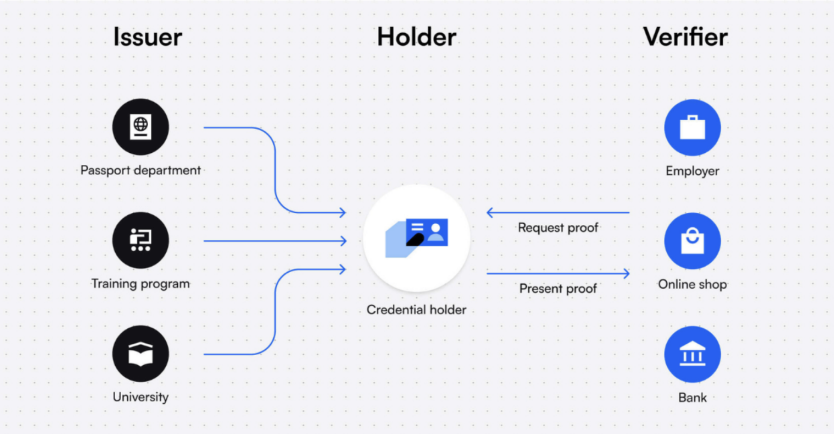


Figure 3.8: The three parties involved in decentralized identity management [55].

Truvera stores identity information on the blockchain via decentralized identifiers (DIDs). DIDs are made up of a combination of letters and numbers, and they are encrypted using public and private key pairs. The public DID of an issuing party is associated to a credential that they create, which can be used to verify that the issuer's private key was used to sign the credential [51].

In its architecture, Truvera uses a public blockchain to store DIDs. This ensures that even if the application is down or inaccessible, the credentials are still safe and verifiable. The verifiable credentials can also be stored on the client side, which makes them even safer and more easily accessible by users. The system architecture of Truvera can be seen in Figure 3.9.

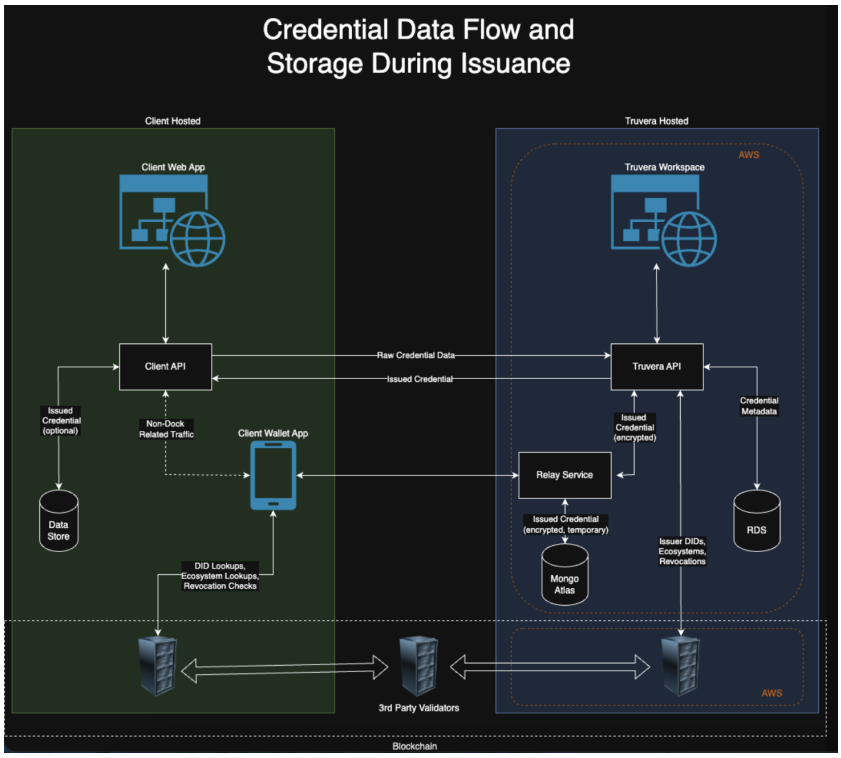


Figure 3.9: System architecture of Truvera [56].

3.2.4 Voting Systems

Elections are another area in which technology has been carefully utilized. The voting systems have progressed from paper ballots to vote collection on machines to online systems that allow voters to vote online [28]. These voting systems, which allow for the collection and storage of votes, require strong security and transparency, as well as voter authentication and vote validation [12].

The paper ballots are still used widely in elections, as well as some electronic voting systems. However, both have been known for their trade-offs. For instance, paper ballots need a lot of human labor in the preparation, processing, and validation steps, which make it error prone. On the other hand, electronic systems

are criticized for their centralized storage and non-transparency. Blockchain is utilized to overcome some of these challenges. The immutability and transparency of blockchains make it a good choice for voting systems [28].

3.2.4.1 Voatz

Voatz is a blockchain-based mobile voting application. It was used during 2018 midterm elections in West Virginia and 2024 federal elections in Mexico. Voatz does not have its system architecture published, and neither does it have many technical details shared to the public. citespecter2020ballot looks into the security of Voatz. As a result of their work, they have a vague understanding of the system architecture of Voatz, shown in Figure 3.10.

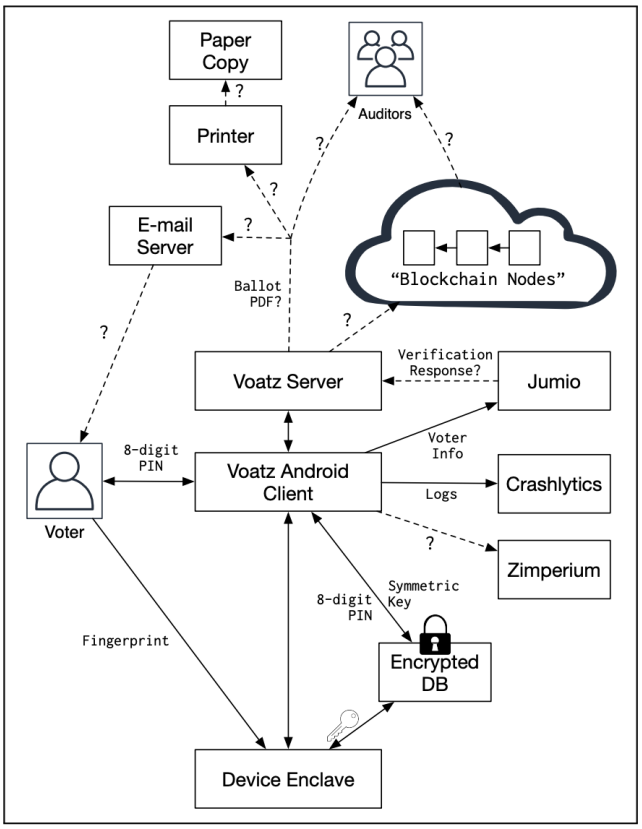


Figure 3.10: Observed system architecture of Voatz [50].

The blockchain stores the vote data, which ensures transparency and data integrity. The system also sends the votes to printers to be printed in paper copies. In Figure 3.10, the bottom half demonstrate the processes on local devices. The system takes care of the voter authentication mainly off the blockchain (on the device), that is, when the device casts a vote onto the blockchain it is after being authenticated as a citizen. The transaction on the blockchain then has a signature, that could be used to validate the vote on the blockchain.

The system overall has some security flows even though it is mostly built well and secure [50]. [25] concludes that after their security report, in 2020 Voatz was already working on to fix their security issues. All in all, the system is considered well built and working well [25].

3.2.4.2 Blockchain Voting in Zug

Another pilot of a blockchain based voting system was conducted in the city of Zug, Switzerland. The e-voting system was decentralized and was distributed over many servers. The city of Zug was already preparing extensively for this system as citizens were being provided with digital identities throughout the previous years [53].

This test of what kind of response blockchain voting would get from citizens was motivated by a number of advantages that this voting system offers. First, it was preferred because of security reasons – since the data was stored in multiple servers instead of one centralized location, it would be difficult for hackers to compromise the network. Immutability of blockchain was also brought up in the report covering this pilot, as it's crucial to ensure that once a vote is cast, it should not be tampered with in any way. Transparency also plays a big role as voters are able to see how their vote is processed and counted.

The implementation requires the voters to have their digital IDs as their public and private keys are used to encrypt and decrypt voting data. The U-Port SSI application is used in order to manage voter identities, as seen in the diagram in Figure 3.11.

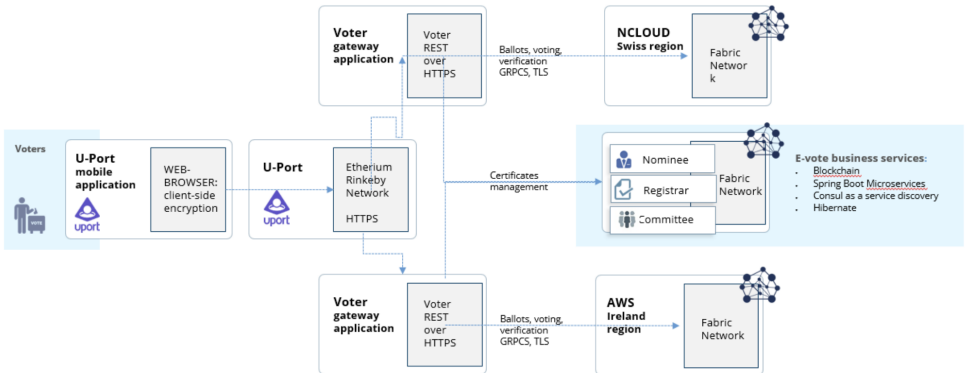


Figure 3.11: Architecture of the e-voting system used in Zug [22].

Some surveys were conducted to see the sentiment around e-voting/blockchain voting. Most citizens were content about this new system being implemented, and they expressed trust in the security of blockchain. As few as 2% of respondents were against e-voting. Even though most citizens gave answers that prefer e-voting to traditional voting systems, the actual participation in the e-voting pilot

was lower than expected. Overall, both the city of Zug and Switzerland in general are making more efforts towards adopting digital identities and blockchain voting in their governmental system [22].

3.2.5 Intellectual Property Protection

Intellectual property (IP) is defined as ideas and information of commercial value, and includes trademarks, copyright, artistic work, and patents. The usage of blockchain to protect intellectual property rights has less active use cases compared to the other sectors covered in this report, but it still has a promising future.

In the age of the internet, new challenges in the field of IP protection have arisen and novel methods and technologies should be utilized to ensure the protection of IP. With the emergence and rise of digital art and other forms of digital products, it has become increasingly difficult to prove ownership and protect these properties [37, 34].

In the heart of intellectual property protection lies the task of proving that an individual or organization is the owner of a certain property. Blockchain offers the ability to store immutable records in the distributing ledger, which prove that a certain transaction has been made. This paves the way for patents, trademarks, and copyright records to be written to the blockchain and thus an immutable proof of ownership will be available in a transparent and secure manner.

There's also the problem of royalty payments, which must be executed every time a copyrighted product is used by someone else. One such example is the usage of copyrighted music or artwork. This issue can also solved by blockchain – namely smart contracts. With the use of these self-executing agreements, an automatic payment can be made to the copyright holder every time their intellectual property is used. This makes royalties very efficient by automating the process and reduces the risk of unauthorized usage of someone else's property [39]. Overall, there are many potential applications of blockchain technology that provide secure and automated protection of IP.

3.2.5.1 KodakOne

Kodak is an American company that specializes in products related to film and digital photography. In 2018, Kodak launched a new digital rights management platform called KodakOne that has blockchain as its core technology.

The platform's target audience is photographers as it promises to streamline the process registering, monetizing, managing, and archiving their work. KodakOne makes use of a distributed ledger where all photograph licenses are written. Then, they can be paid for their photographs when they are used via smart contracts, and they can even be paid when an unlicensed usage of the photo is found, via the use of web-crawling software. KodakOne was targeted towards both professional

and amateur photographers and it was affirmed by the CEO that the platform would provide fair licensing to artists [9, 21].

Despite its promising premise, KodakOne failed to establish much success. One reason for its failure was that KodakCoin, the cryptocurrency associated with the platform, was criticized for being a cash grab. Kodak was not doing very well at the time and the company was chastised for attempting to profit from the blockchain and cryptocurrency wave [46]. Eventually, both the platform and the cryptocurrency project associated with it was shut down.

3.2.5.2 NFTs and OpenSea

Many uses of blockchain, including cryptocurrencies, involves fungible tokens. This means that tokens on the blockchain can be used interchangeably and are not unique in terms of their value. An alternative approach is storing non fungible tokens, commonly referred to as NFTs, on the blockchain. NFTs are digital assets with unique identifiers, and can also be used to represent real-world objects.

NFTs are commonly used to represent digital art, music, video game assets, and photography. Thus they have high potential in the field of IP protection. Applying for patents and trademarks, for instance, are time-consuming and expensive procedures that can take years. However using NFTs to represent the patent on the blockchain while the process is underway and recording patent-based transactions on the distributed ledger can ensure efficiency for all parties involved [7].

OpenSea is the largest marketplace for trading non fungible tokens, it's also a pioneer in the NFT industry. NFTs are written to the blockchain via a process called minting, where an immutable proof of ownership is recorded. OpenSea allows its creators to mint NFTs and collections of NFTs, which can then be exchanged between users [41]. This exchange takes place automatically when the buyer and seller agree, via smart contracts.

NFTs open up a lot of opportunities in IP protection. For starters, they are an immutable proof of authenticity that does not require a centralized entity to authenticate the property. Moreover, the transactions that took place regarding that property are very transparent. For example, in the case of art NFTs, the artist who created the piece has an easily accessible view of how their art is displayed or used [42]. This also allows for a community of people who are interested to form around that art or artist. These apply to other types of NFTs as well, such as photography, music, or gaming.

3.2.6 Banking and Finance

In the banking and finance sector, there are a lot of opportunities for decentralization, fraud prevention and added security that can be addressed by blockchain.

Finance is an industry that strives to keep up with the latest technology, and blockchain is no exception.

Smart contracts, one of the key features of blockchain, is a promising means to enhance payment transactions and reduce their cost. Using smart contracts, transactions carry out automatically when certain conditions are met. Moreover, they don't need to be validated by a third party or authority since the ledger is distributed and the nodes can reach consensus without a central entity. This way payment transactions execute efficiently compared to centralized systems [43]. Transparency is also achieved this way, as all members of the blockchain can view the transactions recorded on that ledger. Banks can make good use of distributed ledger technology in their credit systems, customer data management systems and financial asset management [26].

3.2.6.1 Ripple

With the growing need for more efficiency and security in cross-border payments, blockchain has high potential in managing international transactions. Nowadays, the use of blockchain in cross-border payment systems is becoming more and more mainstream. This is due to the many advantages that blockchain technology offers in improving cross-border payments, such as reducing the number of intermediate parties, offering transparency, and getting rid of the single point of failure that centralized systems have [19].

Ripple is a very popular decentralized payment system that uses blockchain as its core technology. Ripple is currently the leader in blockchain-based cross-border payment systems, and the code is open source. The system is preferred by many banks due to its promise to decrease payment time from days to seconds and enable transparency via the distributed ledger [4].

Financial transactions made using Ripple are recorded on the XRP Ledger (XRPL). XRPL is a decentralized blockchain built for business and it's not owned or controlled by Ripple, making it public and open to everyone [63]. A defining feature of Ripple and XRPL blockchain that is well-known and thoroughly studied is its consensus algorithm. This is also what sets Ripple apart from other well-known blockchains such as Bitcoin.

The consensus algorithm is called the Ripple Protocol or XRP Ledger Consensus Protocol (XRP LCP) and is much more efficient in terms of cost, energy consumption, and latency compared to Proof of Work and Proof of Stake algorithms used by other blockchains, making it a practical and sustainable method of achieving consensus [14]. XRP LCP is a Byzantine Fault Tolerant algorithm, which means that it still functions in the case of failed or malicious nodes. There are many servers or nodes in the XRPL network, and each of these nodes keep track of a list of other trusted nodes. These trusted nodes are also referred to as validators and they are chosen carefully to minimize the risk of most of them (about >80%) staying online and operational most of the time [62].

The aforementioned lists of validators is referred to as Unique Node Lists (UNLs) and are central to the Ripple Protocol. Figure 3.12 shows how UNLs are used to achieve consensus. Each node has their own version of what the current list of transactions to be written to the ledger looks like. Each transaction is then added to the new version of the ledger if the following condition is met: The transaction must be approved by a predetermined ratio of trusted nodes. Transactions that were not approved can stay in the proposed node and be evaluated in another iteration of the consensus algorithm [15].

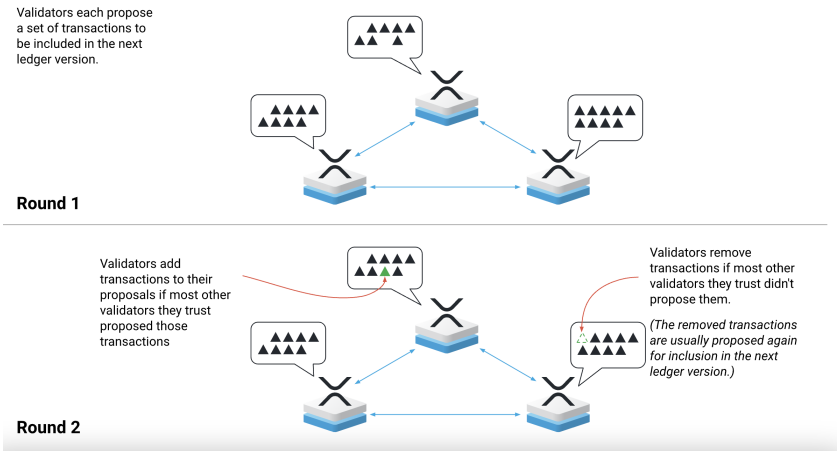


Figure 3.12: How validators reach consensus in Ripple Protocol/XRP LCP [15].

The UNLs must be chosen carefully because the consensus process is risky if there isn't enough overlap between the UNLs of the nodes in the network. Research shows that this overlap ratio is 90% – this means that 90% of nodes in each UNL must overlap [14]. If the overlap is not high enough, the nodes will fail to reach consensus which will result in a fork in the network. To combat the problem of how to configure the node lists so that the overlap goal is reached, lists of recommended validators are made available in the XRPL. Nodes that join the network can download the recommended validator lists and use them as their UNLs. There are official recommendation lists that are published by the XRP Ledger Foundation and Ripple, but anyone can create and publish such a list. The published list must be signed with the publisher's public key [62].

The efficient consensus algorithm allows Ripple to be an efficient and secure distributed ledger for cross-border payments and makes it a popular choice among financial institutions. The network of banks and payment providers worldwide that use Ripple is called RippleNet, and includes more than 100 financial institutions worldwide [1].

3.2.6.2 Santander One Pay FX

A prominent member of the RippleNet is Santander Bank, a financial services company based in Spain. Santander currently ranks as the 14th largest bank in the world [35].

In 2018, Santander rolled out their "One Pay FX" service for cross-border payments using blockchain. International money transfers are known to be slow, and in many cases the sender does not even know the exact amount of money they sent will actually be received. One Pay FX uses the power of Ripple to tackle these problems. The payment service promises to execute international money transfers within the same day, or at least the day after. The decentralized system removes excessive middle-men and reduces costs, which allows Santander to offer these transfers without any fees. The users can also see exactly how much money the receiver will get, in the currency that they will receive it [47].

This use case highlights the practicality of blockchain in banking, as it makes payments very efficient and benefits both banks and customers.

3.2.6.3 UBS Digital Bond

Cryptocurrencies are represented on the blockchain via tokens. Representation of financial assets via tokens is not just limited to cryptocurrencies, the usage can be expanded to include other financial assets. Stock shares, bonds, and even physical assets such as real estate can be tokenized.

UBS, which is not only the largest bank in Switzerland but also the largest private bank in the world [60], took an eminent step in 2022 by introducing the world's first digital bond. The bond can be traded both on blockchain platforms and traditional markets. The advantage of decentralization offered by blockchain is very useful in this case, since the bond can be settled without a clearing party [58].

3.3 Conclusion

Although blockchain was initially developed for cryptocurrencies, it now has many practical use cases outside of it. The reason why it took off so much in the context of the industries covered in this report is its properties that make blockchain a truly unique and effective technology. Decentralization, immutability, consensus, smart contracts, and other features of blockchain allow for a lot of opportunities that were previously not possible.

Whether or not blockchain is the ideal technology to solve a particular problem should be evaluated on a case-by-case basis. In some cases it makes processes much more simple and efficient while in others it may be overkill. The implementation of blockchain would be a tedious transition in some instances and would require a lot of new legislation and effort to convert. Some governments have already adopted blockchain while some are more skeptical.

Nevertheless, it's clear that blockchain's practicality is not limited to cryptocurrencies and it's worthwhile to investigate how it can be integrated into other systems and industries.

References

[1] Amazon Web Services. *Ripple Case Study – Amazon Web Services (AWS)*. Accessed: 2025-04-14. 2025. URL: %5Curl%7Bhttps://aws.amazon.com/partners/success/ripple/%7D.

[2] Amazon Web Services. *What is Blockchain?* Accessed: 2025-04-14. 2025. URL: %5Curl%7Bhttps://aws.amazon.com/what-is/blockchain/%7D.

[3] Daryna Antoniuk. "Bangladesh government fixes website that leaked personal data of 50 million citizens". In: *The Record* (July 2023). Accessed: 2025-04-14. URL: %5Curl%7Bhttps://therecord.media/bangladesh-government-data-leak-50-million%7D.

[4] Frederik Armknecht et al. "Ripple: Overview and outlook". In: *Trust and Trustworthy Computing: 8th International Conference, TRUST 2015, Heraklion, Greece, August 24-26, 2015, Proceedings 8*. Springer. 2015, pp. 163–180.

[5] Oscar Avellaneda et al. "Decentralized identity: Where did it come from and where is it going?" In: *IEEE Communications Standards Magazine* 3.4 (2019), pp. 10–13.

[6] Pedro Azevedo, Jorge Gomes, and Mário Romão. "Supply chain traceability using blockchain". In: *Operations Management Research* 16.3 (2023), pp. 1359–1381.

[7] Seyed Mojtaba Hosseini Bamakan et al. "Patents and intellectual property assets as non-fungible tokens; key technologies and challenges". In: *Scientific reports* 12.1 (2022), p. 2178.

[8] Muhammad Nasir Mumtaz Bhutta et al. "A Survey on Blockchain Technology: Evolution, Architecture and Security". In: *IEEE Access* 9 (2021), pp. 61048–61073. DOI: 10.1109/ACCESS.2021.3072849.

[9] Rebecca Blake. "KODAKOne Uses Blockchain Technology and Cryptocurrency to Manage Image Rights". In: *Graphic Artists Guild* (Mar. 2018). Accessed: 2025-04-14. URL: %5Curl%7Bhttps://graphicartistsguild.org/kodakone-uses-blockchain-technology-and-cryptocurrency-to-manage-image-rights/%7D.

[10] Thomas Bocek et al. "Blockchains everywhere - a use-case of blockchains in the pharma supply-chain". In: *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. 2017, pp. 772–777. DOI: 10.23919/INM.2017.7987376.

[11] Ahto Buldas, Andres Kroonmaa, and Risto Laanoja. *Keyless Signatures' Infrastructure: How to Build Global Distributed Hash-Trees*. Tech. rep. Guardtime, 2013. URL: %5Curl%7Bhttps://guardtime.com/files/BuKL13.pdf%7D.

[12] Prashanth P. Bungale and Swaroop Sridhar. *Requirements for an Electronic Voting System*. Tech. rep. Department of Computer Science, Johns Hopkins University, 2003. URL: %5Curl%7Bhttps://www.cs.jhu.edu/~rubin/courses/sp03/group-reports/group4/group4_requirements.pdf%7D.

[13] Fran Casino, Thomas K. Dasaklis, and Constantinos Patsakis. "A systematic literature review of blockchain-based applications: Current status, classifica-

tion and open issues". In: *Telematics and Informatics* 36 (2019), pp. 55–81. ISSN: 0736-5853. DOI: https://doi.org/10.1016/j.tele.2018.11.006.

[14] Brad Chase and Ethan MacBrough. "Analysis of the XRP ledger consensus protocol". In: *arXiv preprint arXiv:1802.07242* (2018).

[15] Dave Cohen, David Schwartz, and Arthur Britto. *Consensus Structure*. Accessed: 2025-04-14. 2024. URL: %5Curl%7Bhttps://xrpl.org/docs/concepts/consensus-protocol/consensus-structure%7D.

[16] Council of Supply Chain Management Professionals (CSCMP). *Supply Chain Management Definitions and Glossary of Terms*. Accessed: 2025-04-14. Feb. 2025. URL: %5Curl%7Bhttps://cscmp.org/CSCMP/CSCMP/Educate/SCM_Definitions_and_Glossary_of_Terms.aspx%7D.

[17] Špela Čučko and Muhamed Turkanović. "Decentralized and self-sovereign identity: Systematic mapping study". In: *IEEe Access* 9 (2021), pp. 139009–139027.

[18] Jan De Clercq. "Single sign-on architectures". In: *International Conference on Infrastructure Security*. Springer. 2002, pp. 40–58.

[19] Qing Deng. "Application analysis on blockchain technology in cross-border payment". In: *5th International Conference on Financial Innovation and Economic Development (ICFIED 2020)*. Atlantis Press. 2020, pp. 287–295.

[20] Kyt Dotson. "IBM Food Trust commercial blockchain launch links food safety from farm to dinner table". In: *SiliconANGLE* (Oct. 2018). Accessed: 2025-04-14. URL: %5Curl%7Bhttps://siliconangle.com/2018/10/08/ibm-food-trust-commercial-blockchain-launch-links-food-safety-farm-dinner-table/%7D.

[21] Eastman Kodak Company. *Kodak and WENN Digital Partner to Launch Major Blockchain Initiative and Cryptocurrency*. Accessed: 2025-04-14. Jan. 2018. URL: %5Curl%7Bhttps://www.kodak.com/en/company/press-release/blockchain-initiative/%7D.

[22] *Evaluation of the Blockchain Vote in the City of Zug*. Tech. rep. Accessed: 2025-04-14. City of Zug, Nov. 2018. URL: %5Curl%7Bhttps://www.stadtzug.ch/_docn/1938568/eVoting_Final_Report_ENG.pdf%7D.

[23] Md Sadek Ferdous, Farida Chowdhury, and Madini O Alassafi. "In search of self-sovereign identity leveraging blockchain technology". In: *IEEE access* 7 (2019), pp. 103059–103079.

[24] GeeksforGeeks. *Hyperledger Fabric in Blockchain*. Accessed: 2025-04-14. Apr. 2023. URL: %5Curl%7Bhttps://www.geeksforgeeks.org/hyperledger-fabric-in-blockchain/%7D.

[25] Dan Guido. *Our Full Report on the Voatz Mobile Voting Platform*. Trail of Bits Blog. Mar. 2020. URL: %5Curl%7Bhttps://blog.trailofbits.com/2020/03/13/our-full-report-on-the-voatz-mobile-voting-platform/%7D.

[26] Ye Guo and Chen Liang. "Blockchain application and outlook in the banking industry". In: *Financial innovation* 2.1 (2016), p. 24.

[27] Thomas F Heston. "A case study in blockchain health care innovation". In: *International Journal of Current Research* 9.11 (2017), pp. 60587–60588.

[28] Jun Huang et al. "The Application of the Blockchain Technology in Voting Systems: A Review". In: *ACM Comput. Surv.* 54.3 (Apr. 2021). ISSN: 0360-0300. DOI: 10.1145/3439725.

[29] Hyperledger Fabric Documentation Contributors. *Channels*. Accessed: 2025-04-14. 2025. URL: %5Curl%7Bhttps://hyperledger-fabric.readthedocs.io/en/latest/channels.html%7D.

[30] Hyperledger Fabric Documentation Contributors. *Smart Contracts and Chaincode*. Accessed: 2025-04-14. 2025. URL: %5Curl%7Bhttps://hyperledger-fabric.readthedocs.io/en/release-2.5/smartcontract/smartcontract.html%7D.

[31] IBM. *What is Hyperledger Fabric?* Accessed: 2025-04-14. 2025. URL: %5Curl%7Bhttps://www.ibm.com/think/topics/hyperledger%7D.

[32] Investopedia Staff. *What Are Consensus Mechanisms in Blockchain and Cryptocurrency?* 2018. URL: %5Curl%7Bhttps://www.investopedia.com/terms/c/consensus-mechanism-cryptocurrency.asp%7D (visited on 04/23/2025).

[33] IPN. *Fake drugs kill over 700,000 people every year – new report*. Press release. June 2013. URL: %5Curl%7Bhttp://archive.is/ipW8i%7D.

[34] Kensuke Ito and Marcus O'Dair. "A critical examination of the application of blockchain technology to intellectual property management". In: *Business Transformation through Blockchain: Volume II* (2019), pp. 317–335.

[35] Adrian Jimenea, John Wu, and Harry Terris. *The world's largest banks by assets, 2024*. Accessed: 2025-04-14. Apr. 2024. URL: %5Curl%7Bhttps://www.spglobal.com/market-intelligence/en/news-insights/research/the-worlds-largest-banks-by-assets-2024%7D.

[36] Reshma Kamath. "Food traceability on blockchain: Walmart's pork and mango pilots with IBM". In: *The Journal of the British Blockchain Association* 1.1 (2018).

[37] Trevor Krajewski and Rich Lettiere. "Efforts Integrating Blockchain with Intellectual Property". In: *les Nouvelles-Journal of the Licensing Executives Society* 54.1 (2019).

[38] LF Decentralized Trust. *How Walmart brought unprecedented transparency to the food supply chain with Hyperledger Fabric*. Accessed: 2025-04-14. 2023. URL: %5Curl%7Bhttps://www.lfdecentralizedtrust.org/case-studies/walmart-case-study%7D.

[39] Nadcab Labs. *Advanced Smart Contracts Simplify Royalty Payments*. Accessed: 2025-04-14. 2025. URL: %5Curl%7Bhttps://www.nadcab.com/blog/royalty-contract-in-smart-contract%7D.

[40] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. Accessed: 2025-04-22. 2008. URL: %5Curl%7Bhttps://bitcoin.org/bitcoin.pdf%7D.

[41] OpenSea. *Drops on OpenSea: Everything You Need to Know*. Accessed: 2025-04-14. Sept. 2023. URL: %5Curl%7Bhttps://opensea.io/learn/nft/drops-on-opensea%7D.

[42] OpenSea. *What Are Art NFTs? Learn How They're Changing the Art World*. Accessed: 2025-04-14. Sept. 2023. URL: %5Curl%7Bhttps://opensea.io/learn/nft/art-nfts%7D.

[43] Ritesh Patel, Milena Migliavacca, and Marco E Oriani. "Blockchain in banking and finance: A bibliometric review". In: *Research in International Business and Finance* 62 (2022), p. 101718.

[44] Guido Perboli, Stefano Musso, and Mariangela Rosano. "Blockchain in logistics and supply chain: A lean approach for designing real-world use cases". In: *Ieee Access* 6 (2018), pp. 62018–62028.

[45] Mayank Raikwar, Danilo Gligoroski, and Katina Kralevska. "SoK of Used Cryptography in Blockchain". In: *IEEE Access* 7 (Oct. 2019), pp. 1–1. DOI: 10.1109/ACCESS.2019.2946983.

[46] Kevin Roose. "Kodak's Dubious Cryptocurrency Gamble". In: *The New York Times* (Jan. 2018). Accessed: 2025-04-14. URL: %5Curl%7Bhttps://www.nytimes.com/2018/01/30/technology/kodak-blockchain-bitcoin.html%7D.

[47] Santander UK. *Santander launches the first blockchain-based international money transfer service across four countries*. Accessed: 2025-04-14. Apr. 2018. URL: %5Curl%7Bhttps://www.santander.co.uk/about-santander/media-centre/press-releases/santander-launches-the-first-blockchain-based-international-money-transfer-service-across-four%7D.

[48] Ayesha Shahnaz, Usman Qamar, and Ayesha Khalid. "Using blockchain for electronic health records". In: *IEEE access* 7 (2019), pp. 147782–147795.

[49] Vinay Singh and Sanjeev Kumar Sharma. "Application of blockchain technology in shaping the future of food industry based on transparency and consumer trust". In: *Journal of Food Science and Technology* 60.4 (2023), pp. 1237–1254.

[50] Michael A Specter, James Koppel, and Daniel Weitzner. "The ballot is busted before the blockchain: A security analysis of voatz, the first internet voting application used in {US}. federal elections". In: *29th USENIX Security Symposium (USENIX Security 20)*. 2020, pp. 1535–1553.

[51] Manu Sporny et al. *Decentralized Identifiers (DIDs) v1.0*. W3C Recommendation. Accessed: 2025-04-14. World Wide Web Consortium (W3C), July 2022. URL: %5Curl%7Bhttps://www.w3.org/TR/did-1.0/%7D.

[52] Stephanie Susnjara and Ian Smalley. *What Is Blockchain?* IBM. n.d. URL: %5Curl%7Bhttps://www.ibm.com/think/topics/blockchain%7D (visited on 04/23/2025).

[53] SWI swissinfo.ch. "Switzerland's first municipal blockchain vote hailed a success". In: *SWI swissinfo.ch* (July 2018). Accessed: 2025-04-14. URL: %5Curl%7Bhttps://www.swissinfo.ch/eng/business/crypto-valley-_-switzerland-s-first-municipal-blockchain-vote-hailed-a-success/44230928%7D.

[54] Hasan Syed. "Power to The People: How Blockchain Based Digital Identity Can Empower Disadvantaged Individuals". In: (2019).

[55] Truvera Documentation Contributors. *Decentralized Identity Explained*. Accessed: 2025-04-14. 2025. URL: %5Curl%7Bhttps://docs.truvera.io/readme/decentralized-identity-explained%7D.

[56] Truvera Documentation Contributors. *System Architecture*. Accessed: 2025-04-14. 2025. URL: %5Curl%7Bhttps://docs.truvera.io/system-architecture%7D.

[57] Jia-Hng Tseng et al. "Governance on the Drug Supply Chain via Gcoin Blockchain". In: *International Journal of Environmental Research and Public Health* 15.6 (2018), p. 1055. DOI: 10.3390/ijerph15061055.

[58] UBS AG. *UBS AG launches the world's first digital bond that is publicly traded and settled on both blockchain-based and traditional exchanges*. Accessed: 2025-04-14. Nov. 2022. URL: %5Curl%7Bhttps://www.ubs.com/global/en/media/display-page-ndp/en-20221103-digital-bond.html%7D.

[59] Ingo Weber et al. "Untrusted business process monitoring and execution using blockchain". In: *Business Process Management: 14th International Conference, BPM 2016, Rio de Janeiro, Brazil, September 18-22, 2016. Proceedings 14*. Springer. 2016, pp. 329–347.

[60] Oliver Williams. *Global Assets Under Management (AUM) Ranking: Insights from Private Banker*. Accessed: 2025-04-14. July 2019. URL: %5Curl%7Bhttps://www.privatebankerinternational.com/special-reports/global-aum-ranking/%7D.

[61] Yang Xiao et al. "A Survey of Distributed Consensus Protocols for Blockchain Networks". In: *IEEE Communications Surveys & Tutorials* 22.2 (2020), pp. 1432–1465. DOI: 10.1109/COMST.2020.2969706.

[62] XRPL.org Contributors. *Unique Node List (UNL)*. Accessed: 2025-04-14. 2025. URL: %5Curl%7Bhttps://xrpl.org/docs/concepts/consensus-protocol/unl%7D.

[63] XRPL.org Contributors. *XRPL Overview*. Accessed: 2025-04-14. 2025. URL: %5Curl%7Bhttps://xrpl.org/about%7D.

[64] Jesse Yli-Huumo et al. "Where is current research on blockchain technology? – a systematic review". In: *PloS one* 11.10 (2016), e0163477.

[65] Zibin Zheng et al. "An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends". In: *2017 IEEE International Congress on Big Data (BigData Congress)*. 2017, pp. 557–564. DOI: 10.1109/BigDataCongress.2017.85.

Chapter 4

An Overview of Machine Unlearning for Large Language Models

Jamo Sharif

Large Language Models (LLMs) have demonstrated remarkable capabilities across natural language tasks. However, their tendency to memorize sensitive, copyrighted, or harmful content introduces significant privacy, safety, and compliance risks. In response, machine unlearning has emerged as a critical framework for selectively removing unwanted knowledge from LLMs without full retraining. This report provides a comprehensive overview of machine unlearning for LLMs, outlining the key motivation, technical challenges, and distinctions between fine-tuning-based and direct unlearning approaches. Furthermore, it introduces a taxonomy of unlearning methods, spanning gradient-based updates, parameter-efficient fine-tuning, influence-function approaches, and input-level interventions. Evaluation metrics are also reviewed, covering forgetting efficacy, utility preservation, and computational efficiency. Practical applications in data privacy, safety alignment, and black-box API settings are discussed, along with open challenges such as adversarial robustness, continual unlearning, and multi-modal extension. This work aims to support the development of scalable, responsible, and verifiable unlearning strategies for modern LLMs.

Contents

4.1 Introduction 75
4.2 Background 76
4.2.1 LLMs 76
4.2.2 From Machine Learning to Machine Unlearning 76
4.3 Machine Unlearning for LLMs 78
4.3.1 Challenges 78
4.3.2 Defining LLM Unlearning 79
4.3.3 Mathematical Formulation 79
4.4 Methods for LLM Unlearning 80
4.4.1 Model-based Methods 80
4.4.2 Input-based Methods 84
4.4.3 Cross-Cutting Principles 84
4.5 Evaluation Metrics and Benchmarks for LLM Unlearning . . . 86
4.5.1 Evaluation Framework 86
4.5.2 Forgetting Effectiveness 87
4.5.3 Utility Preservation 88
4.5.4 Efficiency and Scalability 89
4.6 Application and Case Studies 89
4.6.1 Data Privacy Compliance 90
4.6.2 Copyright-Protected Content 90
4.6.3 Safety Alignment and Toxic Content Removal 90
4.6.4 Black-Box and API Settings 90
4.7 Discussion 91
4.8 Future Research Directions 91
4.9 Conclusion 93

4.1 Introduction

LLMs have achieved remarkable success across a wide range of natural language processing tasks, driven by their pre-training on massive and diverse text corpora [13, 52]. However, this reliance on large-scale data also raises serious concerns: models can inadvertently memorize and expose sensitive personal information, copyright-protected content, or harmful and biased language patterns [13, 26]. At the same time, emerging regulations such as the EU General Data Protection Regulation (GDPR) and the proposed EU AI Act grant data subjects the *"right to be forgotten"*, requiring mechanisms to remove specific information from deployed models on demand [13].

Conventional approaches involve a full or partial retraining of the model after unwanted data have been removed. However, this approach is very expensive for LLMs - even a single full retraining run can consume millions of GPU hours and enormous energy resources [13, 52]. To address these challenges, the field of machine unlearning has emerged, with the objective of selectively removing the influence of specific data or behaviors of a model without affecting its overall performance. Initial unlearning efforts focused on fine-tuned models, employing techniques such as gradient ascent on the *"forget set"* relabeling with random or adversarial samples, and low-rank adaptation to remove copyrighted or harmful content [1, 28]. Recent work has begun to look at pre-trained LLMs and examine paradigms such as localized parameter editing, activation interference, auxiliary teacher models, and input/output-based interventions [52, 26].

Despite the growing literature on LLM unlearning, to date there is no single reference that systematically organizes existing unlearning methods, evaluation metrics, and benchmark datasets in the context of LLMs. This paper, *"An Overview of Machine Unlearning for Large Language Models"*, addresses the gap by:

- Formalizing the problem definition and unlearning paradigms for LLMs, distinguishing between fine-tuning-first and direct-unlearning approaches [13].
- Presenting a comprehensive taxonomy of existing methods - including direct gradient-based updates, localized parameter modifications, auxiliary model strategies, and input/output interventions - and discussing their trade-offs [13, 26].
- Reviewing the current state of evaluation: how to measure forgetting efficacy versus utility preservation, and surveying common benchmarks [13, 1].
- Highlighting open challenges and future directions, such as extending unlearning to multi-modal models, improving adversarial robustness, and standardizing unlearning scope definitions [26, 28].

This report summarizes recent research into practical recommendations. It guides developers and researchers in creating LLMs that are safer, privacy compliant, and ethical.

4.2 Background

To understand machine unlearning in LLMs, two fundamental components must be introduced. Section 4.2.1 presents LLMs, explaining how they are built and trained on vast textual data. Section 4.2.2 then provides an overview of core machine learning concepts and introduces traditional unlearning techniques. Together, these sections offer the necessary context to explore the challenges and methods of unlearning in LLMs.

4.2.1 LLMs

LLMs are deep neural networks, typically based on the Transformer architecture, that are pre-trained on massive, diverse text corpora to learn the conditional probability of the next token given its context. By encoding both linguistic structure and world knowledge into hundreds of millions or even hundreds of billions of parameters, LLMs can generate coherent, context-aware text, translate languages, answer questions, summarize documents, and perform many other NLP tasks [10, 3].

Prominent examples include BERT [10], GPT-3 [3], T5 [41], and RoBERTa [27], which have set state-of-the-art records across benchmarks for understanding and generation. These models can generate human-like text for chatbots, virtual assistants, and content-creation tools. They are also used in many specialized domains [10, 3].

4.2.2 From Machine Learning to Machine Unlearning

4.2.2.1 Machine Learning

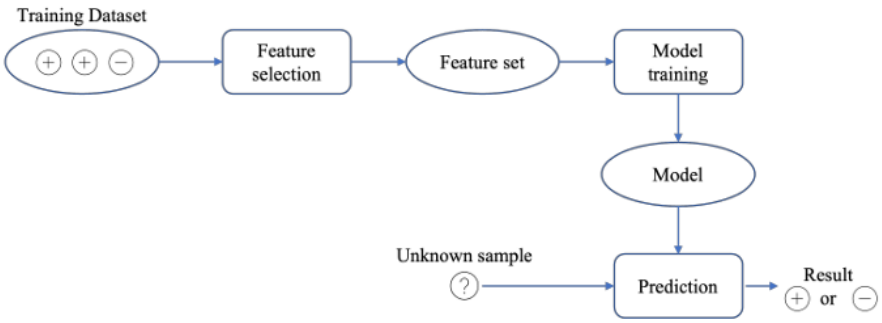


Figure 4.1: Machine Learning System [56].

Machine learning models learn from large datasets by adjusting their parameters to minimize a loss function over training examples. This iterative process involves selecting relevant features, training the model on labeled data, and using

the learned parameters to make predictions. As shown in Figure 4.1, a general machine learning system typically consists of three core stages: feature selection, model training, and prediction. Over successive iterations, the model becomes increasingly accurate in its predictions [56, 2].

However, this learning process can also lead to unintended memorization of sensitive or proprietary data, making models vulnerable to privacy attacks and regulatory non-compliance (e.g., GDPR's *"right to be forgotten"*) [2, 56]. Moreover, malicious actors can poison the training data to inject harmful behaviors, which requires techniques that can selectively remove such influences without resorting to full retraining [56].

4.2.2.2 Traditional Machine Unlearning

Traditional machine unlearning methods aim to remove the effect of specific data points from a trained model. Figure 4.2 shows the high-level anatomy of an unlearning algorithm. As illustrated in Figure 4.2, the unlearning algorithm takes three inputs: (i) a pretrained model; (ii) a forget set (examples to be erased); and (iii) a retain set (examples to preserve) [42].

It then produces a new model in which the influence of the forget set has been removed. In the ideal case, the resulting model is statistically indistinguishable from a model trained from scratch on the data minus the forget set [42].

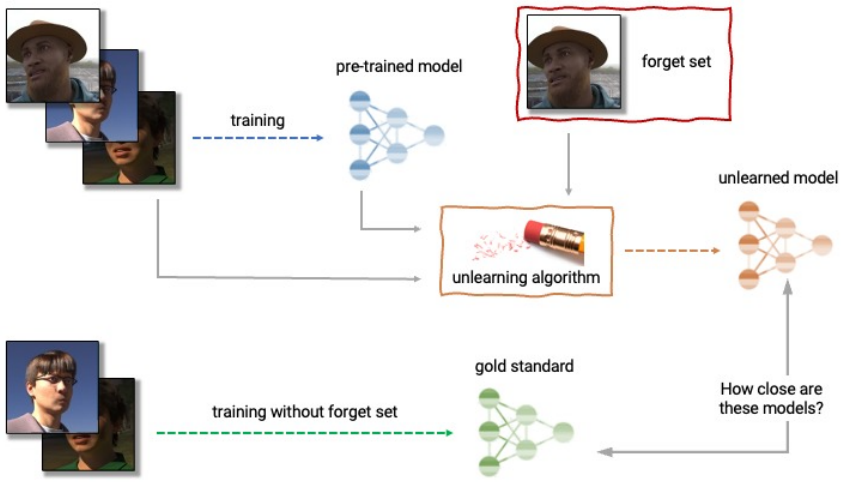


Figure 4.2: Anatomy Machine Unlearning [42]

With this framework in mind, unlearning techniques can be broadly classified into two families:

- **Exact unlearning** reconstructs a model as if the target data were never seen, typically by retraining from scratch on the remaining dataset. While it offers the strongest removal guarantees, its computational cost is often prohibitive for large models or frequent requests [2, 51].

- **Approximate unlearning** applies efficient updates to the existing model to reduce the influence of the forgotten data. Techniques include:
  - *SISA training* (Sharded, Isolated, Sliced, Aggregated), which shards the data and limits each point's impact to a small submodel, reducing retraining time upon unlearning requests [2, 56].
  - *Influence-function adjustments*, which approximate the effect of removing a data point by inverting the Hessian of the loss and adjusting parameters accordingly [51].
  - *Gradient-based interventions*, which perform targeted gradient ascent on the *'forget set'* and optional gradient descent on retain sets to erase specific knowledge [51, 56].

Although approximate methods greatly improve efficiency, they provide weaker theoretical guarantees and may leave residual data traces. Balancing removal accuracy, utility preservation, and computational cost remains a challenge [51, 56].

4.3 Machine Unlearning for LLMs

Machine unlearning in the context of LLMs involves the targeted removal of specific data or behaviors from a model without compromising its overall utility. Unlike traditional machine unlearning settings, LLMs introduce new challenges. This section explores the key difficulties, formal definitions, and mathematical frameworks that underpin unlearning in modern LLMs.

4.3.1 Challenges

Machine unlearning for LLMs brings several novel difficulties. As already mentioned in Section 4.2.1, LLMs are trained on vast, heterogeneous corpora, often embedding biases or memorizing sensitive, confidential content. Thus, precisely defining unlearning targets, whether specific subsets of training data or higher-level knowledge concepts, is inherently challenging. Current machine unlearning methods for LLMs tend to be highly context- and task-specific, and there remains no standardized unlearning corpus [26, 29, 13, 46, 20, 17, 8, 50, 54, 57, 53, 24].

Furthermore, as model sizes grow and access shifts toward black-box APIs, designing machine unlearning techniques that scale and adapt without full retraining is hard. Likewise, assessing their effectiveness is problematic when retraining is not a viable option. [26, 4, 5]. Therefore, innovations such as in-context unlearning and fictitious unlearning have begun to address black-box and synthetic-data settings [38, 33, 26].

Moreover, the precise scope of what to forget versus what to retain is often underspecified [35]. Effective machine unlearning must remove only the targeted knowledge while preserving all unrelated capabilities, yet drawing that boundary is as much art as science [3, 41, 49, 43]. Finally, the absence of adversarial and mechanistic evaluations leaves unlearning vulnerable, allowing attackers to reconstruct or jailbreak forgotten secrets [15, 31, 30, 45]. Rigorous benchmarks and threat models are urgently needed to close these gaps [26].

4.3.2 Defining LLM Unlearning

Based on classic unlearning frameworks and recent advances tailored to LLMs, LLM unlearning can be defined as follows [2, 20, 23, 38, 26, 53, 16, 33, 25]:

> **Problem (LLM Unlearning):** *How can specified unlearning targets and their associated capabilities be removed efficiently and effectively, without compromising performance on all other tasks?*

The above problem can be broken down into four dimensions:

1. **Unlearning targets:** These may be raw data points (e.g., toxic or copyrighted text), or higher level concepts (e.g., *"Harry Potter"* lore) that must be removed wherever they appear [29, 53, 11].
2. **Influence erasure:** Both data source contributions and internal model components that generate unwanted outputs must be addressed. Complete erasure demands robustness against prompts that probe entangled or generalized knowledge [37, 31].
3. **Unlearning effectiveness:** Effectiveness splits into *in-scope* measures - whether the model fails on target prompts - and *out-of-scope* checks - whether it still succeeds on safe tasks. Determining which prompts belong to which category is itself a challenge [14, 9].
4. **Efficiency and feasibility:** Even approximate unlearning methods can be expensive on 100B+ parameters. The deployment of these techniques in black-box or memory-constrained environments introduces additional constraints [38, 55].

4.3.3 Mathematical Formulation

A common formulation for LLM unlearning is [26]:

$$\min_{\theta} \underbrace{\mathbb{E}_{(x,y_f)\in D_f}\big[\ell(y_f \mid x;\theta)\big]}_{\text{forget}} + \lambda \underbrace{\mathbb{E}_{(x,y)\in D_r}\big[\ell(y \mid x;\theta)\big]}_{\text{retain}},$$

where:

- D_f is the *forget set*, containing examples to be erased, such as toxic prompt-completion pairs (i.e., input-response examples).
- D_r is the *retain set*, i.e., representative tasks or prompts for which the model should maintain utility.
- $\ell(y \mid x; \theta)$ is the task loss under parameters θ.
- $\lambda \geq 0$ balances forgetting vs. retention ($\lambda = 0$ ignores the retain term entirely).
- In some methods, y_f is a special unlearning response (e.g., a refusal *"I do not know"* or a masked token), while in gradient-ascent approaches the objective is simply to maximize divergence on D_f rather than hit a fixed y_f [53, 33, 58].

Most unlearning methods rely on first-order optimizers like SGD or Adam, but recent work shows that second-order solvers such as Sophia can achieve stronger knowledge removal with comparable compute budgets [18, 26]. Beyond optimization, selecting the right target response y_f remains challenging: overly strict refusals can harm user experience, while vague paraphrases risk introducing new hallucinations [26].

4.4 Methods for LLM Unlearning

Machine unlearning for LLMs encompasses a diverse set of techniques that either directly modify the model's parameters or steer its behavior at inference time. In general, these are broken down into two families: *(i) model-based methods*, which require access to and updates of internal weights, and *(ii) input-based methods*, which rely purely on prompt engineering without touching the model itself.

Table 4.1 gives a bird's eye view of each approach: summary of access requirements, update mode, computational cost, theoretical guarantees, and typical use cases. The following subsections first detail the various model-based methods and then examine the input-based strategies.

4.4.1 Model-based Methods

4.4.1.1 Gradient Ascent and Variants

Gradient ascent (GA) is among the simplest unlearning techniques for LLMs, adjusting model parameters to maximize the likelihood of incorrect predictions on the forget set D_f [17, 53]. Concretely, given a model parameterized by θ, GA performs updates of the form

$$\theta \leftarrow \theta + \eta \nabla_\theta \mathbb{E}_{x \in D_f} \big[\ell(y_f \mid x; \theta) \big],$$

Table 4.1: Overview of Unlearning Methods for LLMs

Method	**Category**	**Model Access**	**Update Type**	**Compute Cost**	**Guarantee**	**Primary Application**
Gradient Ascent Variants	Model-based	Internal	Parameter	High	Approximate	Privacy & Safety
Localization-informed	Model-based	Internal	Parameter	Medium	Approximate	Privacy & Safety
Influence-Function	Model-based	Internal	Parameter	High	Approximate	Privacy
Parameter-Efficient Fine-Tuning	Model-based	Internal	Parameter	Low	Approximate	Privacy & Safety
RL-Driven Unlearning	Model-based	Internal	Parameter	High	Approximate	Safety
In-Context Unlearning	Input-based	Black-box	Prompt	Low	Approximate	Safety
Prompt Engineering	Input-based	Black-box	Prompt	Low	Approximate	Safety & Copyright
Synthetic-Data Steering	Input-based	Black-box	Prompt	Low	Approximate	Privacy & Safety

where η is the learning rate and ℓ the task loss. In practice, GA is applied for a limited number of epochs to avoid overwriting knowledge outside D_f and thereby preserve utility on the retain set D_r [20].

Naive GA methods can be highly sensitive to hyperparameters, often resulting in unstable forgetting or catastrophic collapse, where the model loses both general knowledge and the specific information intended to be removed [20, 58]. Several variants have been developed to address this:

- *Negative Preference Optimization (NPO)* treats all examples in D_f as *"dislike"* and optimizes a specialized loss to steer the model away from those completions without collapse [58, 40].
- *Relabeled Fine-Tuning* replaces each example in D_f with a neutral or generic response (e.g., a translation) and then applies gradient *descent* to minimize the likelihood of the original targets [53, 54, 11].
- *Incompetent-Teacher Training* uses an auxiliary, randomly initialized *"teacher"* model to generate uniform or random targets for D_f, guiding the primary model to forget by matching those outputs [8].
- *Hybrid GA+Descent* jointly optimizes GA on D_f and standard fine-tuning on a small retain subset $R \subset D \setminus D_f$, balancing forgetting with utility preservation [53].

GA-based unlearning on billion-parameter models often uses parameter-efficient fine-tuning methods such as adapter layers [6] or LoRA task vectors [57]. These ap-

proaches localize weight updates and significantly cut computational costs. Moreover, recent work shows that second-order optimizers like Sophia can further strengthen the unlearning effect without increasing overall compute budgets [18].

GA-based unlearning has also been adapted for safety alignment. For example, the *Selective Knowledge Negation Unlearning (SKU)* framework first uses GA to acquire harmful knowledge. It then applies a negation step to remove it, yielding substantial reductions in harmful outputs while maintaining general utility [28].

4.4.1.2 Localization-informed Unlearning

Localization-informed unlearning targets only the parameters most responsible for the undesired knowledge. By limiting updates to this small subset, it minimizes collateral damage to the model's remaining capabilities. One common strategy is causal tracing, also called representation denoising. Causal tracing systematically disturbs activations in each layer to determine which layers store the unwanted information. By measuring how these perturbations change model outputs, the specific layers responsible for the target content are identified. Fine-tuning is then applied exclusively to those layers to erase the undesired knowledge [34, 37, 26].

Alternatively, gradient-based saliency ranks individual weights by how much their gradients on the forget set exceed a threshold. Fine-tuning these high-saliency weights efficiently removes the unwanted behavior [54, 26]. More fine-grained attribution analysis, such as privacy attribution scores or integrated gradients, can identify the specific neurons or weight blocks most responsible for leaking sensitive data. Those units alone are then updated to *"forget"* the target content [19, 50, 26]. Hence, by localizing updates in this way, these methods achieve effective forgetting with far lower computation overhead than full-model retraining, while preserving unrelated knowledge and overall utility [26].

4.4.1.3 Influence-Function Methods

Influence-function methods approximate how each training example z affects the final parameters θ, and then subtract out that influence to *"unlearn"* the point without full retraining. Formally, the influence of z is estimated as

$$I(z) \;=\; -H_\theta^{-1}\,\nabla_\theta\,\ell(z;\theta),$$

where H_θ is the Hessian of the total training loss [22, 51]. An approximate update $\Delta\theta = -I(z)$ is applied to erase z's effect. Exact inversion of H_θ is infeasible for large models, so practical implementations use Hessian-vector products and truncated conjugate-gradient solvers. Recent work also combines these influence-based updates with second-order optimizers like Sophia to strengthen forgetting while keeping computational costs in check [18].

4.4.1.4 Parameter-Efficient Fine-Tuning

Unlearning at the scale of billion-parameter LLMs can be made practical by restricting weight updates to a small, trainable subset. Parameter-Efficient Fine-Tuning adds small, trainable modules, like adapter layers or low-rank decomposition matrices, into a frozen base model. All unlearning updates are then applied solely to these added parameters, leaving the core model weights untouched. As a result, the computational and memory costs drop by orders of magnitude compared to full-model fine-tuning [6].

Two common approaches are:

- *Adapter Layers*: Small bottleneck networks inserted between Transformer sublayers. During unlearning, only adapter weights are updated, leaving the overwhelming majority of model parameters untouched. This both accelerates fine-tuning and preserves general knowledge housed in the backbone [6].

- *LoRA Task Vectors*: Rank-decomposition matrices added to each attention or feed-forward weight. Unlearning proceeds by modifying only these low-rank components, yielding a compact *"task vector"* that can be negated or removed to forget targeted information [57].

Empirically, parameter-efficient fine-tuning reduces GPU memory use and training time by 5-20x while retaining over 95% of unlearning efficacy compared to dense fine-tuning. Moreover, recent studies show that pairing parameter-efficient fine-tuning with a second-order optimizer like Sophia strengthens forgetting without raising computational costs. This hybrid approach combines the low overhead of lightweight tuning with the fast, stable convergence of curvature-aware methods [18].

4.4.1.5 Reinforcement-Learning-Driven Unlearning

Reinforcement-learning-driven unlearning adapts the Reinforcement Learning from Human Feedback paradigm to penalize undesirable outputs rather than reward helpful ones. First, a reward model is trained (or fine-tuned) to assign low scores to completions in the forget set. Then the base LLM is fine-tuned with a policy-gradient algorithm to maximize this modified reward signal. In effect, the model learns to *"avoid"* the forbidden behaviors, achieving unlearning through preference-based feedback rather than direct gradient adjustments on likelihoods [40, 7].

4.4.2 Input-based Methods

4.4.2.1 In-Context Unlearning

In-context unlearning steers a frozen LLM to *"forget"* specific knowledge purely via carefully crafted prompts and few-shot demonstrations, without any parameter updates. For example, in the paper *Black-Box In-Context Unlearning of Language Models* [38] a small set of *"forget-example"* pairs is inserted before each user query to demonstrate how the model should refuse or safely rephrase forbidden content. This simple in-context steering effectively suppresses unwanted knowledge even when interacting with a black-box API [38, 33].

4.4.2.2 Prompt-Engineering and Instruction Tuning

Rather than touching model weights, prompt engineering crafts carefully designed instructions or *"guardrails"* to steer a frozen LLM away from forbidden content. For instance, one can prepend corrective instructions. As an illustrative example: *"If the user's request involves X"*, respond with *"I am sorry, I cannot help with that."*. In practice, this sharply reduces unsafe or copyrighted outputs [60].

Instruction tuning goes further by fine-tuning the model on a curated dataset of instruction-completion pairs that exemplify safe refusals or safe rewrites. By training the LLM to map *"unsafe"* prompts to neutral or refusal responses, an instruction-tuned model learns to consistently *"forget"* the unwanted behavior even under adversarial queries [47].

4.4.2.3 Synthetic-Data Steering

Synthetic-data steering works at inference time by adding *"fictitious"* examples alongside the user's prompt. This approach, which requires no weight updates, helps weaken any remaining memorization of forbidden content. For example, injecting pairs of distractors crafted that resemble forbidden knowledge can produce harmless completions, effectively breaking misleading associations [33]. Randomly sampled decoy contexts have been shown to block extraction attacks [30], and the addition of benign high entropy *"rescue prompts"* can suppress unwanted memorized outputs in black-box settings [45].

4.4.3 Cross-Cutting Principles

4.4.3.1 Data-Model Interaction

Unlearning at scale requires knowing where in the model the unwanted knowledge lives. As demonstrated in Section 4.4.1.3, influence functions approximate how

upweighting or removing a single training example propagates through, and alters, every model parameter [22, 51]. Beyond influence-function estimators, attribution techniques such as integrated gradients, representational similarity analysis, and causal tracing can localize broader concepts (e.g., toxicity or memorized phrases) to specific neurons, attention heads, or layers. This precise mapping makes it possible to target and remove unwanted knowledge more effectively [34, 14]. Once these *"knowledge loci"* are identified, group-based erasure methods like WAGLE (Weight Attribution for Grouped Learning Erasure) compute a low-dimensional subspace in parameter space that captures a set of related training examples, and then apply a modular update to remove that subspace's influence in one shot [19]. By unifying fine-grained attribution with compact, grouped interventions, this data-model interaction framework enables precise forgetting while preserving the model's remaining capabilities [19].

4.4.3.2 Relation to Model Editing

Model editing performs targeted updates to a model's parameters to inject or update specific behaviors, such as adding a new fact. Unlearning, by contrast, removes or weakens existing unwanted knowledge through focused parameter interventions. Model editors typically identify a small set of parameters to update, using techniques like causal tracing or neuron attribution, and apply a minimal-change update to achieve the desired behavior with minimal collateral effects [35, 34]. Unlearning adapts these same localization tools but flips the optimization objective: instead of reinforcing a target output, it suppresses or *"negatively prefers"* it [26].

Furthermore, group-based editing methods like WAGLE extend single-example edits to entire concept sets by first finding a low-dimensional subspace in the model's parameter space that encodes the unwanted knowledge. Once identified, this subspace is removed in a single update, erasing the concept's influence in one shot [18, 19]. Editing and unlearning act as dual operations: one injects new knowledge, while the other removes it. Adapting practices from the literature, such as interpretability-driven localization and efficient subspace updates, enables more precise unlearning with fewer costly retraining cycles [35, 9, 19].

4.4.3.3 Adversarial Robustness

Unlearning mechanisms are vulnerable to adaptive attackers who craft prompts to extract or *"relearn"* forgotten information. To defend against such threats, unlearning can be framed as a minimax problem. First, adversarial examples are crafted in the model's latent space using projected gradient descent to provoke forbidden outputs. Then the model is fine-tuned to minimize loss on those adversarial inputs. This process reinforces forgetting against worst-case queries [32].

In practice, red-teaming with extraction-style jailbreaks often exposes residual memorization that naive unlearning fails to remove. Incorporating these attacks

into unlearning pipelines, and iterating adversarial training loops, ensures the model maintains its forgetting guarantees even under sophisticated, API-based probing [5, 15, 31].

4.4.3.4 Continual Unlearning

Repeated unlearning can degrade a model's performance on previously mastered tasks, much like catastrophic forgetting in continual learning. To address this, regularization methods such as Elastic Weight Consolidation add a penalty for altering parameters crucial to the model's core competencies. Thus, this approach helps preserve general utility while selectively forgetting new targets [21].

Another strategy casts unlearning as a decision-making problem: a reinforcement learning agent learns when and which examples to unlearn to minimize long-term utility loss. An off-policy reinforcement learning scheduler uses retention performance as its reward signal. Over time, it learns the best sequence of unlearning actions to balance effective forgetting with long-term utility preservation [36, 39]. Periodically replaying a buffer of safe examples stabilizes the model across successive unlearning cycles [39].

4.5 Evaluation Metrics and Benchmarks for LLM Unlearning

Evaluating machine unlearning in LLMs requires more than verifying whether the model forgets specific information. It also demands measuring utility preservation, efficiency, and robustness. This section outlines the standardized evaluation pipeline, commonly used benchmarks, and key criteria that enable a comprehensive and comparable assessment of unlearning methods.

4.5.1 Evaluation Framework

4.5.1.1 Standardized Pipeline

A rigorous LLM unlearning evaluation is carried out in three stages. First, apply the unlearning method to produce an updated model. Second, probe the model with (a) *in-scope* queries targeting the forgotten content and (b) *out-of-scope* queries measuring general capabilities. Third, compute *forgetting efficacy* by measuring how much the unwanted outputs have decreased and assess *utility preservation* through downstream performance metrics such as accuracy or perplexity. Furthermore, report *efficiency* in terms of resource usage, including GPU-hours and peak memory consumption. This three-phase pipeline has become the de facto standard in recent LLM unlearning studies [1, 26].

4.5.1.2 Key Benchmarks

To ensure comparability, most LLM unlearning studies evaluate their methods using three well-established benchmark suites:

Privacy / Extraction: These tasks measure how much sensitive training data remains extractable after unlearning. Membership inference on the Enron email corpus tests whether an attacker can still distinguish *"in-train"* versus *"out-of-train"* examples [50]. The Training Data Extraction Challenge goes further, probing whether literal text strings can be reconstructed by targeted API queries [17]. Success on these benchmarks indicates strong removal of memorized user data [17].

Copyright / Concept: Here the goal is to remove specific copyrighted or proprietary content without harming other facts. The *"Harry Potter"* case study casts the model's ability to recall passages from a well-known novel. Unlearning is successful when the model no longer reproduces those lines [11]. MUSE-style factual sets provide thousands of query-answer pairs (e.g., *"What is X?"*) to systematically verify that only the designated concept has been erased while unrelated world knowledge remains intact [46].

Toxicity / Harm: These benchmarks quantify reductions in unsafe or harmful outputs. Controlled toxicity generation suites such as RealToxicityPrompts measure how often the model produces toxic language when prompted [29]. The TOFU (Toxic Output Frequency Under) benchmarks evaluates a model's refusal rate on socially sensitive queries, while WMDP (Worst-case Model Dangerousness Probing) uses adversarially crafted prompts to stress-test residual harmful behavior [33, 25]. Collectively, these tests verify that unlearning effectively mitigates harmful behaviors while preserving the model's capacity for benign text generation [29, 33].

4.5.2 Forgetting Effectiveness

4.5.2.1 Gold-Standard Comparison

Exact retraining on the retained dataset involves training the model from scratch after excluding all examples in the forget set. This method represents the gold standard for fully removing unwanted information. Full retraining of LLMs is usually infeasible due to its extreme computational cost. Instead, many studies use a surrogate-retrain protocol, such as TOFU's *"surrogate-retrain"*, to approximate the ideal baseline at a fraction of the computational cost. By comparing an approximate method's downstream performance (perplexity, accuracy) and its in-scope forgetting rate against the gold standard, any remaining memorization can be measured. This direct comparison also reveals how much room there is for further improvement [52, 33].

4.5.2.2 In-Scope Hard Tests

In-scope hard tests challenge a model's forgetting by using semantically equivalent or multi-step (multi-hop) prompts and by crafting adversarial queries that explicitly seek the removed content. Success is measured by the model's refusal or incorrect response rates under these tougher conditions, revealing any lasting memorization [37, 31].

4.5.2.3 Membership & Extraction Checks

Membership and extraction attacks probe whether forgotten examples still influence model outputs. Membership inference determines whether a query originates from the training set or not, using the standard Shokri and Shmatikov's protocol [44]. Extraction attacks attempt to reconstruct exact training strings through targeted API queries [48, 20]. Together, these tests quantify any residual memorization after unlearning and the remaining privacy risks on the surface [48, 20].

4.5.3 Utility Preservation

4.5.3.1 Out-of-Scope Task Suite

To evaluate whether machine unlearning methods preserve general model capabilities, researchers use standard natural language understanding and generation (NLU/G) benchmarks as an *"out-of-scope"* task suite. These benchmarks assess performance on tasks that are not related to the unlearning targets, ensuring that the model retains its utility beyond the forget set. Commonly used data sets include GLUE, SuperGLUE, and standard perplexity benchmarks such as WikiText and LAMBADA. Successful unlearning should maintain comparable accuracy or perplexity scores on these tasks, demonstrating that core knowledge and language understanding remain intact despite targeted unlearning interventions [17, 11, 53].

4.5.3.2 Emergent Abilities Trade-Off

LLMs often develop emergent abilities, capabilities that arise only at scale, such as multi-step reasoning or code generation [49]. Machine unlearning methods must balance effective forgetting with the preservation of these high-level behaviors. Recent studies propose charting a Pareto front between unlearning efficacy and retained emergent abilities to visualize and manage this trade-off. A method that forgets too aggressively may compromise complex skills, while overly conservative unlearning may leave residual traces of the target knowledge [26, 49, 53, 20].

4.5.4 Efficiency and Scalability

4.5.4.1 Compute & Memory Footprint

The computational cost of machine unlearning is a critical factor, especially for LLMs with hundreds of billions of parameters. Full retraining to remove even a small forget set can consume millions of GPU hours and incur substantial energy and memory costs [13, 52]. To address this, most modern unlearning methods rely on approximate updates, parameter-efficient fine-tuning (e.g., LoRA or adapter layers), and localized edits to minimize resource usage [6, 18, 26]. Evaluation frameworks typically report wall-clock time, peak GPU memory, and total GPU-hours as key metrics to assess the practicality and scalability of an unlearning technique [1, 26].

4.5.4.2 Scaling with # Forget Examples

As the size of the forget set D_f increases, the performance and efficiency of unlearning methods can degrade. Larger forget sets require more aggressive updates, which may inadvertently harm the general utility of the model or increase the computational cost [26, 52]. To address this, recent work introduces the idea of a *core forget set*, a minimal, representative subset of D_f that achieves similar forgetting efficacy while reducing resource usage and avoiding unnecessary degradation. Identifying such subsets and understanding the scalability of the method are critical to deploying unlearning on a scale in real-world applications [20, 59, 12].

4.5.4.3 Black-Box & API Settings

In real-world deployments, LLMs are often accessed through black-box APIs, where internal parameters are inaccessible. This presents unique challenges for unlearning, as traditional gradient-based methods cannot be applied. To address this, recent work has introduced black-box compatible techniques such as *in-context unlearning*, which uses carefully crafted prompt demonstrations, and *synthetic data steering*, which injects fictitious examples to suppress undesired behavior. These methods are typically evaluated using metrics such as the total number of queries, latency, and refusal rate under adversarial prompting, offering practical benchmarks for unlearning in restricted-access environments [38, 33].

4.6 Application and Case Studies

Machine unlearning is not only a theoretical challenge, but a growing practical necessity. This section highlights key application areas where unlearning plays a crucial role. These include regulatory compliance, copyright enforcement, safety alignment, and real-world API deployments.

4.6.1 Data Privacy Compliance

One of the most pressing real-world applications of machine unlearning in LLMs is data privacy compliance. Regulations such as the GDPR and the California Consumer Privacy Act (CCPA) establish a *"right to be forgotten"*, requiring that user data be erasable from deployed systems upon request [2, 13, 26].

LLMs, trained on massive web-scale corpora, may inadvertently memorize sensitive personal information such as names, emails, or private conversations. This risk is particularly acute in production settings, such as customer support, healthcare, or content generation, where user input can be unintentionally retained [52, 51].

To address this, machine unlearning offers scalable alternatives to full retraining. As discussed in Section 4.4, a range of model-based methods can effectively remove specific user data while preserving general capabilities.

Furthermore, recent studies demonstrate that these methods can reduce the likelihood of memorized output reemergence and pass membership inference and extraction tests designed to detect residual memorization. Thus, machine unlearning forms a critical tool in meeting modern data privacy obligations without sacrificing model performance [26, 50].

4.6.2 Copyright-Protected Content

In the copyright domain, models can memorize protected texts, raising legal concerns. The *"Harry Potter"* case study demonstrated that gradient-based and relabeling-based methods could successfully suppress memorized fictional content while retaining unrelated factual knowledge [11, 53].

4.6.3 Safety Alignment and Toxic Content Removal

Safety alignment is another key application. Techniques such as SKU and in-context guardrails are used to unlearn toxic completions and prevent harmful generations. Such methods have been applied to fine-tuned chatbots and commercial LLM APIs to reduce jail breaking and disallowed behavior [28, 60, 47].

4.6.4 Black-Box and API Settings

Lastly, in black-box and resource-constrained settings, techniques such as in-context unlearning and fictitious data steering enable safe and effective unlearning without requiring access to the internal model. These approaches are particularly relevant for deployed APIs and enterprise SaaS platforms [33, 38].

4.7 Discussion

Despite substantial progress, several foundational challenges remain in the development of robust machine unlearning methods for LLMs.

1. Scope Ambiguity: It remains difficult to precisely define what to forget and how it is encoded. Concept-level knowledge (e.g., copyright text, toxic ideologies) is often distributed across many parameters or attention heads, making targeted removal non-trivial [35, 26, 34]. Efforts such as WAGLE and causal tracing partially address this by localizing knowledge representations [19, 37].

2. Forgetting vs. Utility Trade-off: Strong forgetting can degrade general capabilities such as factual recall or reasoning, especially when unlearning is not well isolated [49, 43, 53]. Methods such as LoRA-based tuning and hybrid GA retention strategies aim to mitigate this, but a fundamental Pareto trade-off persists [26].

3. Adversarial Robustness: Current methods remain vulnerable to extraction and jailbreak attacks, where forgotten knowledge is recovered via adversarial prompts [15, 31]. Incorporating adversarial training or minimax-style optimization is crucial for strengthening forgetting guarantees [32, 5].

4. Black-Box Limitations: In black-box or SaaS contexts, unlearning is further constrained. Approaches such as in-context unlearning and fictitious data steering offer promising workarounds, but suffer from scalability and consistency issues [38, 33].

5. Lack of Standardized Benchmarks: Evaluation remains fragmented. While datasets like TOFU, MUSE, and the Harry Potter case study are widely used, no consensus yet exists on comprehensive benchmarks across all axes: privacy, safety, factuality, and utility [46, 11, 33].

Overall, unlearning is transitioning from a theoretical concern to a practical requirement for privacy, safety, and legal compliance. However, delivering robust, efficient, and interpretable unlearning at scale remains an open research frontier.

4.8 Future Research Directions

As previously discussed, several promising directions could enhance the efficacy and robustness of unlearning in LLMs:

1. **Multi-modal unlearning:** Extending current techniques to vision language and audio text models will require new attribution and editing strategies across modalities [13, 26].
2. **Formalizing unlearning scope:** Better definitions of what constitutes a forgettable unit sentence, concept, behavior are needed to support principled interventions and fairness [24, 35].

3. **Adversarially robust forgetting:** New methods should proactively defend against extraction and relearning attacks via adversarial training or certified removal [32, 15].

4. **Continual unlearning:** Like continual learning, continual forgetting must avoid performance drift as multiple deletions occur over time. Retention buffers, schedulers, and consolidation methods are promising here [21, 39].

5. **Open benchmarking and interpretability tools:** Developing unified evaluation suites and visualization platforms for attribution, retention, and leakage analysis will support reproducible and scalable research [19, 26].

4.9 Conclusion

As LLMs continue to scale and spread through real-world applications, the ability to selectively forget - not just learn - has become an essential requirement for responsible AI development. This report has outlined the conceptual foundations, technical methods, evaluation metrics, and practical challenges of machine unlearning in LLMs. A taxonomy of techniques, ranging from gradient-based updates and parameter-efficient fine-tuning to influence-function methods and in-context interventions, has been presented. These approaches are designed to balance forgetting efficacy with the preservation of model utility.

This report shows that while significant progress has been made, robust and scalable unlearning remains an open research frontier. Key challenges persist in defining unlearning scope, maintaining adversarial robustness, and evaluating effectiveness in black-box and large-scale settings. Nevertheless, the emerging ecosystem of benchmarks, optimization strategies, and interpretability tools provides a strong foundation for further innovation.

Ultimately, machine unlearning is not just a technical tool but a cornerstone of ethical, privacy preserving, and regulation compliant AI. As regulatory frameworks such as GDPR and the AI Act evolve, effective unlearning will become indispensable. The path forward lies in developing principled, efficient and verifiable unlearning methods that enable LLMs to not only learn responsibly but also forget responsibly. Addressing these challenges will be essential for building AI systems that are not only capable but also ethically grounded and legally compliant.

References

[1] A. Arora, S. Gundavarapu, S. Agarwal, et al. “Machine Unlearning in Large Language Models”. preprint. May 2024.

[2] L. Bourtoule et al. “Machine Unlearning”. In: *Proceedings of the 2020 IEEE Symposium on Security and Privacy*. 2020.

[3] Tom B. Brown et al. *Language Models are Few-Shot Learners*. Advances in Neural Information Processing Systems (NeurIPS, 2020.

[4] A. Bucknall and J. Trager. “Machine Unlearning in the API Era”. In: *SysML Workshop on Reliable Machine Learning, 2023*. 2023.

[5] S. Casper, L. Booth, and A. Gupta. “Limitations of LLM-as-a-Service: Unlearning Edition”. In: *NeurIPS Responsible AI Workshop, 2024*. 2024.

[6] X. Chen and Y. Yang. “Parameter-Efficient Fine-Tuning for Machine Unlearning in Large Language Models”. In: *International Conference on Machine Learning, 2023*. 2023.

[7] P. F. Christiano et al. “Deep reinforcement learning from human preferences”. In: *Advances in Neural Information Processing Systems* 30 (2017), pp. 4299–4307.

[8] Vikram S. Chundawat et al. "Can Bad Teaching Induce Forgetting? Unlearning in Deep Networks Using an Incompetent Teacher". In: *Proceedings of the AAAI Conference on Artificial Intelligence, 2023*. 2023.

[9] R. Cohen et al. "Evaluating the ripple effects of knowledge editing in language models". In: *Transactions of the Association for Computational Linguistics* 12 (2024), pp. 283–298.

[10] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186.

[11] S. Eldan and E. Russinovich. "Rethinking Model Unlearning: Case Studies on Harry Potter". In: *International Conference on Machine Learning, 2023*. 2023.

[12] R. Fan, M. Guo, and D. Zhou. "Core Forget Sets for Scalable Machine Unlearning". arXiv preprint. 2024. arXiv: 2404.11098.

[13] J. Geng, Q. Li, Y. Wang, et al. "A Comprehensive Survey of Machine Unlearning Techniques for Large Language Models". Feb. 2025. arXiv: 2503.01854.

[14] P. Hase et al. "Does localization inform editing? Surprising differences in causality-based localization vs. knowledge editing in language models". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 17643–17668.

[15] X. Hu, T. Lee, M. Wang, et al. *Relearning Attacks on Unlearned Language Models*. USENIX Security Symposium, 2024.

[16] T. Ishibashi and Y. Shimodaira. "Selective QA Unlearning in Pretrained LLMs". In: *AAAI Conference on Artificial Intelligence, 2023*. 2023.

[17] E. Jang, S. Eldan, N. Papernot, et al. *Unlearning Private Data from Language Models via Gradient Ascent*. Advances in Neural Information Processing Systems, 2022.

[18] J. Jia et al. "Soul: Unlocking the power of second-order optimization for LLM unlearning". arXiv preprint. 2024. arXiv: 2404.18239.

[19] J. Jia et al. *WAGLE. Strategic Weight Attribution for Effective and Modular Unlearning in Large Language Models*. Advances in Neural Information Processing Systems, 2024.

[20] X. Jia, Y. Li, Z. Smith, et al. "Efficient Unlearning for Language Models". In: *NeurIPS Workshop on Privacy-Preserving Machine Learning, 2023*. 2023.

[21] James Kirkpatrick et al. "Overcoming Catastrophic Forgetting in Neural Networks". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS), 2017*. 2017.

[22] P. W. Koh and P. Liang. "Understanding Black-box Predictions via Influence Functions". In: *Proceedings of the 34th International Conference on Machine Learning (ICML)*. 2017, pp. 1885–1894.

[23] S. Kurmanji, A. Patel, and Q. Chen. "Survey of Unlearning Techniques in Deep Learning". In: *Journal of Machine Learning Surveys* (2023).

[24] T. Lee, M. Nguyen, P. Clark, et al. “Ambiguities in Unlearning Scope for Large Language Models”. In: *International Conference on Learning Representations, 2024*. 2024.

[25] S. Li, M. Zhang, and H. Wang. “Factual Concept Unlearning in Large Language Models”. In: *International Conference on Learning Representations, 2024*. 2024.

[26] S. Liu, Y. Yao, J. Jia, et al. “Rethinking Machine Unlearning for Large Language Models”. Dec. 2024. arXiv: `2402.08787`.

[27] Yinhan Liu et al. “RoBERTa: A Robustly Optimized BERT Pretraining Approach”. 2019. arXiv: `1907.11692`.

[28] Z. Liu, G. Dou, Z. Tan, et al. “Towards Safer Large Language Models through Machine Unlearning”. preprint. June 2024.

[29] J. Lu, A. Shrestha, Y. Wang, et al. “Controlling Toxicity in Large Language Models”. In: *International Conference on Machine Learning, 2022*. 2022.

[30] P. Lucki, A. Fernandez, R. Khanna, et al. “Adaptive Jailbreaking of Unlearned Models”. In: *ACM Conference on Computer and Communications Security, 2024*. 2024.

[31] C. Lynch, S. O’Brien, J. Zhang, et al. “Extraction-Based Jailbreaks in Language Models”. In: *IEEE Symposium on Security and Privacy, 2024*. 2024.

[32] A. Madry et al. “Towards Deep Learning Models Resistant to Adversarial Attacks”. In: *International Conference on Learning Representations, 2018*. 2018.

[33] S. Maini, Y. Li, T. Hase, et al. “Fictitious Unlearning for Large Language Models”. arXiv preprint. 2024. arXiv: `2405.01234`.

[34] K. Meng et al. “Locating and Editing Factual Associations in GPT”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 17359–17372.

[35] E. Mitchell et al. “Memory-based model editing at scale”. In: *International Conference on Machine Learning*. 2022, pp. 15817–15831.

[36] Long Ouyang et al. “Training Language Models to Follow Instructions with Human Feedback”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 27730–27744.

[37] V. Patil, P. Hase, and M. Bansal. “Can sensitive information be deleted from LLMs? Objectives for defending against extraction attacks”. arXiv preprint. 2023. arXiv: `2309.17410`.

[38] S. Pawelczyk, R. Kandpal, M. Sarani, et al. *Black-Box In-Context Unlearning of Language Models*. Association for Computational Linguistics, 2023.

[39] Ming Qi et al. “Adaptive Scheduling for Continual Unlearning in Neural Models”. In: *Proceedings of the AAAI Conference on Artificial Intelligence, 2023*. 2023.

[40] R. Rafailov et al. “Direct Preference Optimization: Your Language Model Is Secretly a Reward Model”. In: *Proceedings of the 37th Conference on Neural Information Processing Systems, 2023*. 2023.

[41] Colin Raffel et al. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. In: *Journal of Machine Learning Research* 21.140 (2020), pp. 1–67.

[42] Google Research. *Announcing the first Machine Unlearning Challenge. Google Research Blog*. 2025. URL: %5Curl%7Bhttps://research.google/blog/announcing-the-first-machine-unlearning-challenge/%7D (visited on 05/11/2025).

[43] D. Schaeffer, Y. Chen, K. Patel, et al. *Revisiting Emergent Abilities of LLMs*. Advances in Neural Information Processing Systems, 2024.

[44] Reza Shokri and Vitaly Shmatikov. "Membership Inference Attacks Against Machine Learning Models". In: *Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP)*. 2017.

[45] P. Shumailov, S. Huang, T. King, et al. "Rescue Prompts for Unlearned Models". In: *International Conference on Learning Representations, 2024*. 2024.

[46] Nianwen Si et al. "Knowledge unlearning for LLMs: Tasks, methods, and challenges". arXiv preprint. 2023. arXiv: 2311.15766. URL: %5Curl%7Bhttps://arxiv.org/abs/2311.15766%7D.

[47] A. Thaker, M. Gupta, and R. Rao. "Jailbreak Deterrence via Instruction Tuning". In: *Proceedings of the 2024 Annual Meeting of the Association for Computational Linguistics*. 2024, pp. 2105–2116.

[48] Shrinidhi Thudi, Anuj Kumar, and Priya Banerjee. "Data Forging Attacks to Extract Training Data from Language Models". In: *AAAI Workshop on Privacy & Security of Machine Learning, 2022*. 2022.

[49] J. Wei, X. Dong, P. Suarez, et al. *Emergent Abilities of Large Language Models*. Advances in Neural Information Processing Systems, 2022.

[50] T. Wu, X. Yang, J. Xiao, et al. "Importance-Based Neuron Editing for Machine Unlearning in LLMs". In: *ACL Workshop on Security and Privacy, 2023*. 2023.

[51] J. Xu et al. *Machine Unlearning. Solutions and Challenges*. IEEE Transactions on Emerging Topics in Computational Intelligence, 2024.

[52] J. Yao, E. Chien, M. Du, et al. "Machine Unlearning of Pre-trained Large Language Models". May 2024. arXiv: 2402.15159.

[53] J. Yao, E. Chien, X. Niu, et al. "Machine Unlearning for Reducing Harmful Hallucinations in LLMs". In: *AAAI Conference on Artificial Intelligence, 2023*. 2023.

[54] Y. Yu, H. Zhang, and N. Cohen. *Bias Unlearning in Language Models via Weight Importance*. Advances in Neural Information Processing Systems, 2023.

[55] H. Zhang, Y. Min, and J. Doe. "Memory-Constrained Unlearning for LLMs". In: *Conference on Empirical Methods in Natural Language Processing, 2024*. 2024.

[56] H. Zhang et al. "A Review on Machine Unlearning". 2021. arXiv: 2111.11315.

[57] W. Zhang, J. Li, and A. Smith. "Task-Specific Knowledge Unlearning for Conversational Agents". In: *Conference on Empirical Methods in Natural Language Processing, 2023*. 2023.

[58] X. Zhang, Y. Liu, and P. Yang. *Negative Preference Optimization for Unlearning*. Association for Computational Linguistics, 2024.

[59] L. Zhao, Y. Yang, H. Liu, et al. "Unlearning at Scale: Efficient Concept Erasure in LLMs". arXiv preprint. 2024. arXiv: 2404.11875.

[60] S. Zheng, L. Wang, and T. Liu. “PromptShield: Instruction-Based Safe Generation for Large Language Models”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 2023, pp. 1234–1245.

Chapter 5

Overview of Crypthographic Techniques in Blockchain Applications

Florian Mattmueller, Jana Joy Anja Muheim

This report looks at how cryptographic techniques are use in blockchain applications. It explains how techniques like hash functions, asymmetric cryptography, and digital signatures help keep blockchain data secure. The report also shows how blockchains deal with challenges such as slow transactions and keeping data private, by using tools like zero-knowledge proof and layer-2 solutions. The report also discusses the risks that quantum computers could bring to blockchain applications and some possible solutions to that. Overall, it gives an overview of how cryptography helps making blockchain work and what challenges may come next.

Contents

5.1 Introduction 101
5.1.1 Permissionless vs Permissioned Blockchains 101
5.1.2 Key Blockchain Components 102
5.1.3 Consensus Models 104
5.1.4 Soft vs Hard Forks 104
5.2 Blockchain Techniques 104
5.2.1 Asymmetric Cryptography (ECC, RSA) 104
5.2.2 Hashing 105
5.2.3 Merkle Trees 107
5.2.4 Layer-2 Solutions 108
5.2.5 State Channels 108
5.3 Blockchain Applications 109
5.3.1 Data Sharing and Multi-Party Computation 109
5.3.2 Internet of Things and PUF 109
5.3.3 Self-Sovereign Identity 110
5.3.3.1 Cryptographic Accumulator 111
5.3.3.2 Zero-Knowledge Proofs (ZKP) 112
5.4 Post-Quantum Security 113
5.4.1 Security Level 113
5.4.2 Post-Quantum Security Level 115
5.4.3 Post-Quantum resistant Encryption and Blockchains . . 116
5.5 Related Work 118
5.6 Conclusion 119

5.1 Introduction

Blockchain was first revealed in 2008 by Satoshi Nakamota in his paper [20]. Blockchain was developed to addressed the problem of trusting distributed systems. "More specifically, the problem of creating a distributed storage of timestamped documents where no party can tamper with the content of the data or the timestamps without detection" [5].

In traditional financial transaction or similar a single trusted ledger, like the bank, is needed so they verify the transaction. The problem with single trusted centralized ledger is that it doesn't scale to large numbers and it needs all involved parties to trust the ledger's maintainer. So when doing a financial transaction through a bank you need to trust your bank with your money and for example that the bank employees don't steal your funds. One alternative to solve this issue is by using a blockchain technology. It uses a decentralized trust system where each participant maintains a record of transactions. Each participant then can independently verify that the order and timestamp of the transactions remain intact and have not been altered [5].

This decentralized ledger records each transaction in a block. Each block contains three key elements: a timestamp, the transaction details, such as who sent money to whom, and a hash of this block, its details and the hash of the previous transaction. Every block is connected to the one before it, forming a chain. This makes it nearly impossible to change past transactions without altering the entire chain.

Blockchain nowadays is applied in a wide range of domains including financial transactions, real estate, smart contracts or patent protection to later prove a first-to-invent claim. While cryptocurrencies remain the best-known application of the blockchain, many companies are exploring new ways to use blockchain for security, automation or keeping record of various things[5]. [5] states that "because a transaction is basically a string, it can contain arbitrary information" it is clear to see that this can contain any kind of notarization and can therefore be used not only when money is involved.

5.1.1 Permissionless vs Permissioned Blockchains

Modern blockchain networks are classified based on their permission model. This permission model determines who can maintain and add new blocks to the blockchain. There are two main categories: permissionless and permissioned blockchains [39].

Permissionless blockchains are decentralized ledger platforms that allow anyone to publish a new block without the need of having the permission from any authority. Famous examples of permissionless blockchains include Bitcoin and Ethereum, where anyone can join the network, validate transactions, and contribute to the blockchain's security. These platforms are often built on open-source software, allowing anyone to download and use them freely. Since anyone can publish blocks

it also means that anyone can read and write transactions on the blockchain. But because this unrestricted participation is allowed there would be a lot of potential for getting attacked by malicious users. To prevent this, these networks use a consensus mechanism, which required participants to either expend or maintain resources before publishing blocks. Popular consensus models include proof of work (PoW) and proof of stake (PoS). Consensus models usually promote honest behavior by rewarding participants with cryptocurrency for following network rules [39].

Unlike permissionless blockchains, permissioned networks can restrict who has access to read or issue new blocks. Users get authorized by a either centralized or decentralized authority. Both blockchain categories share some key features such as traceability of digital assets and decentralized data storage. However, permissioned systems do not require the same consensus mechanisms, since their participants have to be authorized they are already trusted. Instead of PoW or PoS, permissioned blockchains use consensus models that are usually faster and less computationally expensive [39].

5.1.2 Key Blockchain Components

Blockchain technology is composed of several key components that work together to ensure security and integrity. In general terms, blockchain technology brings together well-known concepts from computer science and cryptography along with ideas such as append only legders [39].

Cryptographic Hash Functions

In blockchain, cryptographic hash functions are used to ensure the integrity of each transaction, each block, the whole blockchain, to derive addresses, and to speed up digital signatures that are used in every transaction. They take an input of an arbitrary size and return an output of fixed size. For a hash function to be a cryptographic hash function, it needs to satisfy three properties called preimage resistance, second preimage resistance and collision resistance [37].

Asymmetric-Key Cryptography

Asymmetric-key cryptography, also known as public-key cryptography, is used in blockchain networks for identity verification, digital signatures, and transaction authentication. It uses a pair of keys that are mathematically related to each other: one public key and one private key [39]. The public key is shared openly. The private key, on the other hand, is kept secret by the owner. The private key is used to encrypt a transaction where before anyone with the according public key can decrypt it. This signing with the private key then proves that the signer of the transaction is in possession of the private key. This approach enables participants to confirm the authenticity of transactions without relying on a central authority [39].

But with asymmetric-key cryptography the problem of storing the private key securely arises. Keys are usually stored in so called wallets. Wallets can store private keys, public keys, and also associated addresses. In the case that a user loses a private key, than any digital assets associated with that key is lost, since it is impossible to regenerate the same private key [39].

Transactions

The transaction represents the transfer of the cryptocurrency or information between participants in a blockchain network. Each transaction is digitally signed by the sender using asymmetric-key cryptography to ensure authenticity [39].

According to [39] transactions typically include:

- Inputs: References to previous transactions that the sender is using as the source of digital assets.
- Outputs: Details of the recipient and the amount of digital assets to be transferred.
- Signatures: Digital signatures generated by using the sender's private key to verify transaction authenticity and validity.

Ledgers

A ledger keeps track of the transactions. They are usually digitally stored in a database, "owned and operated by a centralized trusted third party". In blockchain technology the ledger is distributed. This approach uses a much larger set of computers than it does for a centrally managed ledger but is very attractive to users due to trusting and security reasons. Blockchain ledgers can, unlike centralized ledgers, ensure immutability, transparency and redundancy by having the ledger replicated across multiple nodes [39].

Blocks

In blockchain technology the transactions get stored in blocks. In many blockchain implementations each block consists of two main components: block header and block data [39].

The block header contains data such as the block number, timestamp, the previous block header's hash, and a cryptographic hash of the block's data, size of the block and the nonce value. The nonce value is a number that is manipulated by the publishing node to solve the hash puzzle [39].

The block data contains a list of validated transactions included in the block and sometimes other data. Blocks are connected using cryptographic hashes, forming the blockchain. Each block contains the hash of the previous block's header, creating an unchangeable chain. If any previously published block is altered, that hash value of that block would also change, causing all subsequent blocks to become invalid. This provides a secure and transparent digital ledger system, since altered blocks get easily detected [39].

5.1.3 Consensus Models

Through various consensus models, agreements on which user publishes the next block can be reached by a group of users [39].

When a new blockchain network is created, all users agree to the system's initial state based on the agreed-upon consensus model, which is recorded in the genesis block. Every later added block must be added in accordance with the network's consensus model. The integrity of the blockchain is ensured by the genesis block agreement, the chosen consensus model and the independent user verification. This allows users to independently verify the blockchain's integrity, eliminating the need for a trusted third party to do so [39].

5.1.4 Soft vs Hard Forks

Modifications to a blockchain technology are known as forks, which can be categorized into soft forks and hard forks. Implementing changes can be very complex, especially to permissionless blockchains where nodes are distributed and governed by user consensus [39].

A **soft fork** is a change that is backwards compatible, allowing non-updated to continue transacting with updated ones. In contrast **hard forks** are non-backward-compatible. If some nodes continue using the old protocol, the blockchain splits into two independent versions [39].

5.2 Blockchain Techniques

5.2.1 Asymmetric Cryptography (ECC, RSA)

Rivest, Shamir and Adleman - Algorithm (RSA)

RSA is one of the most known and used asymmetric cryptography algorithm. It stands for Rivest, Shamir and Adleman, which are the names of the inventors of this algorithm. The RSA algorithm relies on the fact that, while it is easy to generate and multiply large prime numbers, reversing the process, so factoring the resulting product back into its original primes, is extremely difficult. Once the public key is shared with someone, that person can encrypt a message they want to send using the receiver's public key. The receiver than can encrypt that message using his private key, since the public and the private key are mathematically related [8].

In RSA, the bigger the key size is the better and stronger is the security. This implies more overhead on computing systems. Especially nowadays where there

are more and more small devices in the digital world that have less memory this becomes an issue [19].

Elliptic Curve Cryptography (ECC)

ECC stands for Elliptic Curve Cryptography. "Elliptic curve-based systems can be implemented with much smaller parameter, leading to significant performance advantages when comparing to RSA systems" [16].

ECC relies on an elliptic curve for encryption and decryption. This curve can be written as $y^2 = x^3 + ax + b$ [16].

ECC is based on scalar multiplication, much like exponentiation in RSA. Scalar multiplication involves multiplying a point on an elliptic curve by a scalar, or integer. All points on an elliptic curve are over a finite field and form a mathematical structure with a defined addition rule. This means that when two points on the curve are added together, the result is found by drawing a line through the two points, and finding the third point that intersects with the curve. This third point then needs to be reflected over the x-axis. This process makes ECC suitable for cryptographic algorithms that rely on hard mathematical problems for security [16].

5.2.2 Hashing

Cryptographic hash functions are an important component of blockchain technology. A hash function is any function that takes an input of arbitrary size and generates an output, which is referred to as a hash or message digest, of fixed size. If this function satisfies the following additional three requirements, then it is considered a cryptographic hash function [37].

- **Preimage resistance:** given the hash H(x), it is computationally infeasible to find input message x. This is also known as a one-way function.
- **Second preimage resistance:** given the hash H(x) and its input message x, it is computationally infeasible to find another input message x' such that H(x) = H(x') with x $\neq$ x'.
- **Collision resistance:** it is computationally infeasible to find a pair of input messages x, x' with x $\neq$ x' such that H(x) = H(x')

Note that collision resistance and second preimage resistance are not the same properties even though they seem similar at first glance. For collision resistance, an attacker can choose x and x' whereas for second preimage resistance, an attacker can only choose x' and is given x. In a brute force attack on collision resistance of a n-bit long hash, the attacker requires effort equal to $2^{n/2}$. However, in a brute force attack on second preimage resistance, the required effort is equal to 2^n. This surprisingly big difference in these two seemingly very similar scenarios, is known as

the birthday paradox. Furthermore, collision resistance implies second preimage resistance but does not imply preimage resistance and preimage resistance and second preimage resistance do not imply each other. takes these three properties and divides them further into seven security properties. Others have come up with additional criteria, such as the avalanche criterion or near-collision resistance [37].

Cryptographic hash functions have many applications throughout cryptography, such as in authentication, efficient digital signatures, pseudo-random number generation, session key generation, detection of duplicate data, creation of unique identifiers and checksums. In blockchain networks, cryptographic hash functions are used for many tasks, including[37]:

- Address derivation
- Creating unique identifiers
- Securing the block data, whose digest is stored within the block header
- Securing the block header

Many hash functions like SHA-1, SHA-256 and MD5 are based on an iterative structure. One such iterative structure called the Merkle-Damgard construct uses a compression function, that is proven to satisfy the three properties needed for a cryptographic hash function. Then it builds a cryptographic hash function that also satisfies the three properties using the compression function[37]. The Merkle-Damgard construct achieves this in four steps, which are shown below using the implementation in MD4 [33] as an example:

1. Append padding bits

First append a bit of value '1' to the end of the input of arbitrary length. Then add bits of value '0' until the padded message is congruent to 488 modulo 512, meaning it is 64-bits smaller than a multiple of 512-bits.

2. Append length and cut up message

Append a 64-bit representation of the length of the initial input to the padded message. Appending a representation of the length is also known as MD-strengthening and makes the hash function more resistant to collisions [37]. The message is now a multiple of 512-bits which is cut up into 16 words of 32-bit size ($16 * 32 - bits = 512 - bits$) for each block of 512-bits.

3. Initialize internal state

The compression function has an internal state of four words, each 32-bits. This internal state of in total 128-bits will also serve as the message digest at the end of the fourth step and is initialized only once as the following four hexadecimal words in little-endian:

word A: 01 23 45 67
word B: 89 ab cd ef
word C: fe dc ba 98
word D: 76 54 32 10

4. Run 16 words through compression function

For each of the 16 words in each 512-bit block, the compression function changes/mixes up its internal state using that word during three rounds. After all the words went through three rounds to mix up the internal state, the last value of the combination of the internal state (A,B,C,D) becomes the message digest.

MD4 accepts messages of any size and returns hashes of a fixed size, which satisfy the basic properties of a hash function. However, it does allow for specific attacks on its compression function, which makes it a suboptimal cryptographic hash function. There are other iterative constructs used for hash functions, such as wide pipe iterated hash design, fast wide pipe design, hash iterated framework (HAIFA) or Sponge Construction which try to improve on the Merkle-Damgard construct and its hash functions [37].

5.2.3 Merkle Trees

Bitcoin can have up to 4000 transactions in the body of one block [10] but only stores one hash in the header and yet if a light-weight client wants to check the integrity of his transaction, he does not have to recalculate all the 4000 hashes. Bitcoin uses a data structure called merkle trees for efficient checking of integrity of a block or a transaction [20].

A merkle tree is a binary tree in which the leaf nodes are individual transactions, and the inner nodes are hashes of their child nodes. The node at the top of the tree is called merkle root [34] and is the only hash that is placed into the header of the block to represent all the transaction of that block [20].

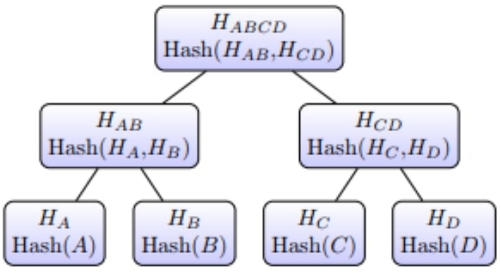


Figure 5.1: Merkle Tree [34].

A merkle proof is a recreation of the merkle root to prove that the hash of a transaction was used in its creation. In Figure 2.1 a light-weight client would only need to access H(A), H(B) and H(CD) to prove that H(A) was used in the creation

of the merkle root. If the recreated merkle root differs from the merkle root on the blockchain, the light-weigh client knows that a different transaction was used in the block. Merkle trees are more efficient than storing a hash of a concatenation of all the transaction in the header, because the light-weight client would have to access all of the transactions [34].

5.2.4 Layer-2 Solutions

Blockchain's main problem is the scalability. Since it has gotten more and more attention through cryptocurrency, more and more people want to use it. The problem here is that the blockchain technology is decentralized, which makes it difficult to scale because transactions have to be broadcasted to the whole network. So the more users there are the more people need to agree on a new block being published. This results in longer periods between block confirmation, potentially slowing down the transaction processes [35].

"The most common approach to achieve a scalable blockchain is generally known as "Layer 2"." The core idea behind Layer 2 is to create a framework that processes transactions off-chain - so separate from the main blockchain. This reduces the on-chain load and can significantly increase the transaction speed. Essentially, a Layer-2 solution is a secondary protocol built on top of an existing blockchain. The fundamental principle here is that transactions are processed off-chain, and only a summarized record of these transactions is reported back to the main chain [35].

To this day, all known solutions have to deal with the **scalability trilemma.** The scalability trilemma says that the scalability, security and decentralization are connected. So every improvement on the scalability of a blockchain has a negative effect either on security or decentralization or both [35].

5.2.5 State Channels

One current in use application of channels is the Bitcoin Lightning Network. This system uses off-chain transaction channels to enable direct transfers between users without the need to store every transaction on the main chain which would end in high transaction fees. Those so called Lightning channels between two participants are typically initiated by committing a single on-chain transaction that locks a predefined amount of currency. Once the channel is established, the participants can repeatedly update the balance distribution between them by exchanging off-chain transactions. These updates reflect the current allocation of the locked funds but are not broadcast to the blockchain unless the channel is being closed, when then the final agreed-upon transaction is submitted to the blockchain [21].

This off-chain structure not only reduces transaction fees and confirmation delays, but also ensures that if one party attempts to cheat by for example broadcasting an

outdated transaction or becomes unresponsive, the other party can recover their funds or, in some cases, even punish the other party by taking the whole amount of funds in the channel [21].

5.3 Blockchain Applications

5.3.1 Data Sharing and Multi-Party Computation

Nowadays a single company can have an enormous amount of data, however it might still have to pool its data with other companies for some of the more data-intensive goals like AI or statistics. Using cloud service providers is an efficient solution for data sharing but it raises concerns about trust, centralization, data sovereignty and data privacy [22]. A blockchain based, decentralized computation platform like Enigma [41] offers decentralized storage of encrypted data on a second chain that links to a main chain. Using multi-party computation (MPC) users can benefit from joint computation on a larger pool of data while still preserving privacy of their data [41]. Enigma eliminates the lack of privacy in blockchains and has no reliance on third party trust but offers less performance than centralized computation and data sharing [41].

Multi-party computation is based on secret sharing [41] and we will take a closer look at Shamir's secret sharing [36] as an example. We first choose a polynomial of t degrees

$$p(x) = a_0 + a_1 x + \cdots + a_t x^t$$

where $a_0 = s = p(0)$ is the private data (s) we want to compute on

From the polynomial function p(x) we extract shards which are points (i, p(i)) on p(x) for $i \in \mathbb{N}$. Note that one shard or point does not contain enough information to reconstruct the secret p(0). We can then distribute n shards to n nodes in the blockchain which run a agreed upon computation on their shard and return their result. The results are combined to receive the result of the computation on the secret. Note that the result using MPC is equivalent to the result we would have gotten if the chosen computation was executed directly on the secret. MPC can be extended from arithmetic computation to turing-completeness. To make the result of the multi-party computation verifiable, it can be combined with publicly verifiable SPDZ protocol which adds proofs of correctness that are stored on the blockchain [41].

5.3.2 Internet of Things and PUF

The Internet of Things (IoT) can be combined with blockchain as has been attempted in [29], [18] and [13]. One particular challenge is data provenance and

integrity of IoT devices which operate with low energy and are physically exposed and vulnerable to side-channels attacks by nature [13], [1]. Data provenance concerns itself with the correct origin of data (e.g. data from a device that has been replaced with a malicious device). One possible solution, that has been implemented in BlockPro [13], uses physical unclonable functions (PUFs) to allow for secure authentication of IoT devices to a blockchain platform. PUFs are cheap, low energy and sometimes take no additional manufacturing at all which makes them very appropriate for IoT devices [1].

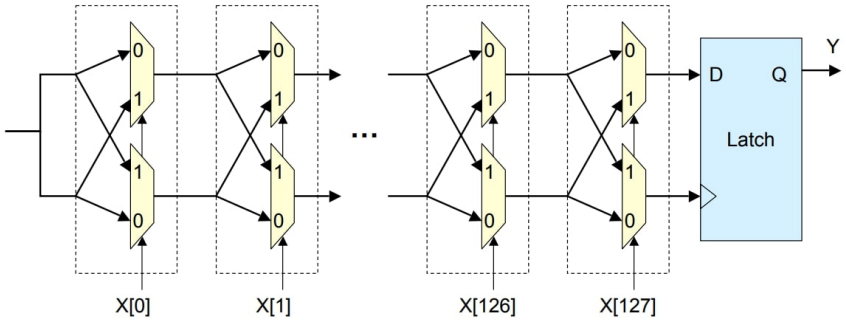


Figure 5.2: Arbiter PUF delay circuit[38]

PUFs utilize built-in, unique imperfections in the complex micro-structure of the circuit in an IoT device to generate a Response from a received Challenge [1]. There exist different types of PUFs, such as arbiter PUFs, Ring oscillator PUFs, optical or acoustic PUFs [38]. An arbiter PUF such as the one in Figure 2.2 consists of a delay circuit, through which two signals run, starting from the left and ending on the latch. Whichever signal gets to the latch first decides the output the latch creates (e.g.: top signal first creates an output of 1 and bottom signal first creates an output of 0). The path that each signal takes through the delay circuit is dependent on the challenge. Each challenge bit inputs to a MUX, which decides the path through the circuit [38]. Even if an attacker knew the exact schematics of the PUF circuit, it would be very hard or even impossible to recreate the same structural imperfections which end up influencing the generation of the response. This makes the circuit physically unclonable and secure [1].

To intigrate an IoT device equipped with a PUF into a IoT blockchain network, the device would first generate a bunch of challenge response pairs (CRP) and hand them to a node in the blockchain system [13]. When the IoT device wants to identify itself, it requests a challenge from the node, then it generates the response using its PUF circuit and sends back the CRP for authentication. The node can then check the received CRP with the CRP it received when the IoT device was first set up to validate the identity of the device [1].

5.3.3 Self-Sovereign Identity

Digital identities are an essential part of modern life, enabling secure access to online services and systems. A digital identity represents selected attributes of a

real-world entity (such as a person or organization) for a specific purpose. These identities vary in the level of detail and accuracy. While each entity has a single "real-life" identity, it can have numerous digital identities [4].

Existing identity systems are often managed by centralized organizations, creating challenges such as privacy concerns and user dependence on third-party institutions [4]. This is where self-sovereign identities (SSI) offer an alternative. Instead of relying on centralized authorities, self-sovereign identities give individuals full control over their digital identities. The individual itself can then administrate its identity and decide who can access which data and for what purpose. Ultimately, self-sovereign identity represents a shift from organization-controlled identity systems to individual- or entity-controlled frameworks, emphasizing autonomy, privacy, and trust in the digital age. With the help of blockchain such self-sovereign identity management systems are very feasible [4].

The following properties make blockchain a great candidate for it:

- Distributed consensus: Blockchain allows participants to agree on the current state of the ledger without needing a central authority. This makes it possible to build systems where every change or transaction is visible and can be verified by authorized users and therefore supports decentralization and trust without a third-party, two key SSI principles [6].
- Immutability: Once an identity credential is issued and recorded on the blockchain, it can't be tampered with. This ensures that the identity data is trustworthy and not altered [6].
- Data origin: Every credential or identity transaction is cryptographically signed by its issuer, ensuring authenticity. This helps verifiers check who issued a credential and whether it's trustworthy [6].
- Decentralized data control: Blockchain stores and shares data in a way that avoids that any single entity can control the system. This ensures user sovereignty and independent data control [6].
- Accountability and Transparency: Blockchains allow any participant to see who did what and when. In an SSI context, this creates verifiability and accountability, both for credential issuers and verifier. It also builds trust in the system [6].

5.3.3.1 Cryptographic Accumulator

In self-sovereign identities one can keep their data more private when being able for other people to check whether a certain item is part of a list or not. This is what an accumulator is able to do. "A cryptographic accumulator is a space- and time-efficient data structure used for set-membership tests" [30]. Accumulators allow to encapsulate a set of elements, allowing a prover to create a membership

prove for each element that is part of the accumulated set. The main function of a cryptographic accumulator comes from its ability to turn any computational problem into a set membership problem [30].

Cryptographic accumulators can be classified based on several propertries:

- Security: soundness, completeness, undeniability and indistinguishability are the four key security properties [30].
- Types of accumulators: A dynamic cryptographic accumulator is able to allow updates to the input set whereas in a static cryptographic accumulator the input set remains unchanged [30].

5.3.3.2 Zero-Knowledge Proofs (ZKP)

In self-sovereign identities, cryptographic accumulators as well as zero-knowledge proofs play a critical role for providing a privacy-preserving mechanism [30].

Zero-knowledge proof is a two-party protocol in which a prover convinces a verifier that a given statement is true without revealing any other information. Early constructions of ZKPs were applied to signature schemes and securing ciphertext. Later work showed ZKP's broad utility such as in cryptography [3].

According to [3] any ZKP system must satisfy three properties:

- Completeness: If the statement is true and both parties follow the protocol honestly, the verifier will always accept the proof.
- Soundness: If the statement is false, the verifier can't be convinced to accept it.
- Zero-Knowledge: Beyond the fact that the statement is true nothing else will learn the verifier; so no additional information is leaked.

In combination with accumulators, ZKPs can be utilized to validate membership proofs in a secure manner. When an accumulator can support ZPKs, the prover can demonstrate that an element belongs to the accumulated set using a witness without exposing the actual content of the accumulator [30]. With integrating ZKPs there are several advantages:

- Privacy preservation: ZPKs ensure that sensitive information remains private while still allowing verification of membership [30].
- Indistinguishability: a well-designed accumulator that has ZKPs integrated achieves indistinguishability, meaning that no information about the accumulated set can be derived from either the accumulator itself or the belonging witness [30].

- Minimal interaction requirements: Traditional ZKP systems, also called interactive ZKPs, require multiple rounds of interaction between the prover and the verifier requiring more communication overhead and making it harder to integrate into large systems such as a blockchain. Therefore in combination with blockchain, mostly non-interactive ZKPs are used where the prover generates only one proof that the verifier then can check [30].

Despite their benefits, incorporating ZKPs into an accumulator's architecture come with quite some challenges. Especially since ZKP implementations often rely on a trusted setup which can introduce centralization risks. But overall, the integration of ZKPs into cryptographic accumulators elevates their performance and security, allowing self-sovereign identity applications based on blockchain to leverage the advantages of both membership proof capabilities and stringent privacy requirements [30].

5.4 Post-Quantum Security

5.4.1 Security Level

To understand the impact quantum computing has on encryption used in blockchain and on blockchain itself, we must first introduce a way of capturing the amount of security a given encryption algorithm provides, so that we can then clearly and more easily see how post-quantum attacks influence the security of encryptions and blockchain systems. We use the notion of security levels to measure the security of encryption and the notion of a general attack as both described in [17]. describes Security level as a measure of how much effort is needed to break an encryption given the fastest general attack and it describes a general attack as an attack that does not have any additional information about the secret (e.g. private key) used for encryption, like a side-channel attack or social engineering. So, the attack only uses basic information such as the ciphertext and a public key or a plaintext and its ciphertext. A general attack might have to search the whole search space of a private key exhaustively or it might only have to search for part of the search space due to an exploited imperfection or mistake in the encryption algorithm. If a symmetric encryption has a private key length of n bits and the fastest attack available is a brute force attack on the whole search space, then it has a n-bit security level. This is because the attacker must search through 2^n possible private keys and thus undertakes an effort proportional to 2^n computations. For example, the symmetric encryption AES-128 has a security level of 128 as the attacker must search through 2^{128} different private key combinations [17].

From [17] we also learn that security level is not an absolute measurement of security rather it needs to be interpreted correctly and relative to other important aspects. First of all, n-bit security level emits a constant of proportionality that would be found in the effort $\alpha * 2^n$ needed to break the encryption. Meaning security level does not take into account how fast the brute force attack can check

its guesses using the encryption. If one symmetric encryption with a security level of 128 runs 100 times faster than another symmetric encryption with the same security level, the first encryption is 100 times easier or faster to break. Security level also does not consider how much computational power is available at a given time in history or how sophisticated and motivated the attackers are. Every year attackers have greater resources available than they had last year. This is due to Moore's Law which claims that every 12 to 24 months the cost of computational power halves, which allows for double the amount of computational power for the same amount of money. Similarly, cryptographic progress is also not taken into account in security level. In this context cryptographic progress lowers the security level of a given encryption because new understandings of encryption leads to more optimized attacks or new, stronger attacks. Cryptographic progress is fundamentally different from improvements in available computational power in two ways. It only affects one encryption or one type thereof, while Moore's Law enables any kind of attack on encryption to run faster. Furthermore, cryptographic progress is unpredictable while improvements in computational power are more predictable. As a consequence of Moore's law and cryptographic progress, the actual security provided by a given security level, diminishes with every passing year. Additionally, a given security level itself does not claim to be adequately secure or not [17]. Whether or not it offers enough security depends on the application and field it is used in, and is ultimately decided by the acting bodies in it. For example, a company that works with personal data over a long period of time might need a higher security level in its encryption than another company that handles less important and more temporary data.

In essence, the security level of an encryption is only an estimate of security proportional to the effort required to break it using the fastest general attack, relative to the available computational power and cryptographic knowledge. This estimate of security is also subject to individual perspective of what it means to be 'secure' [17].

Table 5.1: Relation of key/hash length to security level [17]

name	type	key/hash length	security level
DES	symmetric encryption	56	56
AES-128	symmetric encryption	128	128
RIPEMD-160	hash function	160	80
SHA-256	hash function	256	128
RSA-1024	asymmetric encryption	1024	72
RSA-4096	asymmetric encryption	4096	125

[17] also looked at the relation between different types of encryptions and their security level. Symmetric encryption, asymmetric encryption and hash functions each have a different relation between their key length/hash length and their security level. For symmetric encryption, more precisely for block ciphers, the security level is typically equal to the private key length. So, an n-bit private key operates with n-bit security level because an attacker must consider the whole 2^n search

space. For example, AES-256 has a security level of 256. For Hash functions with a n-bit hash, the security level is typically $n/2$ because an attack on collision resistance takes $2^{n/2}$ effort. For these two types of encryptions, key/hash length grows linearly with security level. This is not the case for asymmetric encryption where key lengths grow much faster with increasing security level as shown in table 2.1. [17] points out that this makes standards for asymmetric encryption more controversial and less universal than standards for hashing and symmetric encryption. The performance of hash functions and symmetric encryption does not deteriorate noticeably with increasing security level and hash/key length because the relation is linear. This makes it easy to propose and follow conservative, meaning secure standards. In asymmetric encryption, performance does deteriorate noticeably with higher security level because the key length increases faster, which makes the balance between security and performance more important and trickier [17].

5.4.2 Post-Quantum Security Level

In this report post-quantum security level is given the same definition as the classical security level above and it should also be interpreted the same way. The only difference is that post-quantum security level considers the fastest attacks using quantum computers. This expands the group of possible attacks considered which lowers the post-quantum security level compared to the classical security level of encryption.

For a post-quantum attack to efficiently run on a quantum computer, one also needs an underlying algorithm that can properly utilize the power of a quantum computer. Perhaps the two most famous and, in the case of Blockchain, most influential algorithms are Shor's algorithm and Grover's algorithm. [7] gives a broad but also detailed view of the impact of Shor's and Grover's algorithm on individual encryption schemes and blockchain. Our overview of Shor's and Grover's algorithm is entirely based on [7]. Shor's algorithm can break encryptions based on the integer factorization problem (e.g. RSA), the discrete logarithm problem (e.g. DSA) [9] or elliptic curves (e.g. ECDSA) [7] in polynomial time.

Grover's algorithm can speed up the search for a specific item in an unstructured database of size N. Where it would normally take an effort of $N/2$, meaning $O(N)$ on average to find the specific item, Grover's algorithm can find the item in $O(\sqrt{N})$. This allows Grover's algorithm to break hash functions and symmetric encryption using the root of the effort it would take a conventional algorithm to [11]. For hash functions and symmetric encryption, the post-quantum security level is half of the classical security level because an encryption with classical security level of 128 takes approximately 2^{128} effort to break but Grover's algorithm reduces this to an effort of 2^{64}, consequently halving the security level. As some might have already guessed, this decrease in security can easily be countered by simply doubling the hash or key length to restore the initial security level and the performance only deteriorates linearly with increased hash or key length, making for a straightforward solution for post-quantum resistant hash functions and symmetric encryptions.

Shor's algorithm has a more severe impact on the security level of asymmetric encryption. Its polynomial time complexity is an enormous improvement compared to, for example the general number field sieve (GNFS) which can break asymmetric encryption in subexponential time. This allows breaking of previously considered secure encryption such as RSA-2048 in only hours given enough noisy qubits [9].The resulting post-quantum security level is considered broken meaning the security level is so low that it is nowhere near secure [7]. For one simple reason the solution for post-quantum resistant asymmetric encryption is not as easy as for hash functions and symmetric encryption. As seen before in [17] key length increases more than linearly when increasing security level. So, if we tried to increase key length in order to achieve a high enough post-quantum security level, the resulting key length would be too large, creating an encryption that runs too slow.

If no precautions are taken, the consequences of Shor's algorithm and Grover's algorithm might completely break blockchain. Using Shor's algorithm an attacker could find the private key to a cryptocurrency wallet from the public key or sign transactions in the name of someone else using that person's private key. Grover's algorithm would allow an attacker to create the longest chain of blocks, essentially changing the whole blockchain or changing the content of individual transactions by finding hash collisions between two valid transactions where one transaction was initially intended and the other enriches the attacker and replace the intended transaction with the other transaction. In order to avoid these possible consequences, there must be standards in place for post-quantum resistant encryption that developers and enterprises can rely on [7].

5.4.3 Post-Quantum resistant Encryption and Blockchains

We will take a closer look at two proposals for post-quantum resistant encryption algorithms. Namely, PQCrypto from Horizon 2020 EU [31] and the call for proposals for post-quantum Cryptography Standardization from NIST USA [27]. During the NIST call for proposals, which started in 2017 and will probably end in 2026, experts from different parts of the world judged a total of 82 algorithms in four rounds and decided on the following five post-quantum resistant algorithms for standardization [26]. However, it should first be noted that NIST uses 5 categories of security strength each associated with a block cipher or a SHA hash function instead of security levels to capture the security of the proposed algorithm [28]. For an algorithm to be in a given security strength category it needs to be as secure or more secure than the underlying encryption. The conversions between security strength category and security level and between security strength category and post-quantum security level that we used are displayed in table 2.2. It takes the Security strength categories and the underlying algorithms from [28] and assigns them both classical security levels and post-quantum security levels based on the relations between key/hash length and security level as described by [17] for classical security level and the fact that Grover's algorithm halves the post-quantum security level of block ciphers and hash functions.

Table 5.2: Security strength categories [28] and corresponding security levels

Security Strength Category	underlying algorithm	security level	pq security level
1.	block cipher 128-bits	128-bits	64-bits
2.	hash function 256-bits	128-bits	64-bits
3.	block cipher 192-bits	192-bits	96-bits
4.	hash function 384-bits	192-bits	96-bits
5.	block cipher 256-bits	256-bits	128-bits

Halving the post-quantum security level due to Grover's algorithm's quadratic speed up is considered by NIST as too extreme of an impact. This is because Grover's algorithm requires very long and difficult chains of serial computation which improves post-quantum resistance of algorithms exposed to Grover's algorithm [2]. This means, that the estimated post-quantum security level in table 2.2 are lowered than they are supposed to be.

Table 5.3: ML-KEM specifications [23]

version	encap key	decap key	ciphertext	shared secret key	SL	pqSL
ML-KEM-512	6400	6528	6144	256	128	64
ML-KEM-768	9472	19200	8704	256	192	96
ML-KEM-1024	12544	25344	12536	256	256	128

Table 5.4: ML-DSA specifications [24]

version	private key	public key	sig size	SL	pqSL
ML-DAS-44	20480	10496	19360	128	64
ML-DAS-65	32256	15616	26472	192	96
ML-DAS-87	39168	20736	37016	256	128

ML-KEM (Module-Lattice-Based Key-Encapsulation Mechanism) in table 2.3 is based on the CRYSTALS-Kyber algorithm and is a key-encapsulation mechanism (KEM) for secure exchange of a private key over a public channel which can then be used for symmetric encryption. Its security is based on the module mearning with errors problem which is believed to be post-quantum resistant. ML-KEM comes with three different, approved versions ML-KEM-512, ML-KEM-768 and ML-KEM-1025 [23] which have a security level of 128, 192 and 256 bits and a post-quantum security level of 64, 96 and 128.

The ML-DSA (module-lattice-based digital signature algorithm) in table 2.4 is a digital signature algorithm based on the CRYSTALS-Dilithium algorithm. ML-DSA's security is also based on the Module Learning with Errors problem and it offers three different, approved versions [24].

Table 5.5: SLH-DSA specifications [25]

version	sig size (bytes)	SL	pqSL
SLH-DSA-SHA2-128s	7856	128	64
SLH-DSA-SHAKE-128s	7856	128	64
SLH-DSA-SHA2-128f	17088	128	64
SLH-DSA-SHAKE-128f	17088	128	64
SLH-DSA-SHA2-192s	16224	192	96
SLH-DSA-SHAKE-192s	16224	192	96
SLH-DSA-SHA2-192f	35664	192	96
SLH-DSA-SHAKE-192f	35664	192	96
SLH-DSA-SHA2-256s	29792	256	128
SLH-DSA-SHAKE-256s	29792	256	128
SLH-DSA-SHA2-256f	49856	256	128
SLH-DSA-SHAKE-256f	49856	256	128

SLH-DSA (stateless hash-based digital signature algorithm) in table 2.5 is also a digital signature algorithm based on SPHINCS+. SLH-DSA has 12 different versions. For each of the Security level 128, 192, 256 there is one version that uses SHA2 for hashing and has small signatures ('s'), one using SHA2 that runs faster ('f') but that has larger signatures, one using SHAKE for hashing and has small signatures ('s'), and one using SHAKE that runs faster ('f') but that has larger signatures [25].

5.5 Related Work

[39] gives an overview of all the basic components used in blockchain and the types of blockchain and clears up misconceptions. It touches on security and the impact of quantum computers, but unlike this paper, it gives guidance on when and where it is advisable to add blockchain to an application, whereas this paper focuses on the applications and additional techniques. [37] explains further properties of the cryptographic hash function, gives a more detailed overview of iterative structures and compression functions to construct hash functions. Further, it describes at length the different types of attacks and specific attacks that hash functions are vulnerable to which this paper skipped nearly completely. [15] and [14] analyze the security of Merkle Trees and more specifically the collision and data falsification probabilities which try to fill a gap present in the literature.

[32] explains the fundamental techniques, network properties and types of blockchain and provides a Systematization of Knowledge (SoK) of all the cryptographic techniques already applied or with potential for application used in blockchain by thoroughly reviewing and systematizing literature. Furthermore, it discusses the challenges faced in blockchain at length and proposes 21 problems in the form of research problems yet to be solved regarding blockchain cryptography. [22] presents

real-world deployments and the lessons learned for various applications of data sharing using blockchain, which goes far beyond Multi-Party Computation, and listS research challenges for future studies. [12] addresses a plethora of security aspects of IoT in the different layers (sensing, network, middleware and application layer) and the future security requirements. Furthermore, it explains the IoT security of blockchain and various other technologies and their benefits.

[40] gives an overview of Quantum Computing, its impact, post-quantum resistant algorithms and blockchains. However, it also discusses a second type of quantum resistant blockchain that utilizes quantum computing to create quantum blockchain. It compares the two approaches to quantum-resistant blockchains, namely blockchain using post-quantum resistant encryption and blockchain using quantum computers to be quantum resistant. It also incorporates research challenges and lessons learned.

5.6 Conclusion

This report has presented various aspects of blockchain. It has first communicated fundamental knowledge about history, components, properties and fundamental cryptographic techniques. It has dived deeper into asymmetric encryption, hashing, merkle trees, layer-2 solutions and state channels, and showed how these techniques provide properties such as data authentication, data integrity, scalability and performance. Furthermore, it discussed the properties of blockchain applications in data sharing, IoT and self-sovereign identity in combination with additional cryptographic techniques such as MPC, PUF, accumulators and zero-knowledge proofs. The report covered classical security aspects of encryption and blockchain and explored post-quantum resistant encryptions proposed by NIST and PQCrypto that are going to be used in post-quantum resistant blockchains in the future.

References

[1] Muhammad Aman, Kee Chua, and Biplab Sikdar. “Position Paper: Physical Unclonable Functions for IoT Security”. In: May 2016, pp. 10–13. DOI: 10.1145/2899007.2899013.

[2] Elaine Barker. *Recommendation for key management: Part 1 - General*. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf [Accessed: (April 30, 2025.)] 2020.

[3] Darko Capko, Srđan Vukmirovic, and Nemanja Nedic. “State of the Art of Zero-Knowledge Proofs in Blockchain”. In: *2022 30th Telecommunications Forum (TELFOR)*. Nov. 2022, pp. 1–4. DOI: 10.1109/âĎąFOR56187.2022.9983760. (Visited on 04/24/2025).

[4] Uwe Der, Stefan Jaehnichen, and Jan Suermeli. *Self-sovereign Identity - Opportunities and Challenges for the Digital Revolution*. arXiv:1712.01767 [cs]. Dec. 2017. DOI: 10.48550/arXiv.1712.01767. URL: http://arxiv.org/abs/1712.01767 (visited on 04/24/2025).

[5] Massimo Di Pierro. "What Is the Blockchain?" In: *Computing in Science & Engineering* 19.5 (2017), pp. 92–95. ISSN: 1558-366X. DOI: 10.1109/MCSE.2017.3421554. (Visited on 04/23/2025).

[6] Md Sadek Ferdous, Farida Chowdhury, and Madini O. Alassafi. "In Search of Self-Sovereign Identity Leveraging Blockchain Technology". In: *IEEE Access* 7 (2019), pp. 103059–103079. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2931173. (Visited on 04/24/2025).

[7] Tiago M. Fernandez-Carames and Paula Fraga-Lamas. "Towards Post-Quantum Blockchain: A Review on Blockchain Cryptography Resistant to Quantum Computing Attacks". In: *IEEE Access* 8 (2020), pp. 21091–21116. DOI: 10.1109/ACCESS.2020.2968985.

[8] A. Gambhir and R. Arya. *Performance Analysis of DES Algorithm and RSA Algorithm with Audio Steganography*. en. 2017. URL: %5Curl%7Bhttps://www.researchgate.net/publication/314521173_Performance_Analysis_of_DES_Algorithm_and_RSA_Algorithm_with_Audio_Steganography%7D (visited on 04/23/2025).

[9] Craig Gidney and Martin Ekera. "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits". In: *Quantum* 5 (2021), p. 433.

[10] J. Goebel and A.E. Krzesinski. "Increased block size and Bitcoin blockchain dynamics". In: *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*. 2017, pp. 1–6. DOI: 10.1109/ATNAC.2017.8215367.

[11] Lov K. Grover. "A fast quantum mechanical algorithm for database search". In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*. STOC '96. Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996, pp. 212–219. DOI: 10.1145/237814.237866.

[12] Vikas Hassija et al. "A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures". In: *IEEE Access* 7 (2019), pp. 82721–82743. DOI: 10.1109/ACCESS.2019.2924045.

[13] Uzair Javaid, Muhammad Aman, and Biplab Sikdar. "BlockPro: Blockchain based Data Provenance and Integrity for Secure IoT Environments". In: Nov. 2018, pp. 13–18. DOI: 10.1145/3282278.3282281.

[14] Oleksandr Kuznetsov et al. "Evaluating the security of Merkle trees: An analysis of data falsification probabilities". In: *Cryptography* 8.3 (2024), p. 33.

[15] Oleksandr Kuznetsov et al. "Merkle trees in blockchain: A study of collision probability and security implications". In: *Internet of Things* (2024), p. 101193.

[16] K. Lauter. "The advantages of elliptic curve cryptography for wireless security". In: *IEEE Wireless Communications* 11.1 (Feb. 2004), pp. 62–67. ISSN: 1558-0687. DOI: 10.1109/MWC.2004.1269719. (Visited on 04/23/2025).

[17] Arjen K Lenstra. "Key lengths". In: *The Handbook of Information Security* (2006).

[18] Pin Lv et al. "An IOT-Oriented Privacy-Preserving Publish/Subscribe Model Over Blockchains". In: *IEEE Access* 7 (2019), pp. 41309–41314. DOI: 10.1109/ACCESS.2019.2907599.

[19] Dindayal Mahto and Dilip Kumar Yadav. "RSA and ECC: A comparative analysis". en. In: *ResearchGate* 12.19 (2017), pp. 9053–9061. ISSN: 0973-4562. URL: %5Curl%7Bhttps://www.researchgate.net/publication/322558426_RSA_and_ECC_A_comparative_analysis%7D (visited on 04/23/2025).

[20] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. en.

[21] Lydia D. Negka and Georgios P. Spathoulas. "Blockchain State Channels: A State of the Art". In: *IEEE Access* 9 (2021), pp. 160277–160298. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3131419. (Visited on 04/23/2025).

[22] Thanh Linh Nguyen et al. "Blockchain-empowered trustworthy data sharing: Fundamentals, applications, and challenges". In: *ACM Computing Surveys* 57.8 (2025), pp. 1–36.

[23] NIST. *FIPS 203*. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.203.pdf [Accessed: (April 30, 2025.)] 2024.

[24] NIST. *FIPS 204*. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.204.pdf [Accessed: (April 30, 2025.)] 2024.

[25] NIST. *FIPS 205*. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.205.pdf [Accessed: (April 30, 2025.)] 2024.

[26] NIST. *NIST Releases First 3 Finalized Post-Quantum Encryption Standards*. https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards [Accessed: (April 30, 2025.)] 2024.

[27] NIST. *Post-Quantum Cryptography*. https://csrc.nist.gov/projects/post-quantum-cryptography [Accessed: (April 30, 2025.)] 2025.

[28] NIST. *Submission requirements and evaluation criteria for the post-quantum cryptography standardization process*. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf [Accessed: (April 30, 2025.)] 2016.

[29] Oscar Novo. "Blockchain Meets IoT: An Architecture for Scalable Access Management in IoT". In: *IEEE Internet of Things Journal* 5.2 (2018), pp. 1184–1195. DOI: 10.1109/JIOT.2018.2812239.

[30] Ilker Ozcelik et al. "An Overview of Cryptographic Accumulators". In: *Proceedings of the 7th International Conference on Information Systems Security and Privacy*. arXiv:2103.04330 [cs]. 2021, pp. 661–669. DOI: 10.5220/0010337806610669. URL: http://arxiv.org/abs/2103.04330 (visited on 04/24/2025).

[31] PQCRYPTO. *PQCRYPTO*. https:\pqcrypto.eu.org [Accessed: (April 30, 2025.)] 2018.

[32] Mayank Raikwar, Danilo Gligoroski, and Katina Kralevska. "SoK of Used Cryptography in Blockchain". In: *IEEE Access* 7 (2019), pp. 148550–148575. DOI: 10.1109/ACCESS.2019.2946983.

[33] Ronald L Rivest. "The md4 message digest algorithm". In: *Lecture notes in Computer Science, Advances in Cryptography CRYPTO '90* (1998).

[34] Haitz Saez de Ocariz Borde. "An Overview of Trees in Blockchain Technology: Merkle Trees and Merkle Patricia Tries". In: (Feb. 2022).

[35] Cosimo Sguanci, Roberto Spatafora, and Andrea Mario Vergani. *Layer 2 Blockchain Scaling: a Survey*. arXiv:2107.10881 [cs]. July 2021. DOI: 10.48550/arXiv.2107.10881. (Visited on 04/23/2025).

[36] Adi Shamir. "How to share a secret". In: *Commun. ACM* 22.11 (Nov. 1979), pp. 612–613. ISSN: 0001-0782. DOI: 10.1145/359168.359176.

[37] Rajeev Sobti and Geetha Ganesan. "Cryptographic Hash Functions: A Review". In: *International Journal of Computer Science Issues, ISSN (Online): 1694-0814* Vol 9 (Mar. 2012), pp. 461–479.

[38] G. Edward Suh and Srinivas Devadas. "Physical Unclonable Functions for Device Authentication and Secret Key Generation". In: *2007 44th ACM/IEEE Design Automation Conference*. 2007, pp. 9–14.

[39] Dylan Yaga et al. *Blockchain Technology Overview*. en. June 2019. DOI: 10.6028/NIST.IR.8202. URL: %5Curl%7Bhttps://arxiv.org/abs/1906.11078v1%7D (visited on 04/23/2025).

[40] Zebo Yang et al. "A Survey and Comparison of Post-Quantum and Quantum Blockchains". In: *IEEE Communications Surveys & Tutorials* 26.2 (2024), pp. 967–1002. DOI: 10.1109/COMST.2023.3325761.

[41] Guy Zyskind, Oz Nathan, and Alex Pentland. "Enigma: Decentralized Computation Platform with Guaranteed Privacy". In: (June 2015). DOI: 10.48550/arXiv.1506.03471.

Chapter 6

Securing the Future — Post Quantum Cryptography in Messenger Systems

Shirley Feng Li Lau, Linn Anna Spitz

In the wake of quantum computers unlocking novel computational possibilities, experts are raising serious concerns about the future security of traditional cryptographic methods. Given the impact of Shor's algorithm, many foundational encryption methods are entirely compromised if quantum computers become more practical and available. This is not solely a problem of the future, as "harvest-now-decrypt-later" attacks pose a serious threat right now. A major effort led by researchers and experts has put forth numerous proposals for Post-Quantum Cryptography (PQC) and the National Institute for Standards and Technology (NIST) has played a pivotal role in evaluating and standardizing novel cryptographic methods. Resulting algorithms include CRYSTALS-Kyber, CRYSTALS-Dilithium and SPHINCS$^+$. Messaging systems are a major area of application for PQC, with Apple's iMessage being an early adopter, as it already uses the PQ3 protocol. PQC is still facing architectural challenges, as algorithms generally entail larger key sizes and computational overhead compared to traditional methods.

Contents

6.1 Introduction 125
6.2 Background 126
6.2.1 Overview of Secure Communication 126
6.2.2 Quantum Computing Threats 127
6.2.3 The Need for Post-Quantum Cryptography (PQC) . . . 128
6.3 PQC Algorithms 129
6.3.1 Post-Quantum Cryptography (PQC) 130
6.3.2 Standardization process 132
6.3.3 Post Quantum Key Establishment 133
6.3.4 Post-Quantum Digital Signatures 133
6.4 Approaches for Implementing PQC 134
6.4.1 Applications of PQC 134
6.4.2 Industry Adoption of PQC in Messenger Apps 135
6.4.3 Architectural Considerations 138
6.4.4 Future Directions 140
6.5 Conclusion 141

6.1 Introduction

In the wake of quantum computers unlocking novel computational possibilities, experts are raising serious concerns about the future security of traditional cryptographic methods [5]. Grover's and Shor's algorithm are quantum algorithms that have been proven to accelerate the adversarial decryption of standard encryption methods [46, 18]. The impact of Shor's algorithm, in particular, has been significant, as it offers an exponential speed-up on breaking public-key encryption methods such as Rivest-Shamir-Adleman (RSA), Diffie-Hellman (DH) and Elliptic Curve Cryptography (ECC), on which the current technological ecosystem heavily relies. Meaning that if quantum computers become more practical and available, these foundational encryption algorithms are entirely compromised.

However, this threat is not merely a future concern. Experts believe that attackers may already be harvesting sensitive data that is secure under the current computational limitations. If those attackers gain access to quantum resources in the future, they could decrypt data collected today. This is known as a *"harvest-now-decrypt-later"* attack. Considering this possibility, data must be secured against quantum threats even before such attacks can be practically executed [37].

A major effort lead by researchers and experts has put forth numerous proposals for Post-Quantum Cryptography (PQC). Public-key cryptography relies on the difficulty of solving certain mathematical problems, so it is critical to identify new sets of mathematical problems that are intractable even for quantum computers. Current candidates for such problems are the Learning With Errors (LWE) problem and, its extension, the Module Learning With Errors (MLWE) problem [33, 32, 41], which are both considered forms of lattice-based problems [40]. Another family of problems that are assumed to be quantum resistant are based on error-correcting codes. Additionally, post-quantum hash-based signature schemes have been developed [7, 34].

In the search for PQC algorithms, the National Institute for Standards and Technology (NIST) has played a pivotal role. In the past years, NIST has evaluated candidate algorithms and standardized the most promising among them [35, 36]. Standardization is an important step in cryptography, as a major challenge currently facing the PQC community is the compatibility with legacy systems, as well the interoperability of new systems. The algorithms that NIST has standardized so far include CRYSTALS-Kyber, CRYSTALS-Dilithium and SPHINCS$^+$ [33, 32, 34].

Major application areas of PQC include Internet of Things (IoT) architectures, cloud-computing, military communication and messaging apps [24, 17, 26]. Apple's iMessage is an early adopter of PQC, as its PQ3 protocol has already been fully implemented. PQ3 additionally uses a hybrid approach, as it includes two layers of encryption for key-exchange: a post-quantum layer and a traditional encryption layer. This hybrid approach ensures that an encryption is as least as safe as the traditional approach, while the research community is still evaluating the security of PQC encryption schemes [26].

The adoption of PQC is currently still facing some architectural challenges. PQC generally entails larger key sizes and more computational overhead compared to traditional systems [21]. Given these considerations, the latest research directions are focusing on designing schemes with smaller key sizes that still provide strong quantum resistance, balancing storage, communication, and computational overhead [45].

This paper first establishes the need for post-quantum cryptography by examining the threat that quantum computers pose to current cryptographic systems. It then provides an overview of PQC algorithms, highlighting the mathematical foundations of emerging cryptographic methods. Finally, the paper explores practical implementations of PQC systems, discusses relevant architectural considerations and outlines potential directions for future research in the field.

6.2 Background

This section explores the motivation for post-quantum cryptography. It begins by outlining essential background in traditional cryptographic systems. Subsequently, it discusses the quantum threat posed by Grover's and Shor's algorithms, demonstrating their implications for widely used encryption schemes. This leads to the necessity for developing cryptographic techniques that remain secure in the presence of quantum adversaries.

6.2.1 Overview of Secure Communication

The most significant distinction within encryption schemes lies in the difference between asymmetric and symmetric encryption schemes. In asymmetric encryption, a message can be encrypted by anyone using the receiver's public key. Subsequently, the message can only be decrypted using the receiver's private key. In symmetric encryption, both parties have access to the same secret key [28].

Asymmetric encryption schemes include RSA, DH key exchange and ECC. All asymmetric encryption algorithms are based on a mathematical problem, that is relatively easy to compute in one direction but computationally hard to reconstruct in the other direction. RSA is based on the difficulty of factorizing large prime numbers [43]. DH is based on the discrete logarithm problem (DL) [13] and ECC is based on repeatedly sampling elliptic curves [22, 30]. In the area of symmetric key encryption, the most common standard is the Advanced Encryption Standard (AES), which was introduced by NIST in 2001 [14].

In practice, asymmetric and symmetric encryption are often used together. Transport Layer Security (TLS) is used with the HTTP protocol, among other communication systems. In TLS, asymmetric keys are used for the handshake phase between two communicating parties. During this handshake phase, a shared symmetric key

is established. This is then used to symmetrically encrypt the subsequent exchange between the parties [50].

Cryptography is not exclusively used for confidentiality. It is also essential for verification. For example, digital signatures are used to verify identity. Both symmetric and asymmetric digital signatures exist. Simple hash signatures are an example of an asymmetric signature. This can be expanded to include a symmetric key, as is the case for Message Authentication Codes (MAC) [28].

6.2.2 Quantum Computing Threats

Conventional encryption algorithms are based on mathematical problems that are hard to solve by traditional computers. With advances in quantum computing, some problems could soon be solved in a much shorter time frame. Grover's and Shor's algorithms are quantum algorithms that have had significant impact on the field of cryptography.

Grover's Algorithm

Grover's algorithm was originally designed to search unstructured databases with N entries. As there is no sorting or index in unstructured databases, this task is traditionally performed in $O(N)$ time. Grover's algorithm leverages quantum computing to achieve this in $O(\sqrt{N})$ time [18].

Unstructured search can be generalized to a wide array of problems. For example, optimization is an unstructured search for the optimal solution. Consider the traveling salesman problem (TSP). It is concerned with finding the minimal round trip that covers N cities. The TSP is both an NP-complete problem, as well as an unstructured search.

NP-complete problems are a class of problems in computer science and complexity theory. No polynomial-time algorithm is currently known for any NP-complete problem, and it is conjectured that none exists, which is known as the P vs NP question. A key feature of NP-complete problems is their inter-reducibility. Every NP-complete problem can be reduced to any other problem of its class in polynomial time. Thus, if one NP-complete problem can be solved in polynomial-time, then so can every other NP-complete problem [48].

Given the nature of NP-complete problems, the fact that the traveling salesman can be generalized to an unstructured search directly implies that all NP-complete problems can be generalized to an unstructured search. Thus, Grover's algorithm is very significant in computer science [19].

As there are $N!$ possible paths in the TSP, this unstructured search has a traditionally factorial runtime of $O(N!)$, which has a super-exponential growth rate.

However, the speed-up that Grover's algorithm provides is quadratic, which limits the impact of the algorithm. For example, even with a quadratic speed-up, exponential problems remain exponential. Therefore, NP-complete problems that are exponential in complexity or worse, remain so even in the realm of quantum computing. Within the field of encryption, the quadratic speed-up of Grover's algorithm reduces the security level of the affected schemes by half. Thus, in return, doubling key-sizes effectively eliminates the impact of Grover's algorithm [5].

Given the highly general nature of unstructured search, the majority of existing encryption schemes are affected by Grover's algorithm. However, we consider Grover's algorithm as most relevant to symmetric encryption, as traditional asymmetric encryption is additionally vulnerable to Shor's algorithm, which provides a much greater speed-up [5].

Shor's Algorithm

In 1994, Peter Shor proposed an algorithm that allows for solving prime factorization in polynomial time by employing quantum computer technology [46]. In RSA encryption, the large number N is the public key, with the secret prime factors p and q, so that $p * q = N$. Shor's algorithm can find p and q by analyzing the periodic function $f(x) = a^x \bmod N$, where $a < N$ is a random number, co-prime with N. If the resulting period is non-trivial, it can then in turn be used to compute the factors p and q. Using a quantum computer, a suitable input for a can be found in polynomial time, by analyzing the periodicity of many inputs of a at the same time using the *Quantum Fourier-Transform* [19].

More generally, Shor's algorithm can be applied to any problem that can be analyzed through periodicity in a similar manner. Subsequently, not only RSA, but DH and ECC are vulnerable to attacks that leverage periodicity.

6.2.3 The Need for Post-Quantum Cryptography (PQC)

While both Grover's and Shor's algorithm have been proven theoretically, the corresponding hardware does not yet exist. However, it is possible that it will in the future. Given this future potential, it has been argued that current communications are still under threat. The authors of [37] posit that the *"harvest-now-decrypt-later"* strategy, abbreviated as HNDL and pronounced *"Handle"*, poses a significant risk today. HNDL is an attack where encrypted messages are collected and stored by attackers in anticipation of quantum computing being available to decrypt the content in the future. Therefore, current communication ought still be protected against decryption techniques that are not yet viable.

To mitigate future risks, cryptographic researchers and organizations have started to work on alternative algorithms that remain safe even against quantum computers. NIST has proposed new encryption schemes such as CRYSTALS-Kyber [33]

Name	Function	Pre-quantum security level	Post-quantum security level
Symmetric cryptography			
AES-128[8]	Symmetric encryption	128	64 (Grover)
AES-256[8]	Symmetric encryption	256	128 (Grover)
Salsa20[58]	Symmetric encryption	256	128 (Grover)
GMAC[59]	MAC	128	128 (no impact)
Poly1305[60]	MAC	128	128 (no impact)
SHA-256[61]	Hash function	256	128 (Grover)
SHA3-256[62]	Hash function	256	128 (Grover)
Public-key cryptography			
RSA-3072[1]	Encryption	128	Broken (Shor)
RSA-3072[1]	Signature	128	Broken (Shor)
DH-3072[42]	Key exchange	128	Broken (Shor)
DSA-3072[63,64]	Signature	128	Broken (Shor)
256-bit ECDH[4–6]	Key exchange	128	Broken (Shor)
256-bit ECDSA[66,67]	Signature	128	Broken (Shor)

Figure 6.1: Traditional encryption methods and their respective security levels with respect to quantum computers [5].

and CRYSTALS-Dilithium [32]. This paper explores how messaging platforms are integrating PQC and the associated challenges.

Figure 6.1 shows traditional encryption methods that are widely used today and analyzes the threat quantum computing poses to their prolonged security. As the table shows, all traditional PKE systems are rendered insecure by the exponential speed-up provided by Shor's algorithm [5, 19]. Symmetric key encryption schemes are more resilient, as there is no public information to exploit, an attack requires an exhaustive search for the secret key. The same applies to hash functions, where an attack must search for the preimage. As previously discussed, this can be sped-up quadratically through Grover's algorithm [19]. As a result, the security level of the algorithms is halved. This can be effectively addressed by doubling the key-sizes of the respective algorithms. MAC-based schemes are not impacted, as they don't allow for the verification of a given guess. Thus even a brute-force search is impossible [11], which rules out quantum-acceleration.

6.3 PQC Algorithms

If quantum hardware advances, the current standard public-key cryptographic methods would be compromised. Thus, the search for viable replacements has

been extensive. This section presents key post-quantum cryptographic algorithms along with the underlying mathematical foundations. It also outlines the standardization process led by NIST and discusses essential cryptographic mechanisms, such as key encapsulation and digital signatures, in the context of post-quantum security.

6.3.1 Post-Quantum Cryptography (PQC)

Public-key encryption (PKE) is based on math problems that are hard to solve with traditional computing technology. It follows that a new set of math problems must form the basis of future PKE algorithms. One such problem is the Module Learning with Errors Problem (MLWE) [41]. NIST has standardized two algorithms based on MLWE, CRYSTAL-Kyber and CRYSTAL-Dilithium [33, 32]. Hash-based systems are also presumed to be quantum-resistant. Consequently, NIST has standardized SPHINCS$^+$, a hash-based digital signature algorithm [34].

Module Learning With Errors Problem

The MLWE is an extension of the Learning with Errors Problem (LWE). In the LWE, the private key is a vector $x \in \mathbb{Z}_q^n$. The public key is a set of m equations, where $m > n$, with n unknown variables each. All of the equations are satisfied given the values of the secret vector. Small errors are then added to the public equations. The equations are used to encode secret messages. With the private key, one can easily evaluate equations and get the correct result through rounding. Reconstructing the solution, thus the secret vector, to the public equations is very hard, given the errors. Consider that with the added errors the system of equations is likely not even solvable [41]. In MLWE, the system of linear equations is replaced with a ring of polynomials, which is defined as follows.

$$\mathbb{Z}_q[x]/(x^n + 1) \tag{6.1}$$

A ring of polynomials in x is thus a set as polynomial equations $f(x)$, where coefficients are in the set $\mathbb{Z}_q$ and the order of polynomials is always smaller than n. In the definition, the polynomial $\mathbb{Z}_q[x]$ is divided by $(x^n + 1)$, which a similar effect as a modulo-operation in modular arithmetic. Polynomials of degree $>= n$ are divided, so that they remain within the range of the ring [40]. A matrix A is defined to contain only entries from $\mathbb{Z}_q[x]/(x^n + 1)$. The secret vector is multiplied with A and errors are added. Thus, the construction of an MLWE-based encryption is similar its LWE-based counterpart, but the secret vector $x \in \mathbb{Z}_q^n$ is replaced by secret vector $x \in R_q^n$ as defined previously [33].

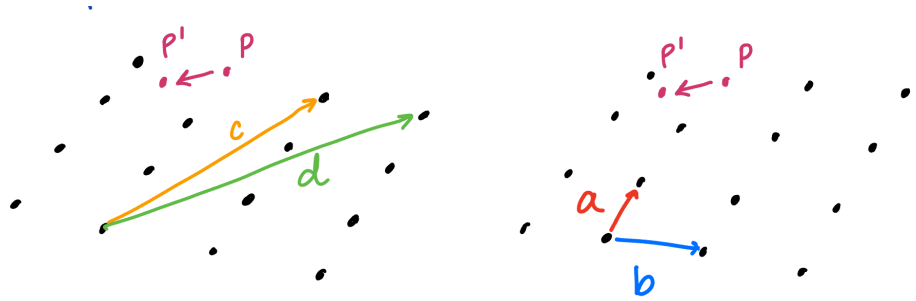


Figure 6.2: An example of a 2D-lattice with basis vectors (a, b) and (c, d), as well as the message P and the encrypted message P'.

Lattice-Based Cryptography

Lattice-based problems are related to LWE and MLWE problems [40]. In fact, either can be transformed into the other. Consider the system of equations discussed previously with regards to LWE- and MLWE-based systems. These equations contain errors, as previously established. This is related to the Closest Vector Problem (CVP) in lattice-based cryptography, shown in Figure 6.2. In this context, a lattice is defined by basis vectors which constitute the private key. For simplicity, the lattice is assumed to be 2D-dimensional with the two basis vectors (a, b). A message corresponds to a point P within a lattice. Its encryption involves adding a small random error to P to get the nearby point P'. The public key consists of a different set of basis vectors, (c, d), which generate the same lattice, don't allow for efficient computation. In order to decrypt, one must find the closest point to P' on the lattice, P. This is fast using the private basis (a, b), but computationally hard to with the public basis (c, d) [29].

CRYSTALS-Kyber

CRYSTALS-Kyber, also referred to as ML-KEM (Module-Lattice-Based Key-Encapsulation Mechanism), is a quantum-resistant key-encapsulation mechanism (KEM) standardized by NIST in 2024 [35, 33]. KEMs are a class of algorithms that use public-key cryptography with the goal of safely exchanging a shared secret key between two communicating parties. CRYSTALS-Kyber is based on the MWLE problem [33]. As MLWE is a lattice-based, the Kyber algorithm also falls within the category of lattice-based cryptography.

CRYSTALS-Dilithium

CRYSTALS-Dilithium is officially referred to as the Module-Lattice-Based Digital Signature Algorithm (ML-DSA). It is a digital signature scheme based on the MWLE-problem and is therefore lattice-based [32]. It was standardized by NIST in 2024.

SPHINCS$^+$

SPHINCS$^+$ is a stateless hash-based signature scheme that is assumed to be quantum resistant and has been standardized by NIST. A binary message $m \in 0, 1^n$ can be authenticated by publishing it alongside a set of $2n$ hash values, generated by a known hash function $h(x)$. Each bit in m is associated with two values x_0 and x_1, corresponding to the two possible values of the bit. These values are kept secret, while their hashed values $h(x_0)$ and $h(x_1)$ are published. To authenticate a specific message, the author must reveal the input value x_0 or x_1 corresponding to the value of each bit. Observers can verify that the given value indeed hashes to the hash value that was originally published with the message. As this method generates large public keys, optimizations exist. SPHINCS$^+$ uses a merkle tree, where the $2n$ hash values are places on the leaves of a binary tree. In each internal node, two hash values are combined into one, an rehashed. This results in a single hash value at the root, which replaces the public key [34, 7].

As the secret values have to be revealed for authentication, reusing secret values must be avoided, as this compromises the security. This would render the system stateful, as a history of previous values must be maintained. SPHINCS$^+$ uses an additional technique to avoid this problem. It constructs a hypertree, a tree of merkle trees, where each merkle tree contains different values. Using a mapping function with the message as input, a given message will always use a specific merkle tree and thus a specific set of values. However, two different messages will always use different sets of values. As the secret values can now only be used for a specific message, impostors can now recreate the signature of a message previously posted by the author, but they cannot sign a new message [34, 7, 6].

6.3.2 Standardization process

The National Institute of Standards and Technology (NIST) is leading an effort to develop and standardize new cryptographic algorithms that will be secure against quantum attacks. This initiative started at PQCrypto 2016 conference and by the end of 2017, NIST had received 23 digital signature schemes and 59 encryption or key encapsulation mechanisms (KEMs) [35]. After an initial review, 69 candidates were deemed eligible to proceed to the first round of evaluation. Over several rounds, NIST carefully assessed the security, efficiency, and practicality of these algorithms.

As of end of 2024, NIST officially released the first set of finalized post-quantum cryptography standards, marking a major milestone. These standards are:

- FIPS 203 [33] — for general encryption, based on the CRYSTALS-Kyber algorithm, now named ML-KEM (Module-Lattice-Based Key-Encapsulation Mechanism).

- FIPS 204 [32] — for digital signatures, using the CRYSTALS-Dilithium algorithm, now named ML-DSA (Module-Lattice-Based Digital Signature Algorithm).
- FIPS 205 [34] — also for digital signatures, employing the SPHINCS$^+$ algorithm, now named SLH-DSA (Stateless Hash-Based Digital Signature Algorithm), which offers a backup approach in case lattice-based methods show weaknesses.

To help organizations smoothly migrate [36] to utilizing new Post Quantum Cryptography standards, NIST also develop frameworks that focuses on two aspects - interoperability and performance. Interoperability ensures that cryptographic systems can exchange data securely without compatibility issues or security vulnerabilities, which is crucial in environments like the internet, banking systems, or cloud platforms where systems from multiple vendors or regions need to communicate. NIST also provides a controlled, non-production environment where these issues can be safely examined and fixed.

6.3.3 Post Quantum Key Establishment

Post-quantum key establishment (PQ KEX) is the process of two parties securely generating a shared secret using cryptographic techniques that are resistant to attacks from both classical and quantum computers.

To do this, each party generates a public-key and private-key pair (Keygen). [16] One classical key (e.g., Elliptic Curve P-256) for compatibility with existing systems, and one post-quantum key (e.g., Kyber-1024) for future security. If one party, Alice wishes to send Bob a message, she retrieves Bob's public keys. After validating it, she encapsulates a random secret using Bob's public key, and sends over this ciphertext to Bob. Bob decapsulates it using his private key, recovering the same shared secret. Alice and Bob have now established a shared session. The shared secret is processed using a Key Derivation Function (KDF) e.g., HKDF-SHA384) to create the final encryption key. If a hybrid approach is used, the system combines both a classical and post-quantum secret to ensure security even if one is broken. The derived key is used for encrypting messages which secures all future communication. With this, the protocol has also ensured forward secrecy, protecting any future attacks from exposure to past sessions. The hybrid approach [42] is also essential to ensure progress to post-quantum algorithms and provide post-quantum resistance while retaining the existing security of current cryptographic schemes.

6.3.4 Post-Quantum Digital Signatures

Currently, hybrid signature schemes combining classical (ECDSA) and quantum-resistant algorithms (Falcon) are used. These methods produce compact hybrid

signatures, which are shorter than simple concatenations of classical and post-quantum signatures. They maintain backward compatibility and regulation conformance by allowing selective verification of either signature type individually [25]. It is done by merging two randomized digital signature schemes in a way that is reversible and can be individually verified. This construction allows for a smooth transition from legacy systems that only understand classical signatures (like ECDSA) while leveraging on post-quantum security.

Originating from Shannon's theory, error-correcting codes are integral in scenarios like sending images from deep space and maintaining integrity in everyday identifiers such as credit card numbers, ISBN on books, and phone numbers [47]. Unlike RSA and ElGamal, whose security depends on factoring large integers or solving discrete logarithms, McEliece cryptosystem relies on the complexity of decoding random errors in linear error-correcting codes. Quantum computing, notably Shor's algorithm, does not significantly enhance attacks against these code-based schemes. Hence, the McEliece cryptosystem remains robust against quantum threats.

Algorithms like Falcon also specifically offer compact signatures using lattice structures and Fast Fourier Transform optimizations [38], which significantly reduce the size compared to classical counterparts like RSA, facilitating efficient transmission which is useful in high-volume data scenarios. Hash-based schemes (e.g., SPHINCS$^+$) provide multiple variants of hash functions (e.g. SHA2 or SHAKE) which can be easily adapted according security requirements. This makes them accessible for practical use despite their relatively large signature sizes.

6.4 Approaches for Implementing PQC

As quantum computers get more powerful, they could break the encryption that messaging apps rely on to keep conversations private. This section looks at how post-quantum cryptography (PQC) can be used in messaging systems to protect user data. We explore the current use of encryption in popular apps and how PQC can help make them more secure in the future.

6.4.1 Applications of PQC

It is anticipated that the development of large-scale quantum computers will pose a significant threat to modern communication channels and the cryptographic foundations that enable them. Messaging platforms, Internet of Things (IoT) networks, and cloud computing environments have become central components of modern digital infrastructure, enabling real-time communication, remote control, and scalable data processing. Across all three, the potential for *"harvest-now-decrypt-later"* attacks and the long-term vulnerability of current public-key cryptographic schemes call for the integration of quantum-resistant solutions in these environments.

By far, messaging apps are one of the widely-used form of communication channels. In addition to data leaks and illegal third party sales of consumer data [31], consumers have also become more privacy-aware to attempts of mass surveillance from different parties. These platforms rely heavily on public-key cryptography to provide end-to-end encryption, ensuring the confidentiality, integrity, and authenticity of messages exchanged between users. Widely used communication and messaging protocols, e.g., TLS 1.3 and Signal, use modifications of the Diffie-Hellman algorithm for key exchange [8]. More specifically, messaging apps like Whatsapp, Signal and Facebook Messenger use end-to-end encryption [2]. In addition to end-to-end encryption, messaging apps can also use post-quantum cryptographic techniques, such as digital signatures to verify the authenticity of users and ensure the integrity of the message conveyed.

IoT connects billions of heterogeneous devices (expected 18.8 billion by 2025) across networks, forming a interconnected network of devices that exchange sensitive data using security protocols. Such protocols should include data protection schemes in the design and application phases for high secrecy situations. One such application is the security requirement in military and defense areas where unmanned aerial vehicles (UAVs) are used to exchange information while monitoring large areas [24]. Thus, new cryptographic schemes that do not rely on mathematical difficulty should be developed, and instead utilize complex quantum-resistant formulations that are untraceable.

Large-scale distributed systems have already started to incorporate PQC techniques. With its own internal encryption-in-transit protocol, Application Layer Transport Security (ALTS), Google Cloud [17] has already enabled one of the algorithms in defense to potential post-quantum attacks. It uses public-key cryptography algorithms to ensure that Google's internal infrastructure components talk to each other with more secure authentication and encryption, better safeguarding interservice communication and data storage.

6.4.2 Industry Adoption of PQC in Messenger Apps

At present, the industrial standard for cryptography in messaging apps is end-to-end encryption (E2EE). Messaging apps like Whatsapp and Signal are providing end-to-end encryption in their messaging protocols. However, majority of other messaging apps conventionally only provide encrypted communications without true end-to-end encryption. This means that after the message travels from the sender to the server, it gets decrypted briefly before being re-encrypted and sent to the recipient. This introduces a small loophole at the server, potentially exposing the content of the message to the service provider. With end-to-end encryption, messages no intermediary party, even the service provider, would be able to decrypt and access the message before it reaches the recipient [12]. In E2EE, a message encrypted with the recipient's public key can only be decrypted by the recipient's private key, ensuring that even if an attacker intercepts the message, they cannot read it. Thus any compromise to a long-term key or a temporary access to the session will not threaten the security of past and future messages.

Apple's iMessage: While end-to-end encryption is secure for most applications now, they rely on pre-quantum cryptographic algorithms, such as RSA, Diffie-Hellman and ECC, and are also subject to post-quantum threats. Therefore, messaging apps like Apple's iMessage have already been at the forefront of advancing the state of art in end to end encryption with PQ3 by introducing post-quantum techniques. This includes incorporating post-quantum cryptography from the start of a conversation, so that all communication is protected from current and future adversaries. Furthermore, additional layers of security are utilized with post-quantum algorithms in order to ensure forward secrecy, limiting the number of past and future messages that can be decrypted if a session key is exposed.

The Apple PQ3 Protocol offers heightened encryption with post-quantum algorithms using two techniques [26]:

- Hybrid Key Exchange
- Triple Ratcheting

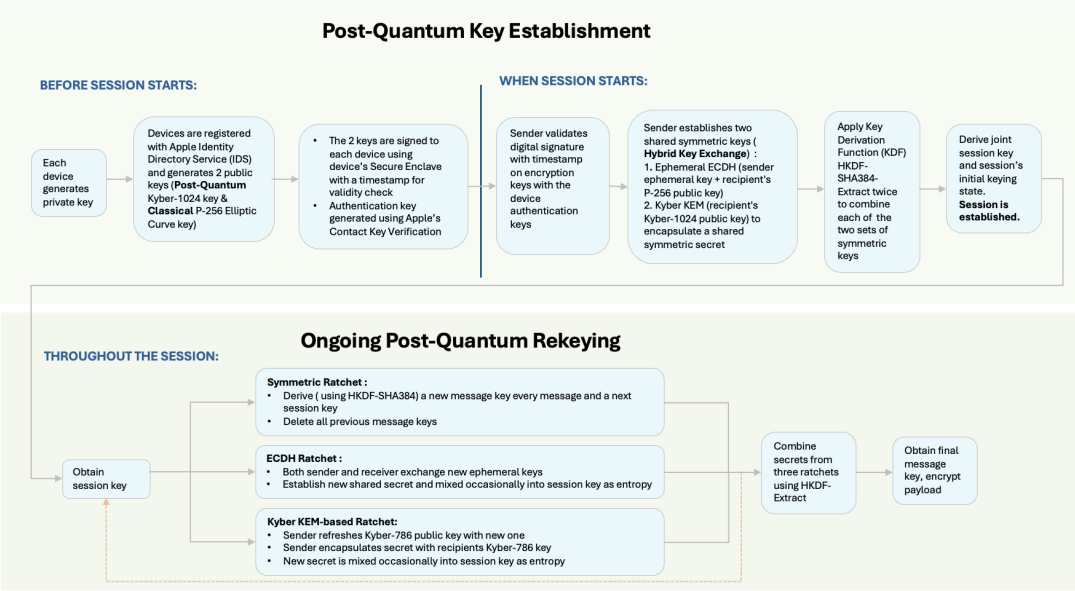


Figure 6.3: An overview of Apple's PQ3 protocol [26]

Key exchange protocols have been widely used in asymmetric encryption, such as Diffie-Hellman-Merkle (DHMS) Key Exchange Scheme for TLS, SSH and IKE network protocols (cite this). KEX methods include Key Encapsulation mechanisms and Key Agreement methods [16]. In PQ3, a post-quantum Kyber-1024 key encapsulation public key as well as a classical P-256 Elliptic Curve key agreement public key are used together in the encryption process. Prior to a session establishment, each device generates its own private keys. This allows for the generation of two public keys, Kyber-1024 and P-256 which are registered to the device using Apple Identity Directory Service (IDS). Each device's public encryption keys are signed

using the device's Secure Enclave, and generates an authentication key using Apple's Contact Key Verification. When the session starts, the sender validates the recipient's digital signature using the device authentication key and timestamp, to make sure that the recipient's public authentication keys are still valid. The sender's device can then use the two public encryption keys to share two symmetric keys with the recipient. Two shared secrets are independently established as a result - a ECDH (P-256) secret using the sender's ephemeral key and the recipient's P-256 public key, and a Kyber (KEM) secret respectively In the Kyber KEM step, the sender encapsulates a symmetric secret using the recipient's Kyber-1024 public key. This forms a Hybrid KEX protocol that makes used of two registered public keys per device to compute two shared secrets instead of one. Moreover, the hybrid combination of pre-quantum keys and post-quantum keys ensures that the key exchange is never susceptible to only either one kind of attack, and any potential attacker has to decrpyt both keys. The use of ECDH ephemeral keys ensures forward secrecy, as an exposure of a encryption key would not be able to allow the attacker to decrypt past messages using this key [10]. The two keys from both devices are then also further combined using a key derivation function (KDF), which is a well-established cryptographic method for producing a joint session key. This is carried out by invoking HKDF-SHA384-Extract twice with an expansion step afterwards [26], generating more entropy and increasing the difficulty of recovering the secret.

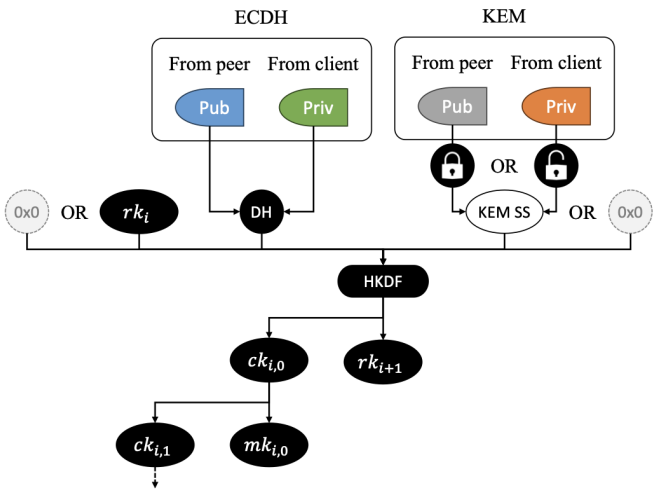


Figure 6.4: Key establishment in PQ3 [26]

PQ3 then combines three ratchets to achieve post-quantum encryption. Ratcheting is defined as a cryptographic technique where encryption keys are constantly refreshed, and deleted once they are no longer required for encryption or decryption [4]. It delivers the basic security properties - forward security, post-compromise security, and "immediate (no-delay) decryption", all of which had never been achieved single-handedly by any prior messaging protocols [1]. After the session is established, both parties use firstly a symmetric ratchet to derive a new message key for each message, which will be then used to encrypt and decrypt a message. After

which, the chain key derived from the shared secret will be then ratchet forward. This is a process that usually involves an update to the chain key using a hash function or a KDF. Upon decryption, the message key is immediately deleted to protect forward secrecy, and will never be accessed or utilized again. In addition to a symmetric ratchet, a second ECDH (Elliptic Curve Diffie-Hellman) ratchet is used to further introduce fresh entropy. Both parties generate an ephemeral keypair to carry out a Diffie-Hellman key exchange to generate another secret. This secret is then mixed into the ongoing session key, and the ECDH ratcheting repeats several times in a conversation. On top of these two ratcheting mechanisms, a post-quantum Kyber Ratchet is periodically introduced to provide quantum-secure entropy into the system as well. The ratchet is initiated by Alice using Bob's public key to encapsulate a shared secret and sends a ciphertext to Bob. Bob can now decapsulate the ciphertext using their Kyber private key, instead of a public key. The derived shared secret is then mixed into the ongoing session key material, using a HKDF. After these three ratcheting mechanisms, the resulting output is a message key which is ready to be used for payload encryption. The key update process strengthens future secrecy [3] and also provides self-healing properties in the face of an attack - even if an attacker compromises part of the session or obtains an encryption key, future keys can recover their security guarantees without requiring a complete reset as the session key has evolved in the future.

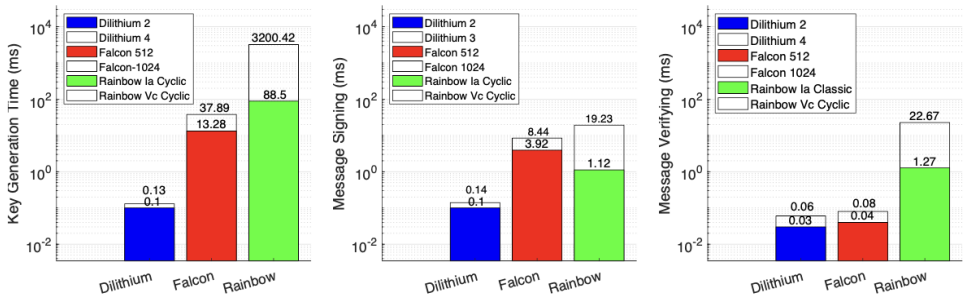
6.4.3 Architectural Considerations

When selecting a post-quantum cryptographic solution, it is crucial to consider key architectural factors. To facilitate the increase in security levels in PQC, key sizes generally need to be larger to maintain resistance against quantum attacks. This includes the significantly larger sizes of encryption keys and digital signatures compared to traditional systems [21]. Additionally, accurate estimation of encryption and decryption times, as well as the anticipated data traffic over communication channels, is essential to ensure system efficiency and scalability. Among PQC schemes, isogeny-based cryptosystems [9] offer the smallest key sizes at practical security levels, while code-based systems like McEliece are known for very large keys, though some variants (e.g., using the Lee metric [20]) aim to reduce this size but may be limited by vulnerabilities to lattice-based attacks. With that said, lattice-based cryptography is still a leading PQC candidate [33]. Latest research directions are focusing on designing schemes with smaller key sizes that still provide strong quantum resistance, balancing storage, communication, and computational overheads [45].

Larger key sizes in PQC can impact performance, including slower execution times, increased storage requirements, and higher communication costs [15]. These factors are critical when selecting algorithms for real-world applications, especially for resource-constrained environments like IoT devices or vehicular communications [45].

Moreover, performance and security of PQC algorithms affect computational efficiency and compatibility across platforms. Farooq *et al.* (2023) [15] evaluated two NIST fourth-round finalists - Classic McEliece and BIKE, using the liboqs library on both Linux and Windows systems. Classic McEliece supports high security levels, including NIST Level 5, but its efficiency is limited by very large public keys, such as 1.36 MB for the McEliece-8192128 variant, and long key generation times, reaching up to 298 seconds on Windows. In comparison, BIKE offers much faster performance and smaller key sizes. For example, the BIKE-L1 variant uses a public key of only 1541 bytes, generates keys in 126.95 microseconds, and performs encapsulation in just 17.55 microseconds on Linux. Decapsulation times also show a similar contrast: Classic McEliece-348864 decapsulates in 33.08 microseconds, while BIKE-L3 requires 993.16 microseconds. However, BIKE's implementation is currently limited to Linux, as it is not supported by default in the Windows version of liboqs. These results highlight a clear trade-off: algorithms like McEliece offer stronger long-term security but demand more resources, whereas BIKE is more efficient and lightweight, but with less extensive platform support. Therefore, choosing an appropriate PQC algorithm involves balancing security needs with performance constraints, especially in applications such as TLS or resource-constrained environments.

A study done by Raavi *et al.* [39] demonstrates clear relationship between signature size, algorithmic efficiency, and security in post-quantum digital signature schemes. By comparing NIST finalist digital signature schemes - Dilithium, Falcon, and Rainbow, the study [39] showed that smaller signature sizes contribute directly to lower communication overhead and faster execution, making such schemes more viable for real-world use. For instance, Falcon achieves strong security at a signature size of just 690 bytes, compared to Dilithium's 2044 bytes, yet outperforms it in signing and verification time.



(a) Key generation time (b) Message signing time (c) Message verifying time

Figure 6.5: Execution time (in milliseconds) for key generation, signing, and verification across post-quantum signature algorithms. Both the fastest and slowest variants from each algorithm family are shown, tested using 100-byte messages [39].

Security comparisons were then evaluated using advanced quantum-resilient metrics, including qubit cost, quantum gate cost, and the Depth-Width (DW) cost which was introduced to account for both resources simultaneously.

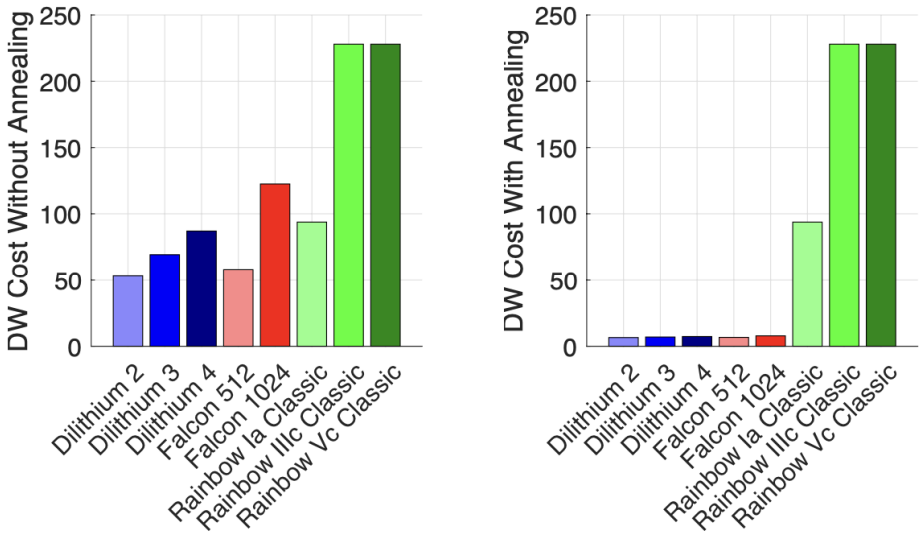


Figure 6.6: Estimated quantum attack cost (DW metric) for each signature algorithm, comparing scenarios with and without quantum annealing. Higher values indicate stronger resistance to quantum attacks [39].

Falcon offers higher classical and quantum security per byte than Dilithium, while Rainbow - despite its smaller signatures, achieves the highest overall security cost but suffers from impractically large certificate sizes, which prevented its integration with TLS 1.3 due to packet size limitations. These findings demonstrate that an optimal signature scheme must balance compact signature length while maintaining robust quantum resistance, as quantified by DW and gate-level metrics, to achieve both high security and practical deployment efficiency.

6.4.4 Future Directions

While post-quantum algorithms offer a higher level of security from mathematical complexity, it is still prone to side channel attacks [44] that bypass mathematical computation. Side channel attacks exploit physical information generated from the execution of the post-quantum algorithm, such as power consumption [23], timing, or electromagnetic emissions to extract secret keys or information [49] through studying the correlation between these factors. This suggests that it is important for post-quantum cryptography protocols to not only be mathematically secure, but also physically secure in infrastructure to prevent such vulnerabilities.

It is also recommended by recent research that striving for a full post-quantum transition might not be the best solution in the near future. Instead, hybrid cryptographic schemes that utilize more than one of either classical, quantum and post-quantum cryptography algorithms facilitate a smooth transition to a post-quantum future [42]. This approach leverages the time-tested security guarantee of existing, well-performing classical cryptography algorithms, while still building

post-quantum resistance, ensuring security even if one scheme is broken. In addition to using classical post-quantum cryptography algorithms that rely on computational difficulty, incorporating Quantum Key Distribution (QKD) [27] promises information-theoretic security independent of computation. Its security is based on laws of quantum physics, with the effect that any eavesdropping of the quantum channels will be easily detected. The resistance of the triple combination specifically allows for resistance to all three types of attack, and until one of the key generation methods remains unbroken,the 3-key Combiner is IND-CCA secure [42]. Such protocols will further strengthen the security of an cryptography system , compared to beyond merely using a single classical or post-quantum cryptography protocol. While post-quantum cryptography has yet to be proven entirely impenetrable, implementing hybrid schemes offers a practical and backward-compatible solution for modern communication systems.

6.5 Conclusion

Post-Quantum Cryptography is a rapidly evolving field, with significant progress in recent years. Thanks to the standardization efforts led by NIST, some PQC algorithms are already finding significant practical applications. For instance, CRYSTALS-Kyber is already encrypting iMessage traffic. However, the number of real-world implementations of PQC remains small. While the quantum-threat is hypothetical, it also affects the present, due to *"harvest-now-decrypt-later"* attacks. This unusual dichotomy is likely a contributing factor to the slow adoption of PQC. It remains to be seen which organizations will choose to invest in the mitigation of a risk that has not yet fully materialized.

Due to their novelty, PQC algorithms are still under scrutiny. While classical methods have benefited from decades of analysis, new standards require further validation. Hybrid approaches, combining post-quantum and classical encryption, provide a workable solution that guarantees at least the security of traditional methods. Considering the additional architectural complexity and resource requirements, pure PQC solutions may become more attractive in the long-term.

As discussed, replacing classical public-key encryption requires exploring new mathematical foundations. However, security cannot solely rely on the hardness of underlying mathematical problems, but includes architectural concerns, physical hardware and many other areas. Side channel attacks in PQC are an example that shows just how many considerations must be made to achieve holistic security. In this sense, security cannot be reduced to a hard problem such as MWLE, but must be evaluated in a much broader context.

Furthermore, architectural challenges such as larger key sizes, computational overhead and backward compatibility still pose barriers. Therefore, real-world implementations of PQC algorithms are essential, as they can inform future approaches and contribute to creating an interoperable, quantum-resistant communication network.

References

[1] Joël Alwen, Sandro Coretti, and Yevgeniy Dodis. *The Double Ratchet: Security Notions, Proofs, and Modularization for the Signal Protocol*. Cryptology ePrint Archive, Paper 2018/1037. 2018. URL: %5Curl%7Bhttps://eprint.iacr.org/2018/1037%7D.

[2] Ştefania Andrieş et al. *A survey on the security protocols employed by mobile messaging applications*. Cryptology ePrint Archive, Paper 2022/088. 2022. URL: %5Curl%7Bhttps://eprint.iacr.org/2022/088%7D.

[3] Daniele Antonioli. "BLUFFS: Bluetooth Forward and Future Secrecy Attacks and Defenses". In: CCS '23. Copenhagen, Denmark: Association for Computing Machinery, 2023, pp. 636–650. DOI: 10.1145/3576915.3623066.

[4] M. Bellare et al. "Ratcheted Encryption and Key Exchange: The Security of Messaging". In: (2017), pp. 619–650. DOI: 10.1007/978-3-319-63697-9_21.

[5] Daniel J Bernstein and Tanja Lange. "Post-quantum cryptography". In: *Nature* 549.7671 (2017), pp. 188–194.

[6] Daniel J Bernstein et al. "SPHINCS: practical stateless hash-based signatures". In: *Annual international conference on the theory and applications of cryptographic techniques*. Springer. 2015, pp. 368–397.

[7] Daniel J Bernstein et al. "The SPHINCS+ signature framework". In: *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*. 2019, pp. 2129–2146.

[8] Julia Bobrysheva and Sergey Zapechnikov. "Post-Quantum Security of Communication and Messaging Protocols: Achievements, Challenges and New Perspectives". In: *2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*. 2019, pp. 1803–1806. DOI: 10.1109/EIConRus.2019.8657136.

[9] Fábio Borges, Paulo Ricardo Reis, and Diogo Pereira. "A Comparison of Security and its Performance for Key Agreements in Post-Quantum Cryptography". In: *IEEE Access* 8 (2020), pp. 142413–142422. DOI: 10.1109/ACCESS.2020.3013250.

[10] R. Canetti, S. Halevi, and Jonathan Katz. "A Forward-Secure Public-Key Encryption Scheme". In: *Journal of Cryptology* 20 (2003), pp. 265–294. DOI: 10.1007/s00145-006-0442-5.

[11] Lily Chen et al. *Report on Post-Quantum Cryptography*. Tech. rep. NISTIR 8105. National Institute of Standards and Technology, 2016. DOI: 10.6028/NIST.IR.8105.

[12] Cloudflare, Inc. *What is End-to-End Encryption (E2EE)?* Accessed: 2025-05-13. n.d. URL: %5Curl%7Bhttps://www.cloudflare.com/learning/privacy/what-is-end-to-end-encryption/%7D.

[13] Whitfield Diffie and Martin E Hellman. "New directions in cryptography". In: *Democratizing Cryptography: The Work of Whitfield Diffie and Martin Hellman*. 2022, pp. 365–390.

[14] Morris J Dworkin et al. "Advanced encryption standard (AES)". In: (2001).

[15] Sana Farooq et al. "Resilience Optimization of Post-Quantum Cryptography Key Encapsulation Algorithms". In: *Sensors (Basel, Switzerland)* 23 (2023). DOI: 10.3390/s23125379.

[16] Alvaro A. Giron, Ricardo Custodio, and Francisco Rodríguez-Henríquez. "Post-quantum hybrid key exchange: a systematic mapping study". In: *Journal of Cryptographic Engineering* 13 (2023), pp. 71–88. DOI: 10.1007/s13389-022-00288-9.

[17] Google Cloud. *Why Google now uses post-quantum cryptography for internal comms*. Accessed: 2025-05-13. 2023. URL: %5Curl%7Bhttps://cloud.google.com/blog/products/identity-security/why-google-now-uses-post-quantum-cryptography-for-internal-comms%7D.

[18] Lov K Grover. "A fast quantum mechanical algorithm for database search". In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 1996, pp. 212–219.

[19] Matthias Homeister. *Quantum Computing Verstehen*. Springer Fachmedien Wiesbaden, 2015.

[20] Anna-Lena Horlemann et al. *Lattice-Based Vulnerabilities in Lee Metric Post-Quantum Cryptosystems*. 2024. arXiv: 2409.16018 [cs.CR]. URL: %5Curl%7Bhttps://arxiv.org/abs/2409.16018%7D.

[21] Jency Rubia J et al. "A Survey about Post Quantum Cryptography Methods". In: *EAI Endorsed Transactions on Internet of Things* (2024). DOI: 10.4108/eetiot.5099.

[22] Neal Koblitz. "Elliptic curve cryptosystems". In: *Mathematics of computation* 48.177 (1987), pp. 203–209.

[23] Kristjane Koleci et al. "A Side Channel Attack Methodology Applied to Code-Based Post Quantum Cryptography". In: (2022), pp. 90–96. DOI: 10.1007/978-3-031-30333-3_12.

[24] S. Kumari et al. "Post-quantum cryptography techniques for secure communication in resource-constrained Internet of Things devices: A comprehensive survey". In: *Software: Practice and Experience* 52.10 (2022), pp. 2047–2076. DOI: 10.1002/spe.3121.

[25] Hee-Yong Kwon, Indra Bajuna, and Mun-Kyu Lee. "Compact Hybrid Signature for Secure Transition to Post-Quantum Era". In: *IEEE Access* 12 (2024), pp. 39417–39429. DOI: 10.1109/ACCESS.2024.3374645.

[26] Felix Linker, Ralf Sasse, and David Basin. *A Formal Analysis of Apple's iMessage PQ3 Protocol*. Cryptology ePrint Archive, Paper 2024/1395. 2024. URL: %5Curl%7Bhttps://eprint.iacr.org/2024/1395%7D.

[27] Miralem Mehic et al. "Quantum Cryptography in 5G Networks: A Comprehensive Overview". In: *IEEE Communications Surveys & Tutorials* 26.1 (2024), pp. 302–346. DOI: 10.1109/COMST.2023.3309051.

[28] A Menezes, P van Oorschot, and S Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[29] Daniele Micciancio and Oded Regev. "Lattice-based cryptography". In: *Post-quantum cryptography*. Springer, 2009, pp. 147–191.

[30] Victor S Miller. "Use of elliptic curves in cryptography". In: *Conference on the theory and application of cryptographic techniques*. Springer. 1985, pp. 417–426.

[31] Mohamed Nabeel. "The Many Faces of End-to-End Encryption and Their Security Analysis". In: *2017 IEEE International Conference on Edge Computing (EDGE)*. 2017, pp. 252–259. DOI: 10.1109/IEEE.EDGE.2017.47.

[32] National Institute of Standards and Technology. *Module-Lattice-Based Digital Signature Standard*. Tech. rep. 204. National Institute of Standards and Technology (NIST), 2024.

[33] National Institute of Standards and Technology. *Module-Lattice-Based Key-Encapsulation Mechanism Standard*. Tech. rep. 203. National Institute of Standards and Technology (NIST), 2024.

[34] National Institute of Standards and Technology. *Stateless Hash-Based Digital Signature Standard*. Tech. rep. 205. National Institute of Standards and Technology (NIST), 2024.

[35] National Institute of Standards and Technology (NIST). *Post-Quantum Cryptography Standardization: Round 1 Submissions*. Accessed May 13, 2025. 2017. URL: %5Curl%7Bhttps://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-1-submissions%7D.

[36] Bill Newhouse and Andrew Regenscheid. *NIST PQC Crypto Update*. Presented at the 2025 PQC Conference, Austin, TX. 2025. URL: %5Curl%7Bhttps://pkic.org/events/2025/pqc-conference-austin-us/WED_PLENARY_1000_Bill-N_Andrew-R_NIST-PQ-Crypto-Update.pdf%7D.

[37] Abayomi Titilola Olutimehin et al. "Future-Proofing Data: Assessing the Feasibility of Post-Quantum Cryptographic Algorithms to Mitigate 'Harvest Now, Decrypt Later' Attacks". In: *Archives of Current Research International* 25.3 (Feb. 2025), pp. 60–80. DOI: 10.9734/acri/2025/v25i31098.

[38] Filip Opika et al. "Performance Analysis of Post-Quantum Cryptography Algorithms for Digital Signature". In: *Applied Sciences* (2024). DOI: 10.3390/app14124994.

[39] Manohar Raavi et al. "Security Comparisons and Performance Analyses of Post-quantum Signature Algorithms". In: June 2021, pp. 424–447. DOI: 10.1007/978-3-030-78375-4_17.

[40] Oded Regev. "On lattices, learning with errors, random linear codes, and cryptography". In: *Journal of the ACM (JACM)* 56.6 (2009), pp. 1–40.

[41] Oded Regev. "The learning with errors problem". In: *Invited survey in CCC* 7.30 (2010), p. 11.

[42] Sara Ricci et al. "Hybrid Keys in Practice: Combining Classical, Quantum and Post-Quantum Cryptography". In: *IEEE Access* 12 (2024), pp. 23206–23219. DOI: 10.1109/ACCESS.2024.3364520.

[43] Ronald L Rivest, Adi Shamir, and Leonard Adleman. "A method for obtaining digital signatures and public-key cryptosystems". In: *Communications of the ACM* 21.2 (1978), pp. 120–126.

[44] Ari Shaller, Linir Zamir, and Mehrdad Nojoumian. "Roadmap of post-quantum cryptography standardization: Side-channel attacks and countermeasures". In: *Inf. Comput.* 295 (2023), p. 105112. DOI: 10.1016/j.ic.2023.105112.

[45] Kyung-Ah Shim. "A Survey on Post-Quantum Public-Key Signature Schemes for Secure Vehicular Communications". In: *IEEE Transactions on Intelligent*

Transportation Systems 23 (2022), pp. 14025–14042. DOI: 10.1109/TITS.2021.3131668.

[46] Peter W Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th annual symposium on foundations of computer science*. Ieee. 1994, pp. 124–134.

[47] Harshdeep Singh. *Code based Cryptography: Classic McEliece*. 2020. arXiv: 1907.12754 [cs.CR]. URL: %5Curl%7Bhttps://arxiv.org/abs/1907.12754%7D.

[48] Michael Sipser. *Introduction to the Theory of Computation*. 3rd. Cengage Learning, 2012.

[49] Tristen Teague et al. "Towards Cloud-based Infrastructure for Post-Quantum Cryptography Side-channel Attack Analysis". In: *2023 IEEE Design Methodologies Conference (DMC)*. 2023, pp. 1–6. DOI: 10.1109/DMC58182.2023.10412485.

[50] Sean Turner. "Transport layer security". In: *IEEE Internet Computing* 18.6 (2014), pp. 60–63.

Chapter 7

Overview of the AI Agent Systems and Its Protocols

Erik Avtandilyan, Lukas Sager

Large-language models (LLMs) have made the long-standing vision of autonomous software agents and humanoid robots a reality. This paper surveys the spectrum of AI Agents, beginning with Simple Reflex Agents to advanced Agents working together in a Multi-Agent System. After outlining the capabilities and limitations of each architecture, this paper takes a closer look at what is needed for Multi-Agent Systems to effectively communicate with each other and what the structure and semantics of established communication protocols look like. Then, how the scope and autonomy of contemporary agents is extended through the role of Large-language models. The research is supplemented with a review of case studies, one looking into a Single Agent System and the other examining a Multi-Agent System. It is then concluded with looking at the limitations of today's Agents like their fragile reliability, latency issues, and privacy and security concerns, as well as a reflection on where we currently are in the still young age of AI Agents.

Contents

7.1 Introduction 149
7.1.1 Evolution of AI 149
7.2 Fundamentals of AI Agents and Multi-Agent Systems 150
7.2.1 Definition and Characteristics of AI Agents 150
7.2.2 Types of AI Agents 150
7.2.2.1 Types of Agents by Architecture 151
7.2.2.2 Types of Agents by Capabilities 152
7.2.3 Introduction to Multi-Agent Systems (MAS) 153
7.2.4 Interaction Models in MAS 154
7.3 Communication Protocols in Multi-Agent Systems 155
7.3.1 The Need for Communication Protocols 155
7.3.2 Inter-Agent Protocols 156
7.3.2.1 FIPA-ACL 156
7.3.2.2 KQML 157
7.3.3 Intra-Agent Protocol 158
7.3.4 Extra-Agent Protocols 159
7.3.5 Analysis of Communication Protocols 160
7.3.6 Extending Communication Protocols 160
7.4 Integrating Large Language Models into AI Agents 161
7.4.1 Role of LLMs in Enhancing AI Agents 161
7.4.2 Techniques for LLM Integration 162
7.5 Case Studies 164
7.5.1 Single-Agent: Devin.ai 164
7.5.2 Multi-Agent System: Figure AI 165
7.6 Challenges and Future Directions 166
7.6.1 Technical Challenges 166
7.6.2 Scaling Secure and Interoperable Agent Societies 167
7.7 Conclusion 167

7.1 Introduction

7.1.1 Evolution of AI

Although the idea of Artificial Intelligence (AI) has ancient roots, with Aristotle's development of syllogisms and the "laws of thought" to formalize logical reasoning, major contributions from psychology, philosophy, mathematics, neuroscience, and economics have laid the foundations for modern intelligent agents. A groundbreaking milestone was reached in 1943 when the mathematician Walter Pitts and the neurologist Warren McCulloch proposed a model of artificial neurons connected through logical rules, conceptually aligned with Turing's ideas on computation. Independently, Donald Hebb introduced a learning theory, now known as Hebbian learning, which remains influential: it says that the connection between two neurons is strengthened when they are activated simultaneously [23, 19].

Through those concepts, enthusiasm for the field of Artificial Intelligence grew. In 1956 a two-month workshop was held at Dartmouth College *"to find how to make machines use language, form abstractions and concepts, solve kind of problems now reserved for humans, and improve themselves"*[19]. What is to some extent possible today was unsuccessful then. In the coming years, further progress was made with game programs based on reinforced learning for popular games like checkers, chess and backgammon. So-called microworlds were created that were able to solve limited problems such as calculus integration or geometric analogy problems. With further advancements in the field eventually Boolean logic was extended by probabilistic reasoning and machine learning in the mid 80's. In the new century, new opportunities arose in the field of AI due to the advances in computing power, as well as the rise of the Internet created gigantic data sets, known as big data. The focus shifted from improving the algorithm to the use of big data and machine learning, after the realization that it led to much larger advancements in the field. Those advancements and their proven capabilities resparked interest in the field for both academia and business. Through deep learning, classification in image recognition improved from a 28% error rate in 2010 to a 2% error rate in 2017 and therefore exceeded human image recognition. In the coming years, large technology companies made AI widely available through their features in smart phones and computers. The age of the Agents started in 2022 with the launch of ChatGPT by OpenAI which made an API-based generative model available to the public. Through the large-language model (LLM), its use was made possible to any user and could be implemented in an almost unlimited number of ways. The AI world is now in the middle of a race. Since then, many new models have been launched from different companies and countries all over the world with continuous improvement and new purposes[19, 23].

7.2 Fundamentals of AI Agents and Multi-Agent Systems

7.2.1 Definition and Characteristics of AI Agents

The term Agent is described by [34] as *"an element or an entity having the power to act on behalf of another."* In the technical context, Agents are systems that are *"designed to interact with users using natural language"* [11]. So the term Agent includes both human agents as well as the discussed AI Agents. The distinction between the two is, that the AI Agent merely tries to act like a human agent. The similarities are in the interaction and the purpose:

The AI is powered through Large Language Models (LLMs) as a way to interact with it. This means that an AI Agent, similar to a human agent, can receive information in natural language. This information helps the Agents interact with their environment. In a real world setting, this could be an order to a Real Estate Agent to put up a house for sale or a Web Agent to adjust the design on a website. For the AI Agent it is the same, they receive information and take actions based on the input and context given to them. AI Agents have underlying data they are trained on for a specific use. In the example of ChatGPT, this use can be general. Others might be focusing on language learning, physics, information systems or other topics. [14]

The purpose of an AI Agent is, that the user can provide information of the task at hand in the form of natural language to the AI Agent. The AI is able to interpret the natural language given by the user and execute the desired action in its defined context. [14]

The characteristics of an AI Agent are, that with the fast advancement in language modeling, the AI Agent can not only understand what is being said, but also build reasoning upon that. Therefore, natural language that is conversational and non-technical can be interpreted and applied for the purpose of the Agent. The last characteristic of the agent is, that it can, based on the input and reasoning, act upon it instead of just responding [14].

7.2.2 Types of AI Agents

In [19] the agents are built on the sum of a program (or capability) and its architecture. The AI has the responsibility to design a program to represent the agent function. The architecture is the resource on which it can rely to realize its function. We get this simple equation: *"agent = architecture + program"* [19].

7.2.2.1 Types of Agents by Architecture

Furthermore, [19] defines five different types of agents, that cover most intelligent systems:

The Simple Reflex Agent makes decisions based on current perception and does not take historical data into consideration. They are usually based on simple condition-action rules. A common example would be a cleaning robot. If it detects dirt, then it will clean. It does not matter what happened before. The same goes for its path. If it detects a wall, then it turns right, even if it has been there before. Its program is based on the simple if-then rules, and the architecture is defined through its sensors and actors [19].

Model-based Reflex Agents are from a functional perspective similar to the Simple Reflex Agents. The difference is, that they maintain a model that helps understand the world around them and to fullfil their purpose. This model consists of two components. The transition model describes how the world works in the response to the agents actions. In the example of a autonomous car, if the wheel is turned clockwise, the car will turn to the right. The second component, the sensor model, exists to track changes in the environment to consider them in their actions. In the same example this can be a sensor for temperature to track if roads are frozen and the speed needs to be adjusted [19].

Goal-based Agents also consider a model to describe and react to their environment. If a car's target destination changes, a Reflex Agent would need to have their model adjusted for the new roads taken. For the Goal-based Agent, the rules are explicit for the previous scenario and can change towards the new goal. Reflex Agents consider an action because they have been told to react this way. Goal-based Agents make decisions based on their goals and are more flexible. Their ability to search and plan ahead helps to achieve those goals [19].

Similar to what was observed in the previous agents, the **Utility-based Agents** adds another dimension. So far there are I) the ability to react to the environment, II) to model the environment, and III) to adjust the model towards a goal. The ability to rationalize and make decisions for a certain utility to reach the goal is added. To reach a destination, there are multiple ways to achieve this goal, those ways are considered utility. Depending on what is prioritized, the agent can take the fastest road or the environment-friendly route. The goal is therefore extended to how it should be reached or what utility it should maximize [19].

The last agent type is the **Learning Agent**. It was first described by Touring in 1950. Instead of programming a machine capable to act as the four previous agents, he proposed a learning machine which then can be taught how to behave. Today, this is considered the dominant way to create any agent and their systems. While the performance element, which includes the capabilities described previously, is already given, the learning element is introduced. Through a critic and a problem generator, it can improve in connection with its defined performance element. The critic gives feedback to the agent on how to improve in the future. The problem

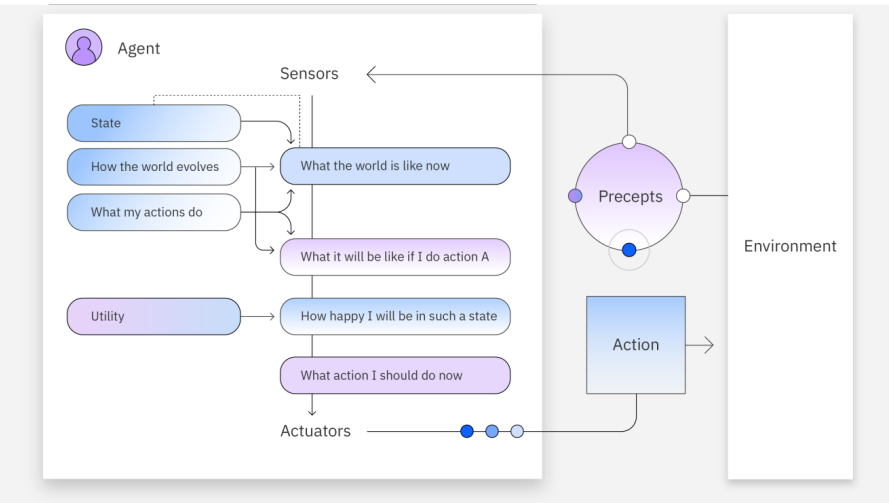


Figure 7.1: Architecture of Utility-based Agents [26]

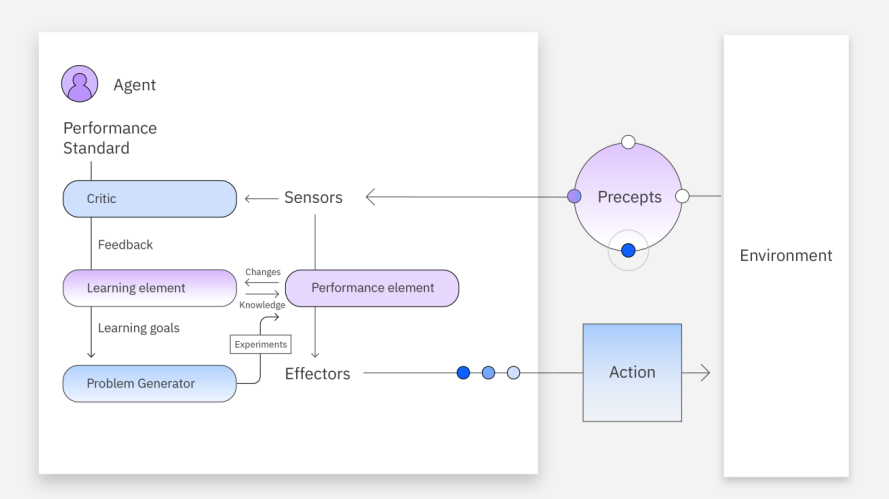


Figure 7.2: Architecture of Learning Agents [26]

generator is responsible for testing different scenarios to gain further information and experiences, even if they do not necessarily mean improvement but better understanding [19, 26].

7.2.2.2 Types of Agents by Capabilities

While there are different architectures for Intelligent Agents, their capabilities need to be considered as well. The capabilities describe what the agent can do. [19] includes the following:

The previous chapter has shown that there is the **Reactive Capability** of Agents. They are able to perceive their environment and act without deeper reasoning [19].

Problem-solving Capability, as its name describes, is to solve problems by planning ahead in atomic representations of their environment. Within these represen-

tations, they rely on search algorithms to reach their goal [19].

Logical Agents or Knowledge-based Agents have the **Capability of Reasoning**. Problem-solving Agents use their limited representation to act. They also often know all possibilities and their outcomes. Knowledge-based Agents are extended by general knowledge. Therefore they are able to reason on a decision through logic. They can approach problems differently and argue for a decision they make. An example would be finding the way back to the hotel in an unknown city. The Problem-solving Agent would know all the streets around it and decide based on the search algorithm, which route to take. The Knowledge-based Agent would perhaps know that the hotel lies on the west end of the city and therefore takes the street leading west [19].

The **Learning Capability**, often referred to as 'Machine Learning', is a capability of Learning Agents. The agent is given data sets to understand patterns and build a model based on the data received. Learning techniques include supervised, unsupervised and reinforcement learning. Deep Learning techniques are most widely used in modern applications. They rely on multilayered reasoning models called neural networks that represent the complexity of real-life scenarios more realistically [19].

Language models are capable of **Natural Language Processing**. They can communicate, learn, and understand through the interpretation of natural language. This means that it is able to translate the language spoken or written by humans, even if it is ambiguous or vague, to create its models. In this way, it can learn and improve to mimic the same language [19].

7.2.3 Introduction to Multi-Agent Systems (MAS)

Transitioning from the previous discussions on individual AI agents, it is crucial to explore how these agents interact within a larger collective framework. A Multi-Agent System (MAS) consists of multiple autonomous agents collaboratively interacting within a shared environment, aiming to accomplish complex goals that surpass individual capacities [12].

Fundamentally, MAS are designed to address problems characterized by distributed knowledge, dynamic conditions, and the necessity for coordinated action. To effectively manage such complexity, MAS leverage modular specialization by breaking down tasks into submodules, each addressed by agents specialized in distinct aspects of the overall problem. This approach is particularly suited to applications such as transportation networks, where geographic dispersion and continuously shifting components like vehicles and passengers demand sophisticated coordination strategies[17].

MAS distinguish themselves from single-agent systems by the depth of inter-agent interactions. Unlike single-agent scenarios, where agent-to-agent interactions are primarily tool-based and limited, MAS require agents to proactively model and

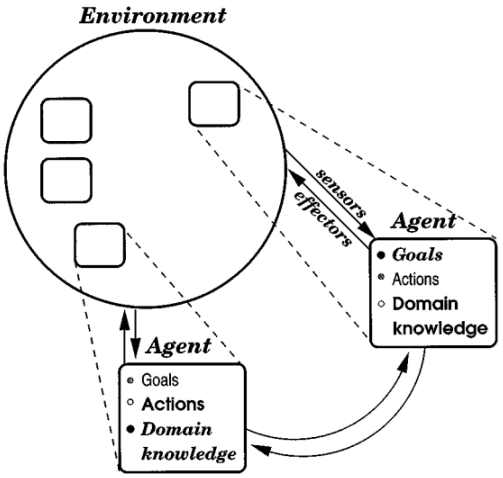


Figure 7.3: The Fully General Multiagent Scenario [25]

account for each other's goals, knowledge, and behaviors. Such intricate interactions facilitate collective decision-making and enhance adaptability, enabling MAS to dynamically respond to environmental changes by flexibly adjusting agent composition and roles[12].

7.2.4 Interaction Models in MAS

In recent developments within the artificial intelligence landscape, numerous startups have begun marketing their products as Multi-Agent Systems (MAS). However, upon closer examination, significant distinctions emerge between genuine MAS architectures and other forms of AI implementation. Fundamental to identifying these differences is the examination of how individual agents interact within such systems.

At its core, a Multi-Agent System represents a distributed framework consisting of multiple autonomous agents operating within a shared environment, where their individual decisions and actions inherently influence each other as well as the broader system. Understanding these agent interactions is critical, particularly when designing MAS capable of effectively addressing intricate real-world challenges across diverse domains, ranging from autonomous transportation systems to financial market operations [13].

Multi-Agent Systems are typically classified based on agent composition into two categories: homogeneous and heterogeneous systems. Homogeneous MAS comprise agents sharing identical objectives, actions, and domain knowledge. Conversely, heterogeneous MAS involve agents with varying goals, capabilities, and specialized knowledge, which introduces complexity into the system interactions [25].

A vital dimension in understanding MAS interactions lies in distinguishing between cooperative and competitive behaviors among agents. Cooperative agents willingly assist one another, even if individual objectives differ, thereby fostering a collaborative environment aimed at mutual benefit. In contrast, competitive agents prioritize their own objectives, potentially at the expense of other agents. In extreme scenarios, competitive interactions can manifest as zero-sum situations where agents actively impede others to advance their own goals [25].

Interestingly, theoretical analyses suggest that cooperation might occasionally emerge from behavior that appears irrational, as purely rational agents could exploit cooperative tendencies for personal advantage. Recognizing these behavioral intricacies is essential for effectively leveraging MAS to solve complex tasks [32].

7.3 Communication Protocols in Multi-Agent Systems

In Multi-Agent Systems, there are different setups that define the application. Agents work together on a common goal, the system function. Their roles in the communication can be cooperative, competitive, hierarchical, or a mix. Also, their planning can vary from centralized to decentralized planning and execution. With the Agent's need to communicate, an important role in creating efficient MAS lies on the communication protocols. A guide for the Agents to know how to communicate with each other [3].

Next to the inter-agent communication this section will also deal with examples of an internal protocol that facilitates the setup of a single Agent and also external facing protocols which improve the interoperability of Agents or MAS outside of their initial environment.

7.3.1 The Need for Communication Protocols

In the nature of communication lies ambiguity, the need for coherence of structure and semantics, and, due to many factors like culture, environment, and relationship, miscommunication. Those also lead to inefficiencies in reaching the system function. To mitigate those aspects in Agent communication, Communication Protocols are introduced in LLM-based Agents working in an MAS. They facilitate *"the timing, content and modality of interactions among agents"* [3].

In the Agents communication, there are four aspects that need to be considered:

- Message Semantics
- Message Syntax
- Communication/Interaction Protocol

Parameter	Category of Parameters
performative	Type of communicative acts
sender	Participant in communication
receiver	Participant in communication
reply-to	Participant in communication
content	Content of message
language	Description of Content
encoding	Description of Content
ontology	Description of Content
protocol	Control of conversation
conversation-id	Control of conversation
reply-with	Control of conversation
in-reply-to	Control of conversation
reply-by	Control of conversation

Figure 7.4: FIPA ACL Message Parameters [3]

- Transport Protocol

In LLM-based agents, the semantics of a message is given through the natural language used and partly also the syntax. The protocols therefore focus on the expression of messages and the structure of the communication between agents. The method of sending and receiving messages, the transport protocol, is not addressed in communication protocols and needs to be addressed based on the application's requirements [3].

7.3.2 Inter-Agent Protocols

When it comes to the protocols used in today's communication between Agents, two protocols have emerged as the standard; FIPA-ACL and KQML.

7.3.2.1 FIPA-ACL

To improve communication, one of the established protocols is the Foundation for Intelligent Physical Agents - Agent Communication Language (FIPA-ACL) [10]. The protocol ensures that the content, based on the Speech Acts Theory, is coherent in both structure and semantics to improve efficiency [3].

The ACL, established in 1996, is a structure on how Agents communicate. The content of a message contains most importantly a performative. The performative describes what the intent of a message is. This, for example, can be ”inform”, ”request”, or ”disagree”. Usually the performative is extended by further parameters like sender, receiver, and content [9].

The table in Figure 7.4 shows that with its modularity next to the mandatory performative, messages can become strongly structured and therefore provide additional information to a MAS conversation. At the same time, they can also be

```
(request
        :sender (agent-identifier :name alice@mydomain.com)
        :receiver (agent-identifier :name bob@yourdomain.com)
        :ontology travel-assistant
        :language FIPA-SL
        :protocol fipa-request
        :content
                ""((action
                    (agent-identifier :name bob@yourdomain.com)
                        (book-hotel :arrival 15/10/2006
                                    :departure 05/07/2002 ... )
                ))""
)
```

Figure 7.5: FIPA ACL Message Semantics [28]

purposely omitted. In the example of the sender parameter, it can be removed to ensure anonymity of the Agents.

A special focus lies on the parameter *protocol*. It dictates the structure to control the conversation and support the interpretation of messages. As an example, the initiating agent can use a protocol parameter designed for a request that only needs a simple accept or decline answer. The receiving agent then knows exactly what his output should look like and that there will be no follow-up messages. In this way, conversations are kept to a minimum, and all participating agents know their role in the conversation [3, 9]. What a technical representation of a message in the FIPA ACL could look like is shown in Figure 7.5 [28].

7.3.2.2 KQML

The Knowledge Query and Manipulation Language (KQML) was introduced in 1990 as an effort to *"develop techniques, methodologies, and software tools for knowledge sharing"* [28]. It is build in three layers:

- The content layer contains the content of the message. There is no defined format, in what way content must be provided. This is up to the agents to decide on structure and language used.
- The communication layer defines parameters such as sender, receiver and unique identifiers. It ensures an efficient communication and clear involvement of the necessary agents.
- The message layer contains two parts, the content message, which includes a performative, as described in ACL, and a description of the intent of the message. Further there is a declarative message that agents can use to declare information about themselves to other agents, like their existence, purpose, or meta information [28].

```
(ask-one
:sender joe
:content (PRICE IBM ?price)
:receiver stock-server
:reply-with ibm-stock
:language LPROLOG
   :ontology NYSE-TICKS)
```

Figure 7.6: Example of a KQML Message [28]

Since KQML has no specified formal semantic model, it has been applied in various different forms. This allows more flexibility and adaptability to systems in specific use-cases. This has also created criticism of the protocol since it leads to more misunderandings and ambiguity compared to other protocols. The lack of semantic rules allows it to add information to the content, which are not neccessarily focused on the current communication but have the goal to share and develop the understanding of other interacting agents about itself [24].

7.3.3 Intra-Agent Protocol

The fast development of LLM's has led to many new Agents and agentic systems rising in academia and the industry. With this development, Agents need to fulfill more complex tasks and create the need to be increasingly reliant in successfully completing those tasks. However, with a multitude of Agents built on different standards, the interoperability, scalability and ability to collaborate is hindered. Therefore, a multitude of new protocols have emerged. In this and the following section 7.3.4 a selection of established and emerging protocols is discussed [35, 6].

The **Model-Context Protocol (MCP)** emerged in 2024 and has already become a standard due to its wide adaptation in Agents. Its use is within the Agent. LLMs were usually connected to its systems and data bases through an application or 'glue code'. While the code had to be maintained and ensure that everything is working as intended, the MCP provides a more elegant solution for an Agent's architecture to ensure continuity without much maintenance [6]. MCP introduces an intermediary component called the MCP Server between the LLM and the resources it uses. The MCP Server consists of:

- Protocol Layer: Defines how messages between the LLM and external systems are formatted and structured, using JSON-RPC 2.0 standards. It ensures requests and responses match correctly and follow established patterns.
- Transport Layer: Manages the actual transmission of messages between the LLM and the external system.

The MCP Server also enforces access rights, maintains the operational security and notifies in case of changes in the system capabilities. The advantage of this architecture is the modularity within the Agents to change, extend or reduce the

LLM's access to other and new systems [6]. The key components that allow those capabilities are:

- Tools: Active functionalities allowing the LLM to perform tasks by invoking external APIs or services.
- Resources: Passive data sets and information sources controlled by the application, providing content that the LLM uses.
- Prompts: Templates or predefined instructions managed by users or the system to ensure consistent, reusable, and concise interactions, reducing redundancy in communication.
- Sampling: A controlled selection mechanism managed by the MCP Server, determining how the LLM generates responses, and therefore overseeing and influencing the agent's output [6]

7.3.4 Extra-Agent Protocols

So far, Inter-Agent and Intra-Agent protocols have been discussed. Extra-Agent Protocols differ in that they are specifically designed to facilitate communication with external entities, such as other systems or Multi-Agent Systems (MAS), that lie beyond the initial setup or internal architecture of an agent.

The **Agent Protocol** is an example of such an Extra-Agent Protocol, specifically categorized as a System-Agent Interaction Protocol. Its primary goal is to simplify the integration of agents into external systems. The Agent Protocol is designed to be universally compatible, allowing any existing agent, regardless of its current framework, to adopt and integrate this standardized interaction method. It achieves this by exposing standardized endpoints with predefined response models. For instance, the endpoint "Create Agent Task" receives an input, typically a prompt, and returns a standardized response indicating the creation of a new task [1, 35].

A further protocol that emerged in 2025 is the **Agent-2-Agent (A2A) Protocol**. It is a peer-to-peer Protocol that facilitates the communication between two independent systems to work together and achieve their respective task. It was introduced by Google in collaboration with over 50 other software companies as a way to improve Agent interoperability and their ability to cooperate with each other [27]. This is how it works; the user initiates a task to the Client Agent. The Client Agent then analyses the intent and searches for a suitable Remote Agent that can help complete the task. The Remote Agent is found through its Agent Card, a way to identify its existence and capabilities. The two Agents then interact and exchange artifacts, and ultimately the result is returned to the user [6]. The A2A Protocol builds on existing API standards like HTTP, SSE and JSON-RPC which allows an easy integration on existing tech stacks. The protocol empowers any Agent, able to 'speak' A2A protocol to interact with other Agents that also adhere

to this protocol. Further the supported JSON-RPC 2.0 standard also supports A2A protocol to effectively deploy the capabilities of MCP [6, 27].

7.3.5 Analysis of Communication Protocols

When comparing the two protocols, KQML has no commitment in the use of the content layer. The involved agents decide themselves how they want to specify and structure the message's content. FIPA's ACL on the other hand has a more strict policy for the content and provides a structure for the transported content [28].

A further advantage of KQML is the possibility to send arbitrary, useful information in the message's content layer to help other agents improve their understanding of the agents in their environment as well as their capabilities. Meanwhile FIPA-ACL is more strict in what is contained in a message for the sake of clarity and structure [29].

While both protocols enhance the communication in MAS, there are slight differences. KQML leaves more freedom in communication and therefore provides a stronger possibility of development and improvement in Agent communication. This freedom is trading in the benefits of FIPA's ACL. The ACL has more structure and leaves less room to inform other Agents about capabilities and therefore learning about their environment is reduced for Agents in comparison with KQML. ACL in return gains a more efficient communication and a lower likeliness of ambiguity and misunderstandings. In conclusion, ACL has benefits that work well in a system where roles are clear and communication needs to be stable and efficient. KQML is more fitting in ecosystems that embrace innovation and development.

The capabilities of MCP allow the modularity of Agents and reduces maintenance while also addressing security and privacy concerns. Therefore, it also increases the attractiveness of building Agents in business and academic settings and enabling their application in a larger variety. Combined with the Agent Protocol, which enables integration through common standards into various systems, Agents or MAS further increase the use of Agents. The A2A protocol, which specifically emphasizes the co-existence with MCP and allows the collaboration with other Agent systems and is supported through large global software providers further drives the trend [6, 27, 1]. Through standardization of protocols, building, deploying and enabling Agents provides reduce risks for investing in AI technologies and driving progress.

7.3.6 Extending Communication Protocols

The described Protocols have a clear purpose of helping systems interact with each other. There are further elements in communication methods, that help MAS improve in performance. By extending them with Mediator Models, they can be

optimized to also provide value with the improvement of efficiency in communication. Extensive communication is costly both in time and in computing power. It is in some cases desirable to reduce communication to only what is relevant [3]. A recent statement by Sam Altman, CEO of OpenAI, is, that only saying please and thank you to the chatbot results in computing power costs of *"Tens of millions of dollars"* [5].

A further extension to communication is focused on mitigation of wrongly generated output. This can be done through a process called Chain-of-Verification. It reduces hallucinations by generating a draft and then fact-checking the draft response by breaking it down. Due to this process, the draft can be refined and a validated answer is published [3].

7.4 Integrating Large Language Models into AI Agents

7.4.1 Role of LLMs in Enhancing AI Agents

Large-language models (LLMs) now constitute the principal engine of contemporary natural-language technology. At their core they are Transformer networks that learn long-range statistical dependencies in sequential data, allowing them to capture subtle semantics and discourse structure . Scaling these architectures to hundreds of billions of parameters, epitomised by GPT-3's 175 billion weights, has enabled near-human text generation from a prompt of only a few tokens, though at the cost of substantial capital outlay, expert talent, and specialised compute infrastructure that remain prohibitive for most organisations . The energy footprint of inference alone can rival that of entire traditional software stacks, reinforcing the need for careful deployment planning [18].

When embedded in an agent loop, the generative capacity of an LLM translates into a new form of autonomy. With minimal instruction it can sustain multi-turn dialogue, compose previously unseen narratives, or reformulate objectives without step-by-step supervision, exhibiting a creativity that traditional symbolic or statistical systems lacked. Prompting techniques such as chain-of-thought reasoning further elicit logical inference, mathematical deduction, hierarchical planning, and even self-critique, supplying the internal deliberation an agent requires to decompose goals and revise them in real time. Practical deployments often pair the model with retrieval-augmented memory, vector databases that surface long-tail facts on demand, so that reasoning is grounded in up-to-date knowledge rather than the frozen weights of pre-training. Recent multimodal fusion methods extend perception beyond text to images and audio, enabling language-driven agents to ground their decisions in richer sensory evidence gathered from the environment [33].

Natural-language programmability radically lowers the barrier to specifying complex workflows. Procedures that once demanded hand-coded rules can now be

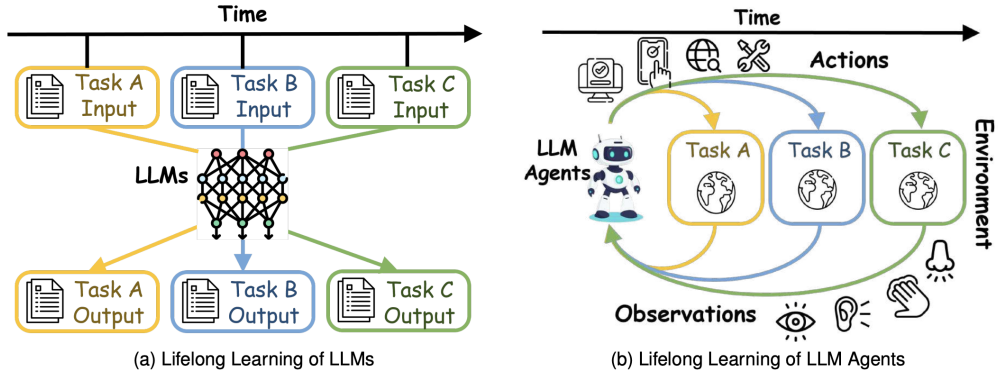


Figure 7.7: Comparison of lifelong learning between LLMs and LLM Agents [36]

expressed in plain language, allowing non-technical domain experts to prototype, audit, and refine agent behaviour directly. This shift accelerates iteration cycles, narrows the gap between subject-matter expertise and system logic, and democratises access to advanced AI tools across an organisation. Moreover, agents equipped with tool-calling capabilities can recognise when an external API, calculator, or database is required, invoke it autonomously, and fold the result back into their reasoning loop, expanding the effective action space far beyond text generation alone [31].

A critical limitation of conventional LLMs, however, is their static nature: once training ends, the model's knowledge and behaviour are frozen. Embedding the language model inside an LLM agent, complete with perception, memory, and action modules, opens the door to lifelong learning. Such agents accumulate episodic experience, update internal representations, and address the stability-plasticity dilemma: preserving mastered skills while remaining adaptable to novel tasks [31]. In practice this means an agent can navigate a videogame in the morning, assist with online shopping at noon, and orchestrate household robotics in the evening without catastrophic forgetting. Illustrations of these two learning paradigms are provided in Figure 7.7 [36].

7.4.2 Techniques for LLM Integration

In practical deployments an LLM-centered agent is usually organized around three tightly coupled subsystems. A perception layer converts raw signals from the outside world, such as numbers, images and speech, into embeddings the language model can understand. The brain layer then draws on stored knowledge and working memory to interpret those embeddings, reason about goals, and decide what should happen next. Finally, an action layer turns those abstract decisions into concrete operations by invoking external tools or APIs, thereby altering the environment and setting up the next round of observations. Because the agent constantly repeats this perceive-think-act loop, it can refine its behaviour over time and adjust to shifting conditions [33].

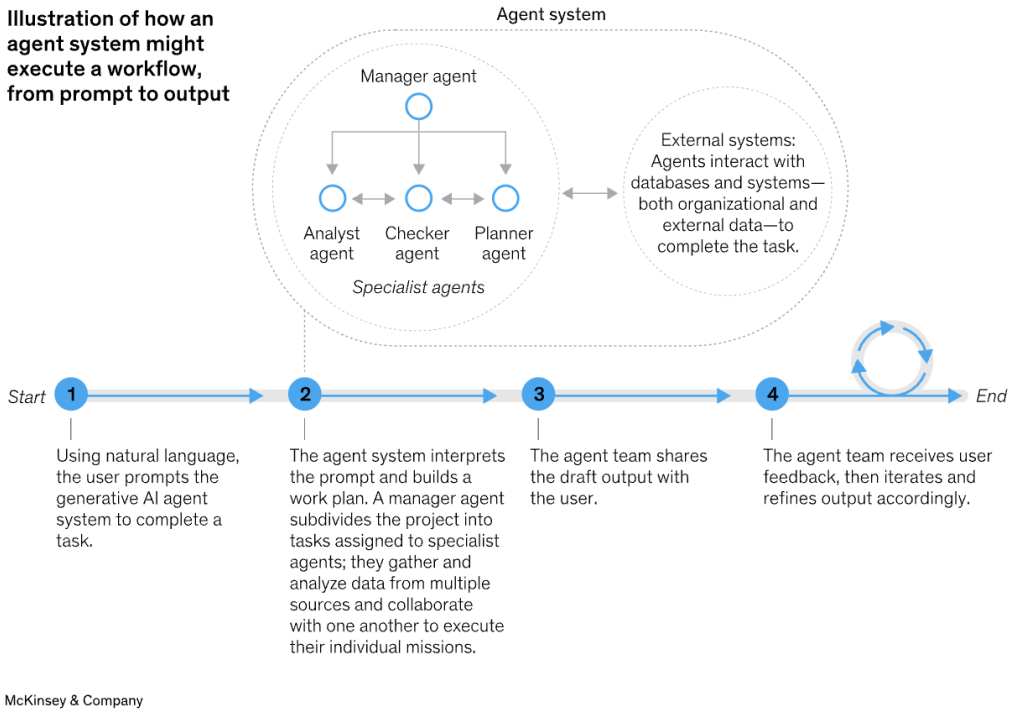


Figure 7.8: Illustration of agent system workflow, from prompt to output [4]

A useful way to see how these subsystems collaborate is to follow a complete end-to-end workflow. Figure 7.8 tracks the process from the user's initial prompt through to the final, polished result. At Step 1 the human issues a natural-language request; at Step 2 a manager agent decomposes that request into smaller jobs for specialist agents, each of which gathers data and proposes partial solutions; Step 3 presents a draft to the user; Step 4 incorporates feedback and iterates until the outcome meets the specification. The figure makes clear that direct human interaction happens only at the beginning and the end, while the internal stages are handled autonomously through repeated cycles of perception, deliberation, and action [4].

Deep reasoning inside the brain layer is often strengthened by chain-of-thought prompting and fine-tuning. Instead of forcing the model to jump straight to an answer, chain-of-thought techniques prompt it to articulate intermediate steps, breaking a complex problem into a transparent sequence of smaller inferences. Training or fine-tuning on such step-wise rationales produces agents that plan more coherently and whose decisions can be audited line by line before any real-world action is taken [33].

7.5 Case Studies

7.5.1 Single-Agent: Devin.ai

An example of an advanced AI Agent is the coding agent Devin by Cognition. Devin's purpose is to support software engineers in programming with remarkable capabilities. It takes instructions in natural language and end-to-end programs the task at hand. This includes writing functional code, which it will also debug, most likely through a Chain-of-Verification process. Further, it independently searches the web to find documentation for specific problems or tools and also accesses the integrated tools' terminals, code editors and browsers. Like a software engineer it can push code to GitHub and collaborate with other software engineers on a project [20].

Although it is uncertain, how reliable this or any similar Agent currently is, its impact is apparent. While the need for actual software engineers prevails, their increase in efficiency through the extention of such coding agents is remarkable. It can be applied for almost any case in their work; maintaining and updating software, debugging code, writing documentation and tests or creating full-stack web-applications. The ability to work autonomously is the emphasis on the Agent, where Devin does not only help with research and providing code but actually fully does the work and can present working software. An important mention in this particular AI Agent is the ability to self-diagnose his code and reasoning and improve the final code [20, 15].

Self-improving Coding Agents are the next development of Coding Agents. Other Agents have been implemented in certain tools and are more commonly used by software developers. The most commonly used Coding Agents are GitHub Copilot, JetBrain's AI Assistant and ChatGPT, which does not have this specific purpose but is often used according to [22]. The study also found that software engineers do not necessarily have the desire to delegate everything to the agent. More enjoyable tasks like implementing new features would rather not be delegated while undesirable tasks will. The most important aspect of the study is, why developers will not use Coding Agents at all. The reasons they gave were in over 22% of cases, the lack of need for them. Other popular reasons were inaccurate output, lack of trust or the feel of not being in control, the AI has a lack of understanding of the context, and the user's limited understanding of the technology [22].

It is apparent that new AI Agents like Devin.ai are to be used with caution. While arguments against them regarding the correctness, understanding of context and accountability can be made and have valid grounds. A soft factor, the position of the user, needs to be considered as well. Lack of understanding and trust in technology can hinder adaptation, as well as the belief that you do not need to rely on them to be efficient. Therefore can be concluded that altough Coding Agents are improving quickly with astonishing capabilities, their adaptation is not necessarily following immediately.

7.5.2 Multi-Agent System: Figure AI

The development of autonomous humanoid robots represents a significant advancement in artificial intelligence (AI) and robotics, particularly in the context of Multi-Agent Systems (MAS). Figure AI was founded in California in 2022 with the goal of building a general-purpose humanoid platform that operates through a proprietary vision–language–action architecture called Helix. The first two hardware iterations, Figure 01 and the more advanced Figure 02, each supply thirty-five actuated joints. Helix drives these joints at two hundred hertz, allowing the robot to perceive its surroundings, interpret spoken instructions, and perform fine-grained manipulation in spaces designed for people. Because a single Helix back-end can control multiple bodies, a group of Figure units behaves as a coherent multi-agent system rather than as isolated robots [8].

A public demonstration released in early 2025 illustrates this capability convincingly. Two Figure 02 units, both connected to the same Helix instance, emptied a bag of unfamiliar groceries into a kitchen cabinet without human tele-operation. One robot grasped each item while the other reorganised shelf space, and both continuously adjusted their trajectories to avoid collisions or idle time. The scene captures three canonical properties of multi-agent systems. First, co-operation emerges because the robots pursue a shared objective and divide the labour spontaneously. Second, coordination becomes visible in the synchronised arm and torso movements that prevent interference. Third, successful communication must occur, either implicitly through the shared world model maintained by Helix or explicitly over a wireless data link, even though the company has not disclosed its low-level message formats [32].

Helix itself is layered. A fast, sensor-driven component converts visual and proprioceptive input into millisecond-scale motor commands; a slower, language-conditioned component conducts symbolic reasoning, task decomposition, and error recovery. The fast tier guarantees smooth and collision-free kinematics, whereas the slow tier decides on suitable next actions and explains the rationale behind them. This separation enables the platform to generalise: Helix can receive a novel task, plan a sensible sequence of steps, and then hand control back to the motion layer for precise execution, a pattern that aligns neatly with multi-agent requirements for adaptability and robustness [8].

Although Figure AI keeps its networking stack private, established practice suggests a hybrid strategy. Structured messages inspired by FIPA-ACL might be serialised as lightweight JSON packets over Wi-Fi or 5G links. Such packets would broadcast object poses, task states, or reservation signals for shared workspace, while high-bandwidth event streams would handle real-time proprioceptive data. The visual scene reconstructed by Helix could act as a common blackboard so that every robot sees the same set of affordances. This arrangement mirrors the publish-subscribe designs documented in the broader multi-agent literature [30] .

Natural-language interaction, which Figure AI originally developed in collaboration with OpenAI, extends the agent society to include humans. Warehouse staff,

homeowners, or line-side operators can issue verbal commands that the reasoning tier translates into goals inside the global task graph. The robots then negotiate among themselves to determine who will execute which portion of the request. This integration of human and robotic agents exemplifies current thinking on collaborative multi-agent systems and underlines the importance of reliable, safety-aware communication channels at the human–machine boundary [7].

7.6 Challenges and Future Directions

Recent case studies illustrate the promise and practical reach of AI-agent systems. Large language models already coordinate humanoid robots, write software, and support rich interactions with people. Each demonstration also uncovers limitations that grow more serious once single agents evolve into interconnected societies. Before these systems can be trusted in safety critical or high stakes settings, researchers and engineers must confront a set of technical, operational, and governance obstacles that stretch beyond any one application. The following discussion distils those hurdles and points toward directions that can yield resilient, efficient, and trustworthy multi-agent architectures.

7.6.1 Technical Challenges

The transition from single-agent prototypes to fully deployed multi-agent systems built on large language models exposes several technical fault lines that must be bridged before such systems can be trusted outside the lab.

First, model reliability remains fragile. Even frontier models continue to hallucinate citations, code, and domain facts. In a multi-agent workflow a single false output can ripple through the task graph, undermining the collective result. Interactive agents propagate those mistakes unless a human or an automated verifier intervenes [31].

Second, latency and compute budgets are tightening. The 2025 International AI Safety Report estimates that inference, rather than training, already dominates electricity consumption for many commercial AI services, and projects data-centre demand could reach national-scale levels within a few years [2].

Third, rising autonomy amplifies privacy and security risk. Natural-language interfaces encourage users to reveal sensitive details, which an agent may store indefinitely. Long-term memory, if left unfiltered, also enables covert profiling and automated surveillance [33].

Finally, evaluation and oversight lag behind capability growth. The AI Safety Report finds no consensus methodology for stress-testing plan-capable agents, and notes that benchmarking suites still miss many open-ended failure modes. Without transparent metrics, developers and regulators cannot quantify residual risk or set deployment thresholds [2].

7.6.2 Scaling Secure and Interoperable Agent Societies

Brett Adcock, Figure AI's chief executive, says the company's new BotQ factory is being laid out so the current generation of humanoids can help to build the next one. A self-amplifying loop he sees as the only realistic path to six-figure annual output. Elon Musk offers a parallel vision for Tesla's Optimus programme, telling investors that thousands of robots will join Tesla lines this year and that the total could rise to millions once the machines handle their own assembly tasks. These ambitions shift the scaling challenge from laboratory prototypes to factory throughput, supply-chain resilience, and careful resource scheduling.

Larger fleets also widen the attack surface. Multi-agent prompt injection lets a single poisoned instruction move through shared memories and tool calls. Experiments by Lee and Tiwari show that one adversarial message can leak private data and trigger unintended code across an entire agent cluster [16].

Long-term interoperability and oversight depend on early agreement on shared rules. An OpenAI policy brief argues that monitoring and coordination mechanisms must be ready before systems approach or exceed human-level competence across domains [21]. A common ontology for objects, actions, and error codes, validated through formal conformance tests, would let heterogeneous vendors exchange plans without brittle glue code and would give regulators a clear audit path.

7.7 Conclusion

In this survey, we have followed the journey of intelligent systems from early symbolic programs and simple reflex agents to today's sophisticated entities powered by large language models. We began by defining what makes an AI agent - its autonomy, perceptual abilities, and goal-driven actions - and showed how these individual capabilities scale into multi-agent systems through carefully designed communication protocols. By examining both legacy standards like FIPA-ACL and KQML, we highlighted the essential role of robust messaging in enabling agents to share knowledge, negotiate plans, and cooperate on complex tasks.

Although AI agents have been studied for decades, the emergence of transformer-based LLMs, most visibly through tools like OpenAI's ChatGPT, has offered a first glimpse of their full potential. For the first time, agents can carry on human-level conversations, decompose goals with chain-of-thought reasoning, and ground language in vision and action, all within a unified architecture. This moment feels like the opening chapter of a much larger story.

At the same time, we must acknowledge unanswered questions and persistent limitations. Will agent capabilities continue to grow exponentially, or are we destined to hit a performance stagnation? Can we control model unreliability, manage energy and latency at scale, and secure long-lived data against misuse? These

challenges demand new benchmarks for lifelong, multi-agent learning, shared protocol standards for seamless interoperability, and governance frameworks that keep pace with advancing autonomy.

One certainty stands out: the AI tools we deploy today are the worst we will ever use again. As we push forward, refining standards, stress-testing resilience, and building transparent oversight - each iteration will unlock more powerful, efficient, and trustworthy agent societies. The work we do now will set the stage for an era where AI agents amplify human skills rather than replace it.

References

[1] *Agent Protocol*. Accessed: 2025-05-14. 2025. URL: %5Curl%7Bhttps://agentprotocol.ai/%7D.

[2] Yoshua Bengio et al. *International AI Safety Report 2025*. Tech. rep. Chair: Yoshua Bengio. This publication is licensed under the Open Government Licence v3.0. UK Department for Science, Innovation and Technology, Jan. 2025. URL: %5Curl%7Bhttps://www.gov.uk/government/publications/international-ai-safety-report-2025/international-ai-safety-report-2025%7D.

[3] Yuheng Cheng et al. *Exploring Large Language Model based Intelligent Agents: Definitions, Methods, and Prospects*. 2024. arXiv: 2401.03428 [cs.AI]. URL: https://arxiv.org/abs/2401.03428.

[4] McKinsey & Company. *Why agents are the next frontier of generative AI*. Accessed: 2025-04-30. 2024. URL: %5Curl%7Bhttps://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/why-agents-are-the-next-frontier-of-generative-ai%7D.

[5] Sopan Deb. "ChatGPT Is Costly. But Maybe It's Worth the Price." In: *The New York Times* (2025). Accessed: 2025-04-29. URL: %5Curl%7Bhttps://www.nytimes.com/2025/04/24/technology/chatgpt-alexa-please-thank-you.html%7D.

[6] Abul Ehtesham et al. *A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A), and Agent Network Protocol (ANP)*. 2025. arXiv: 2505.02279 [cs.AI]. URL: %5Curl%7Bhttps://arxiv.org/abs/2505.02279%7D.

[7] Figure AI. *Helix Accelerating Real-World Logistics*. Accessed: 2025-04-30. 2025. URL: %5Curl%7Bhttps://www.figure.ai/news/helix-logistics%7D.

[8] Figure AI. *Helix: A Vision-Language-Action Model for Generalist Humanoid Control*. Accessed: 2025-04-30. 2024. URL: %5Curl%7Bhttps://www.figure.ai/news/helix%7D.

[9] *FIPA Communicative Act Library Specification*. Tech. rep. SC00061G. Accessed: 2025-04-24. Foundation for Intelligent Physical Agents (FIPA), 2002. URL: http://www.fipa.org/specs/fipa00061/SC00061G.html.

[10] Foundation for Intelligent Physical Agents (FIPA). *FIPA ACL Message Structure Specification*. Accessed: 2025-04-24. URL: %5Curl%7Bhttp://www.fipa.org/repository/aclspecs.html%7D.

[11] Andreas Goeldi and Roman Rietsche. *Making Sense of Large Language Model-Based AI Agents*. eng. 2024.

[12] Anna Gutowska. *What is a Multiagent System?* Accessed: 2025-03-20. 2024. URL: %5Curl%7Bhttps://www.ibm.com/think/topics/multiagent-system%7D.

[13] Deekshitha Kosaraju. "AI and Multi-Agent Systems: Collaboration and Competition in Autonomous Environments". In: *International Journal of Research and Review* (2024). URL: %5Curl%7Bhttps://api.semanticscholar.org/CorpusID:274499970%7D.

[14] Benjamin Labaschin. *What are AI agents?* eng. First edition. Sebastopol, CA: O'Reilly Media, Inc., 2023.

[15] Cognition Labs. *Introducing Devin*. Accessed: 2025-04-30. 2025. URL: %5Curl%7Bhttps://docs.devin.ai/get-started/devin-intro%7D.

[16] Donghyun Lee and Mo Tiwari. *Prompt Infection: LLM-to-LLM Prompt Injection within Multi-Agent Systems*. 2024. arXiv: 2410.07283 [cs.MA]. URL: %5Curl%7Bhttps://arxiv.org/abs/2410.07283%7D.

[17] Martin Luther Mfenjou et al. "Methodology and trends for an intelligent transport system in developing countries". In: *Sustainable Computing: Informatics and Systems* 19 (2018), pp. 96–111. ISSN: 2210-5379. URL: %5Curl%7Bhttps://www.sciencedirect.com/science/article/pii/S2210537918300477%7D.

[18] NVIDIA. *What are Large Language Models?* Accessed: 2025-04-29. NVIDIA Corporation. 2024. URL: %5Curl%7Bhttps://www.nvidia.com/en-us/glossary/large-language-models/%7D.

[19] Stuart J. (Stuart Jonathan) Russell and Peter. Norvig. *Artificial Intelligence : a modern approach*. eng. 4th ed. Harlow: Pearson, 2021.

[20] Francesco Antonio Russo. "Devin AI: the first AI Software Engineer that writes code by itself". In: *The Cryptonomist* (Mar. 2025). URL: %5Curl%7Bhttps://en.cryptonomist.ch/2025/03/24/devin-ai-the-first-ai-software-engineer-that-writes-code-by-itself/%7D.

[21] Sam Altman. *Planning for AGI and Beyond*. Accessed: 2025-04-30. 2023. URL: %5Curl%7Bhttps://openai.com/index/planning-for-agi-and-beyond/%7D.

[22] Agnia Sergeyuk et al. "Using AI-based coding assistants in practice: State of affairs, perceptions, and ways forward". eng. In: *Information and software technology* 178 (2025), pp. 107610–. ISSN: 0950-5849.

[23] Zhou Shao et al. "Tracing the evolution of AI in the past decade and forecasting the emerging trends". eng. In: *Expert systems with applications* 209 (2022), pp. 118221–. ISSN: 0957-4174.

[24] Gan Soon et al. "A Review on Agent Communication Language: 5th ICCST 2018, Kota Kinabalu, Malaysia, 29-30 August 2018". In: Jan. 2019, pp. 481–491. DOI: 10.1007/978-981-13-2622-6_47.

[25] Peter Stone and Manuela Veloso. "Multiagent Systems: A Survey from a Machine Learning Perspective". In: *Autonomous Robots* 8.3 (June 2000),

pp. 345–383. URL: %5Curl%7Bhttps://doi.org/10.1023/A:1008942012299%7D.

[26] Cole Stryker. *Types of AI Agents*. Accessed: 2025-04-21. Mar. 2025. URL: %5Curl%7Bhttps://www.ibm.com/think/topics/ai-agent-types%7D.

[27] Rao Surapaneni et al. *Announcing the Agent2Agent Protocol (A2A): A New Era of Agent Interoperability*. Google Developers Blog, accessed 2025-05-14. Apr. 2025. URL: %5Curl%7Bhttps://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability/%7D.

[28] Georgi Tsochev, Roumen Trifonov, and Georgi Naydenov. "Agent communication languages comparison: FIPA-ACL and KQML". In: *The International Scientific Conference Computer Science'2015*. 2015.

[29] Sandip Vaniya, Bhavesh Lad, and Shreyansh Bhavsar. "A survey on agent communication languages". In: *2nd International Conference on Innovation, Management and Service (ICIMS)-Singapore*. 2011.

[30] Analytics Vidhya. *Figure's Helix: AI that Brings Human-Like Robots to your Home*. Accessed: 2025-04-30. 2025. URL: %5Curl%7Bhttps://www.analyticsvidhya.com/blog/2025/02/figures-helix/%7D.

[31] Lei Wang et al. "A survey on large language model based autonomous agents". In: *Frontiers of Computer Science* 18 (2024), p. 186345. DOI: 10.1007/s11704-024-40231-1.

[32] Michael Wooldridge. *An Introduction to MultiAgent Systems*. 2nd. John Wiley & Sons, 2009.

[33] Zhiheng Xi et al. *The Rise and Potential of Large Language Model Based Agents: A Survey*. 2023. arXiv: 2309.07864 [cs.AI]. URL: %5Curl%7Bhttps://arxiv.org/abs/2309.07864%7D.

[34] Satya Prakash Yadav, Dharmendra Prasad Mahato, and Nguyen Thi Dieu Linh. *Distributed artificial intelligence: A modern approach*. CRC Press, 2020.

[35] Yingxuan Yang et al. *A Survey of AI Agent Protocols*. 2025. arXiv: 2504.16736 [cs.AI]. URL: %5Curl%7Bhttps://arxiv.org/abs/2504.16736%7D.

[36] Junhao Zheng et al. *Lifelong Learning of Large Language Model based Agents: A Roadmap*. 2025. arXiv: 2501.07278 [cs.AI]. URL: %5Curl%7Bhttps://arxiv.org/abs/2501.07278%7D.

Chapter 8

Fact-Checking with Large Language Models

Fabien Morgan, Yijie Tong

This seminar paper investigates the role of automated fact-checking systems, focusing on how Large Language Models (LLMs) are working with fact-checking. We introduce key terminology and outline the typical pipeline of fact-checking, including claim detection, evidence retrieval, and verification. Different methods for fact-checking are compared, with special emphasis on the growing advantages of LLM-based approaches, such as improved information retrieval, reasoning, multimodal processing, and scalability. The paper also addresses critical challenges, including reliability issues, ethical concerns, and transparency limitations. Overall, the report highlights both the opportunities and current limitations of using LLMs for automated fact-checking.

Contents

8.1 Introduction 173
8.2 Terminology 173
8.2.1 What is Fact-Checking? 174
8.2.2 What is a Fact? 174
8.3 Fact-Checking Pipeline 174
8.3.1 Check-Worthy Claim Detection 175
8.3.2 Previously Fact-Checked Claims Detection 175
8.3.3 Evidence Retrieval 175
8.3.4 Fact Verification & Fake News Detection 176
8.4 Metrics 176
8.4.1 Claim Detection Metrics 177
8.4.2 Retrieval Metrics 177
8.4.3 Fact Verification Metrics 178
8.4.4 Human Evaluation and Explainability Metrics 178
8.5 Methods 179
8.5.1 Rule-Based and Knowledge-Based Methods 179
8.5.2 Machine Learning-Based Methods 180
8.5.3 Large Language Model-Based Methods 181
8.6 Superiority of LLMs 182
8.6.1 Enhanced Information Retrieval for Fact-Checking . . . 182
8.6.2 Advanced Logical Reasoning and Context Awareness . . 183
8.6.3 Fact-Checking Across Different Modalities 184
8.6.4 Real-Time Fact-checking Automation with Large Scalability 185
8.6.5 Summary of Superiority 186
8.7 Challenges and Limitations 186
8.7.1 Accuracy and Reliability Concerns 186
8.7.2 Ethical and Bias Considerations 187
8.7.3 Transparency and Explainability Challenges 189
8.8 Summary 190

8.1 Introduction

Online misinformation has surged in recent years, becoming a global challenge with dire societal implications [15]. The rapid spread of falsehoods on social media and other platforms can distort public opinion and incite real-world harm. For instance, during the COVID-19 pandemic, viral conspiracy theories about vaccines and cures created widespread confusion and undermined public trust in scientific institutions. Similarly, politically motivated disinformation has fueled social division and threatened democratic stability. In response, fact-checking has emerged as a crucial tool to counter misinformation by providing evidence-based assessments of claims [37].

Manual fact-checking is an extremely time-consuming process. Professional fact-checkers may spend hours or even days verifying a single claim, while vast amounts of misinformation continue to circulate online. This discrepancy between the volume of misinformation and the capacity of human fact-checkers has driven the need for automated fact-checking solutions that can scale efficiently.

Recent advancements in artificial intelligence, particularly in natural language processing, have paved the way for innovative automated fact-checking methods. In particular, LLMs have demonstrated unprecedented capabilities in understanding, generating, and reasoning about text [34]. These models are trained on massive datasets, endowing them with broad world knowledge and the ability to capture complex linguistic nuances. As a result, LLMs offer promising opportunities to enhance fact-checking processes by identifying check-worthy claims, retrieving relevant evidence, and even generating human-readable explanations of their verdicts [30].

At the same time, early studies caution that LLM-based approaches have their own flaws. Despite their impressive performance, these models can sometimes produce plausible yet incorrect information, a phenomenon known as hallucination, and may require careful calibration and external verification [26]. This report examines the development of automated fact-checking systems from a technical perspective. We begin by outlining the key components of a computational fact-checking system, then discuss the evaluation metrics used to assess their performance, and finally survey various methodological approaches-ranging from traditional rule-based systems and machine learning models to modern LLM-based strategies.

Through this overview, the report aims to shed light on the progress made in automating fact-checking and to identify the challenges and emerging opportunities in using LLMs to enhance the fact-checking process.

8.2 Terminology

This section clarifies key concepts in fact-checking and establishes a foundation for subsequent discussions.

8.2.1 What is Fact-Checking?

Fact-checking is a systematic procedure aimed at assessing the accuracy of claims through critical examination of supporting evidence [6][33]. Rather than simply categorizing statements as true or false, it encompasses several well-defined steps. Initially, a claim is identified and extracted from a larger discourse. Next, evidence is gathered from reliable, independent sources such as academic research, official records, and reputable media outlets. Finally, the claim is evaluated by comparing it against the collected evidence to determine its veracity.

Unlike information retrieval systems that focus on locating relevant documents/datasets based on keyword queries [21], or question answering systems that aim to provide direct responses to queries [12], fact-checking involves a critical evaluation of claims against established evidence.

The evaluation inherent in fact-checking is vital for upholding public accountability and trust.

8.2.2 What is a Fact?

In fact-checking, a fact is defined as an objective assertion about the world that can be verified through empirical evidence or authoritative records [10][29]. Unlike opinions or conjectures, facts are based on measurable data, such as scientific findings, historical records, or statistical reports that can be corroborated by multiple sources. A central notion in this context is *ground truth*, which refers to the verified reality against which claims are measured.

Moreover, fact-checking necessitates a clear distinction between verifiable claims and subjective assertions. Compared to Verifiable claims, subjective opinions are influenced by personal beliefs and cannot be conclusively proven. This distinction between both is crucial in prioritizing efforts, ensuring that fact-checkers concentrate on claims that have the potential to impact public understanding and policy decisions [29].

8.3 Fact-Checking Pipeline

Building upon the definitions and concepts introduced earlier, this chapter outlines the sequential stages of an automated fact-checking system. The pipeline is organized into four core components: check-worthy claim detection, previously fact-checked claims detection, evidence retrieval, and fact verification with fake news detection. Each stage addresses unique challenges in systematically filtering, retrieving, and evaluating claims and evidence from a broad corpus of information.

8.3.1 Check-Worthy Claim Detection

The fact-checking process begins by determining which claims warrant further investigation. Not every statement in public discourse is equally significant; some assertions carry a greater potential impact on public understanding or policy decisions. This stage focuses on identifying those claims that, if incorrect, could lead to misinformation with severe consequences. The challenge lies in defining what makes a claim "check-worthy" - a notion that inherently depends on contextual factors such as the claim's subject matter, its potential influence on public opinion, and the risk of harm if left unverified.

Conceptually, check-worthy claim detection involves discerning objective assertions from subjective or trivial commentary. The criteria may include factors such as the presence of quantitative data, references to authoritative events or statistics, and the claim's prominence in the discourse. The goal is to establish a clear, reproducible set of guidelines that help filter out non-essential statements, thereby allowing fact-checkers to concentrate their resources on the most impactful claims. This stage lays the groundwork for a streamlined verification process by ensuring that only the most pertinent claims advance to later stages [9].

8.3.2 Previously Fact-Checked Claims Detection

Before investing resources in verifying a claim from scratch, it is efficient to determine whether the claim has already been examined. This stage involves comparing new claims with a repository of verified statements to identify overlaps or paraphrases. The key conceptual challenge is to recognize when a new claim is semantically equivalent to an existing one, despite variations in wording or structure.

This stage emphasizes the need for a robust framework that captures semantic similarity and detects subtle variations in language. In addition, the problem is compounded by cross-lingual and cultural differences, where a claim fact-checked in one language might reappear in another. The objective is to ensure that redundant work is minimized, thereby allowing fact-checkers to focus on claims that are genuinely new or significantly rephrased [28].

8.3.3 Evidence Retrieval

For claims that have not been previously verified, the system must gather external evidence to support or refute them. Evidence retrieval is a critical step that involves searching large corpora – such as news archives, encyclopedic entries, or academic literature – for documents or passages relevant to the claim. The key challenge here is to ensure that the collected evidence is not only topically related but also reliable and contextually appropriate.

Conceptually, evidence retrieval must address several issues. First, it requires a clear definition of what constitutes "relevant" evidence – evidence should directly address the core components of the claim and provide sufficient context for evaluation. Second, the quality and trustworthiness of sources must be considered; not all retrieved information is equal, and some sources may be biased or outdated. Finally, language variability means that the system must account for synonymous expressions and subtle contextual nuances. This stage serves as the bridge between identifying claims and arriving at a final verification decision [21].

8.3.4 Fact Verification & Fake News Detection

The final stage involves evaluating the claim in light of the gathered evidence to determine its veracity. At its core, fact verification is the process of ascertaining whether external evidence supports, contradicts, or is insufficient to address the claim. This stage is crucial for transforming a collection of texts into a clear judgment regarding the truthfulness of a statement.

Conceptually, the verification process requires a careful comparison between the claim and the retrieved evidence. A major challenge is the possibility of conflicting evidence: different sources may present opposing views or contradictory facts. The system must conceptually weigh the credibility and relevance of these sources to reach a balanced conclusion. Moreover, ambiguous or context-deficient claims add another layer of complexity, as the absence of clear context can hinder the verification process.

In addition, fake news detection extends the verification process by focusing on identifying claims that are not only false, but also deliberately misleading. Here, the emphasis is on understanding the intent and potential impact of the misinformation. This stage encapsulates the ultimate goal of the pipeline to provide an evidence-based judgment on the claim's truthfulness while addressing the inherent challenges of conflicting information and ambiguity [28][33].

In summary, the fact-checking pipeline integrates these four interdependent components to systematically verify claims. Each stage, from the initial filtering of check-worthy statements to the careful evaluation of external evidence, addresses specific conceptual challenges. This structured approach not only streamlines the verification process but also establishes a solid foundation for the subsequent chapters on evaluation metrics and detailed methods.

8.4 Metrics

Each stage of the automated fact-checking pipeline is evaluated using specific metrics that gauge performance and quality. In this section, we outline the metrics used for claim detection, evidence retrieval, claim verification, and explanation

quality, focusing on what these metrics measure and why they matter for fact-checking systems.

8.4.1 Claim Detection Metrics

Claim detection (identifying check-worthy claims) is evaluated with standard classification metrics. Precision is the fraction of retrieved claims that are truly check-worthy, indicating how many flagged claims are actually relevant. Recall is the fraction of all check-worthy claims that the system successfully flags, reflecting the coverage of the system in catching important claims. High precision means that when the system predicts a claim needs checking, it is usually correct (few false positives), whereas high recall means the system finds most of the check-worthy claims (few false negatives). The F1-score, defined as the harmonic mean of precision and recall, provides a single summary metric that balances these two; it is especially useful when both precision and recall are critical.

Another metric used is the ROC-AUC (Receiver Operating Characteristic - Area Under Curve). ROC-AUC measures the model's ability to discriminate between positive (check-worthy) and negative (not check-worthy) instances across all classification thresholds. It is equivalent to the probability that a randomly chosen check-worthy claim is ranked higher by the model than a randomly chosen non-check-worthy claim. An ROC-AUC of 1.0 indicates perfect discrimination, whereas 0.5 indicates performance no better than chance. This metric is threshold-independent and valuable for evaluating claim detection when the decision threshold is adjusted; a high ROC-AUC indicates the detector can maintain good true/false separation regardless of where the cutoff is set [9].

8.4.2 Retrieval Metrics

For the evidence retrieval stage, metrics focus on whether relevant sources are retrieved and how they are ranked. A primary measure is Recall@K, which quantifies the proportion of relevant evidence documents present in the top K retrieval results. For instance, a recall@5 value of 0.8 means that 80% of all relevant documents for a claim appear in the top 5 results. High Recall@K is crucial in fact-checking because missing a key piece of evidence can cause the verification to fail; the system should retrieve as many relevant sources as possible in the initial results.

Ranking quality is evaluated by Mean Reciprocal Rank (MRR). MRR is the average of the reciprocals of the rank of the first relevant result for each query. It ranges from 0 to 1, with higher values indicating that the first relevant evidence tends to appear very early in the ranking. For example, if for one claim the first relevant source is at rank 1 (reciprocal 1.0) and for another it is at rank 3 (reciprocal $\frac{1}{3}$), the MRR over those queries would be approximately $(1.0 + 0.33)/2 \approx 0.665$. A higher MRR means that users or downstream components do not have to search far to find useful evidence. In summary, Recall@K emphasizes completeness, while MRR emphasizes efficiency [21].

8.4.3 Fact Verification Metrics

The claim verification stage is evaluated on both the correctness of the verdict and the supporting evidence. A prominent metric is the FEVER score, introduced by the FEVER fact-checking challenge. The FEVER score is essentially label accuracy conditioned on evidence: to count as correct, the system must predict the right label and provide at least one correct supporting evidence set for the claim. For example, if the system labels a claim as "Supported," it must also supply valid evidence supporting the claim; if it labels a claim "Refuted," it needs to provide evidence that contradicts the claim. Claims labeled "Not Enough Information" require no evidence by definition, but for other labels, missing or incorrect evidence will cause the system to lose the FEVER point even if the label is correct. This strict metric ensures that the system not only makes the right true/false judgments but also justifies them with evidence, reflecting the real-world requirements of professional fact-checking.

Standard accuracy is also measured as the proportion of claims correctly classified, although it does not penalize unsupported answers. In practice, the FEVER score is more stringent than plain accuracy and is often regarded as the primary metric [33].

Another important aspect is the model's confidence in its verdicts. Fact-checking systems may assign probabilities to labels, and it is important that these probabilities are well calibrated. The Expected Calibration Error (ECE) measures how well the predicted probabilities of correctness align with actual outcomes. In this calculation, the predictions are grouped into confidence bins (e.g., 0.5–0.6, 0.6–0.7, etc.), and for each bin, the average predicted confidence is compared to the actual accuracy. ECE is the weighted average of the absolute differences across all bins. A low ECE (closer to 0) indicates that the model's confidence estimates are reliable. Similarly, the Brier Score is used to evaluate the accuracy of probabilistic predictions. The Brier Score is the mean squared error between the predicted probability distribution and the true outcome (treated as 0 or 1). Lower Brier scores imply better-calibrated and more accurate probability estimates. Both ECE and Brier Score assess the quality of uncertainty quantification in the verifier [7].

8.4.4 Human Evaluation and Explainability Metrics

Beyond automated accuracy measures, fact-checking systems are evaluated on the quality of the explanations they provide and on their fairness or bias. One key explainability metric is faithfulness. An explanation is faithful if it accurately reflects the model's actual reasoning process and the evidence used. In other words, the explanation should be based on the same evidence and logic that the system internally used to arrive at its verdict. This is important because an unfaithful explanation, even if it sounds plausible, can mislead users about the rationale behind a decision.

Another crucial evaluation area is bias and fairness. Fact-checking systems should operate impartially across different domains, topics, or demographic groups. Bias auditing involves checking for systematic discrepancies or biases in the system's behavior. For example, one can measure if the system's error rate is higher for certain categories of claims (e.g., political statements from a particular party) than for others. A metric such as delta accuracy can be defined as the difference in accuracy between two subsets of data (such as claims about group X versus group Y); a fair system would have a delta accuracy close to zero, indicating balanced performance. Another measure is disparate impact, which examines whether the probability of assigning a particular verdict (e.g., labeling a claim as false) differs significantly across groups. Overall, these fairness evaluations ensure that the system's decisions and explanations are impartial and equitable.

Often, human evaluation is used to supplement these automated metrics. Experts or users may rate explanations on clarity, correctness, and completeness, or assess whether the explanation adequately reflects the evidence. Although subjective, such evaluations are valuable for capturing aspects that automated metrics may not fully quantify.

In conclusion, a comprehensive evaluation of fact-checking systems spans multiple metrics: precision, recall, F1, and ROC-AUC for claim detection; Recall@K and MRR for evidence retrieval; FEVER score and accuracy, along with ECE and Brier Score for fact verification; and faithfulness and bias measures for explanation quality. Analyzing each stage with these metrics provides insight into a system's strengths and weaknesses and informs improvements in reliability and trustworthiness.

8.5 Methods

This section introduces various approaches to automated fact-checking, providing a high-level overview of the main methodologies used in the field. These approaches include traditional rule-based and knowledge-based methods, machine learning-based methods, and methods that leverage large language models (LLMs). In this chapter, we explain the fundamental ideas behind each approach and discuss how they are applied across the fact-checking pipeline steps: Check-Worthy Claim Detection, Previously Fact-Checked Claims Detection, Evidence Retrieval, and Fact Verification & Fake News Detection.

8.5.1 Rule-Based and Knowledge-Based Methods

Traditional fact-checking often relies on explicit rules and structured data sources. Rule-based systems use manually defined logic or pattern-matching techniques to identify and verify claims. For example, a rule may check whether a numerical fact or a date mentioned in a claim exactly matches an entry in a curated database such

as Wikidata or DBpedia. Knowledge-based methods extend this idea by leveraging large, structured repositories and linked data. These systems are typically applied in the early stages of the pipeline (such as Check-Worthy Claim Detection and Previously Fact-Checked Claims Detection), where fast, precise decisions are required.

A key advantage of rule-based and knowledge-based methods is their transparency. Since the logic is explicitly defined, it is straightforward to understand why a claim was flagged or verified. Users can inspect the rules or database entries to see exactly which fact or value was used as a reference. This interpretability builds trust in the system because every decision can be traced back to a known, verifiable source. However, this transparency comes at the cost of coverage. Such systems are inherently brittle, as they depend on exact matches and fixed patterns; they often fail when claims are rephrased, when novel topics arise, or when the available structured data is incomplete [29]. Their rigidity also means that updating or expanding the knowledge base requires significant manual intervention.

8.5.2 Machine Learning-Based Methods

Machine learning-based methods represent a shift from manually defined rules to models that learn patterns from data. In these systems, fact-checking is formulated as a classification or inference task, where a model is trained on labeled examples of claims and their veracity. Early approaches used traditional classifiers with handcrafted features, while modern techniques increasingly rely on deep neural architectures such as those based on BERT.

These methods are applied in various pipeline steps. For instance, in Check-Worthy Claim Detection, machine learning models can learn from past annotated claims to distinguish which statements are important. In the Fact Verification stage, they help assess whether the retrieved evidence supports or refutes a claim. An important benefit of these models is their ability to handle linguistic variability. Unlike rule-based methods, ML models can generalize over different phrasings and contextual nuances, thus offering broader coverage.

However, a drawback of many ML-based methods is that they often operate as "black boxes." While they achieve high performance, the internal decision-making process is not easily interpretable. This lack of transparency can be a concern, especially in domains where users demand to understand the rationale behind a verdict. Moreover, these models require large amounts of labeled data, which can be resource-intensive to collect and may lead to biases if the training data is not representative [9]. Despite these challenges, the adaptability of ML methods makes them well-suited for handling the complexities of real-world fact-checking.

8.5.3 Large Language Model-Based Methods

The recent emergence of LLMs has introduced a new paradigm for automated fact-checking. LLM-based methods leverage extensive pretraining on vast corpora, endowing them with robust language understanding and reasoning capabilities. In the fact-checking pipeline, LLMs can be applied at several stages. They can enhance Check-Worthy Claim Detection by capturing subtle linguistic cues, assist in identifying previously fact-checked claims through semantic matching, improve Evidence Retrieval by generating effective queries, and contribute to Fact Verification by synthesizing evidence and producing natural language explanations.

One of the most promising aspects of LLM-based methods is their potential for integrated reasoning. Techniques such as retrieval-augmented generation (RAG) enable LLMs to combine internal knowledge with external, up-to-date information. For instance, an LLM can generate a verdict for a claim while concurrently retrieving and citing relevant evidence. In addition, methods like claim decomposition allow LLMs to break down complex claims into simpler sub-claims, which can then be verified individually before aggregating the results. These capabilities not only improve the overall performance of fact-checking systems but also enhance their ability to provide coherent, human-readable justifications.

Nevertheless, LLM-based methods are not without challenges. Despite their impressive performance, LLMs can sometimes generate plausible-sounding but incorrect information, a phenomenon known as hallucination. Moreover, ensuring that these models are well-calibrated and that their generated explanations are faithful to the actual reasoning process remains an ongoing research challenge. Surveys such as that by Vykopal et al. [34] provide an in-depth look at generative LLM approaches in fact-checking, while studies by Fadeeva et al. [3], Li et al. [17], and Tan et al. [30] explore methods for uncertainty quantification and self-verification.

In summary, automated fact-checking systems draw on a spectrum of methodologies. Rule-based and knowledge-based methods offer high transparency and precise decision-making, which is particularly useful in the initial stages of the pipeline, yet they lack flexibility. Machine learning-based methods provide broader coverage and adaptability but often function as black boxes with less interpretability. Large language model-based methods integrate advanced reasoning with natural language understanding, promising a unified solution across multiple pipeline steps, though they present new challenges such as hallucination and calibration. Collectively, these approaches form the foundation of modern fact-checking systems and establish a basis for further discussion on optimization, evaluation, and the integration of more advanced techniques in subsequent chapters.

8.6 Superiority of LLMs

8.6.1 Enhanced Information Retrieval for Fact-Checking

The ability to retrieve information and the reliability of sources are important parts of fact-checking. LLMs can access and synthesize information from a large number of sources, improving coverage compared to traditional fact-checking models. Unlike keyword-based searches, LLMs can retrieve semantically relevant evidence, even if phrased differently. Some LLMs can integrate up-to-date sources, allowing them to verify recent claims better than static models. By aggregating multiple sources, LLMs mitigate bias and increase reliability, and reduce over-reliance on single sources.

Traditional information retrieval (IR) systems, such as search engines, have long been fundamental to efficiently acquiring information. Their evolution from term-based, sparse retrieval systems to neural-based architectures has brought substantial improvements in semantic understanding and contextual relevance. LLMs such as ChatGPT and GPT-4 offer a new paradigm for IR in the context of fact-checking. Due to their expansive pre-training and emergent reasoning abilities, LLMs can function as end-to-end IR agents, participating not only in information retrieval but also in query reformulation, passage re-ranking, and answer synthesis. This enhances them with the ability to interpret vague, ambiguous, or semantically complex claims more accurately than traditional models.

Recent work highlights how LLMs are now embedded across major stages in modern IR pipelines. LLMs can automatically reformulate user inputs into clearer or more complete queries, enhancing the retrieval precision. This is crucial in fact-checking, where the original claim may contain colloquialisms, figurative language, or incomplete context. Beyond keyword matching, LLMs leverage dense embeddings and learned semantics to retrieve information semantically aligned with the claim, even when the evidence is phrased differently. The results were reordered on the basis of relevance, coherence, and credibility. LLMs perform reranking with advanced contextual and logical reasoning, filtering out irrelevant or misleading sources more effectively. Finally, LLMs can read the top-ranked passages, extract factual evidence, and either directly answer the claim or evaluate its veracity.

To achieve this, LLMs frequently interface with external tools such as search engines and knowledge bases to retrieve, verify, and enrich the information they present [45]; [43]; [27]. A particularly effective approach is RAG, which integrates traditional retrieval mechanisms with generative models, allowing LLMs to ground their outputs in verifiable external sources such as scientific literature and authoritative fact-checking databases [16]. Practical systems like New Bing can utilize LLMs to consolidate information from disparate sources into well-structured, factual summaries in response to user queries. Such applications exemplify how LLMs are transforming IR into a more interactive, contextual, and reliable process. As highlighted in the recent comprehensive survey on LLM-based IR systems [46],

the convergence of IR and LLMs is both rapid and profound. Fact-checking systems that harness this synergy stand to gain not just improvements in recall and precision, but also in reliability and user trust.

To summarize, LLMs have much stronger capability on IR, which directly shows the advantages of fact-checking with LLMs: Broader coverage of evidence including ambiguous sources, higher semantic flexibility to detect paraphrased or reworded misinformation, more accurate claim-evidence matching through context-aware embedding and generation, and the ability to distill cross-source information into concise conclusions.

8.6.2 Advanced Logical Reasoning and Context Awareness

LLMs have shown substantial advancements in logical reasoning and contextual understanding, representing a major improvement over earlier NLP models. Unlike traditional systems that often rely on surface-level keyword matching or predefined rule sets, current LLMs can analyze arguments, evaluate logic structures, and understand what users actually mean in a deep intent. This allows them to detect less obvious types of misinformation, including those hidden in sarcasm, rhetorical questions, or figurative language. As a result, LLMs help reduce the number of false alarms in fact-checking systems.

Recent surveys have extensively categorized the logical reasoning abilities of LLMs into several types, including deductive, inductive, abductive, and analogical reasoning [20]. These reasoning modes are increasingly supported by advanced prompting techniques, such as Chain-of-Thought (CoT) and Tree-of-Thought (ToT). These prompting techniques encourage the LLM to break down its thought process into smaller steps, instead of providing the answer directly. [25]. Such kind of structured process helps LLMs better explain their reasoning and identify faulty logic in user-written claims or online content.

In fact-checking systems, such reasoning capabilities allow LLMs to identify contradictions between a claim and retrieved evidence, even when the contradiction is implied rather than explicit. For example, LLMs can detect when a claim misrepresents a scientific finding or incorrectly links cause and effect [19]. Their understanding extends beyond surface-level lexical matching but also to include pragmatic reasoning and discourse analysis. They can consider how language is used in context, helping to avoid mistakes caused by overly literal or rigid interpretations.

Furthermore, LLMs are increasingly effective in understanding contextual cues, such as temporal references, author intent, and discourse structure. This enables them to recognize whether a claim was meant as a joke, satire, or a serious statement in the detection of misinformation in the real world [19]. Considering both logic and context, LLMs offer a robust foundation for trustworthy, scalable, and adaptive fact-checking systems.

Overall, these advancements, including the combination of improved reasoning strategies, context-sensitive language understanding, are giving LLMs a more human-like ability to tell the truth from the given context. These capabilities are especially crucial to handle misinformation, where careful reasoning and context awareness are essential.

8.6.3 Fact-Checking Across Different Modalities

Although fact-checking is primarily applied to textual content, misleading information is even more harmful when the text is combined with other multimedia types, such as images and videos. Multimodal LLMs (MLLMs) can now verify claims that involve images, videos, and even audio, making them more versatile. They can compare textual claims with visual evidence, such as detecting inconsistencies in news articles compared to actual video footage, or finding misinformation in visual media.

MLLMs combine image comprehension with the rich knowledge and explanatory capability of language models, have become tools for humans to process large amounts of information. Researchers in [5] propose a framework for systematically assessing the capacity of current multimodal models to facilitate real-world fact-checking. Given the image and text claim, models are evaluated as fact-checker on the behavior of predicting misinformation. Responses are still in text type, encompassing predictions, explanations, and confidence levels. Their experiments find that GPT-4V exhibits excellent performance across various datasets, with convincing and impressive explanations. Their results show that open-source models have lower performance on accuracy, but still have potential in remembering checked claims and reasoning out manipulated images.

Study in [2] utilized Vision Language Models (VLMs) for multimodal fact-checking, proposing a probing classifier-based approach that extracts embeddings from VLMs and fuses them into a feed-forward neural network. Experiments showed that employing a probing classifier is more effective than base VLM performance, and extrinsic fusion usually outperforms intrinsic fusion. Multimodal fact-checking and explanation generation benchmark dataset are also proposed by researchers to facilitate progress on multi-modal fact-checking [42].

Combining more multimodal types, the MAFT system [13] presents a comprehensive pipeline for multimodal automated fact-checking. It textualizing all input modalities including text, images, videos, and audio. MAFT converts non-text inputs into textual summaries using VLMs including GPT-4o and speech recognition models involving Whisper. This enables unified claim extraction and evidence retrieval via LLMs. The system also integrates external information retrieval and deepfake detection for comprehensive evidence collection. MAFT generates interpretable reports detailing the verification process and results, addressing the challenge of verifying diverse multimodal misinformation more effectively than

modality-specific approaches. This approach shows the growing trend of leveraging textualization to unify multimodal fact-checking within powerful language model frameworks, which is much stronger than traditional methods.

In addition to using context-claim pairs, LLMs can also define the questions and queries individually from the given context and then perform fact-checking based on them. Lrq-Fact[1] is a fully automated framework for multimodal fact verification, which can provide clear and transparent justifications for its final decisions. It generates comprehensive, multi-perspective questions for both image and textual content. The framework addresses the challenge of detecting misinformation across different modalities. Instead of simply using multimodal datasets, which often include image-text pairs, Lrq-Fact utilizes LLMs to generate precise and contextually relevant questions designed to scrutinize textual content, while using VLMs to analyze visual content to answer image-focused queries. By integrating these models into a unified framework, LrqFact addresses the limitations of isolated text or images.

8.6.4 Real-Time Fact-checking Automation with Large Scalability

LLMs enable automated fact-checking for large datasets and social media content in real time, assisting human fact-checkers by providing pre-processed evidence, making manual verification more efficient. Some LLMs can be fine-tuned with user feedback, improving their accuracy over time.

As section 8.6.1 mentioned, external tools have been deeply integrated into LLM systems for fact-checking. Advanced search engines and knowledge bases always have various up-to-date websites and multimodal information. With external knowledge augmentation, LLMs and also MLLMs can use previously retrieved evidence, web engines, or external databases to assist in making decisions and explanations [34].

Besides making the resource side up-to-date, current research also focuses on making the user side more flexible. Such kinds of frameworks and systems improve the efficiency of doing fact-checking in lower resource environments and lower calculation costs. The paper proposed by [18] introduces Self-Checker, a framework comprising a set of plug-and-play modules that facilitate fact-checking by purely prompting LLMs in an almost zero-shot setting. The framework provides a fast and efficient way to construct fact-checking systems in low-resource environments. Empirical results demonstrate the potential of Self-Checker in utilizing LLMs for fact-checking. Another study by [32] presents MiniCheck, a system that builds small fact-checking models with GPT-4-level performance but at a significantly lower cost. By constructing synthetic training data with GPT-4, the system trains models to check each fact in a claim and recognize the synthesis of information across sentences. The approach enables efficient fact-checking, making it suitable for real-time applications.

Another interesting work looks into insights of both news generation and fact-checking. Instead of checking facts on available articles, the system proposed by [24] uses fact-checking on automatic news generation. This innovative system is capable of extracting key information from massive data and generating well-structured, fluent news articles. By incorporating fact-checking technology, the system can effectively stop the misleading information and raise the accuracy and credibility of news. By doing fact-checking at the source of generating news, the goal of real-time service is met, enhancing the efficiency and quality of news production while ensuring the authenticity and reliability of the news content.

These approaches provide valuable insights into the current advancements in automated and real-time fact-checking using LLMs. Different ways all contribute to the time effect enhancement of fact-checking processes in the speeding digital age.

8.6.5 Summary of Superiority

In summary, LLMs demonstrate clear superiority over traditional systems through their ability to retrieve information more effectively, reason with greater logical depth, interpret complex context, and operate across multiple modalities. Their scalability enables real-time fact-checking over large volumes of data, making them particularly well-suited for current fast-paced and diverse information environments. Beyond these, LLMs also offer potential in areas such as zero-shot generalization, personalized verification, and explainable reasoning, further solidifying their role as essential components in modern fact-checking pipelines.

8.7 Challenges and Limitations

8.7.1 Accuracy and Reliability Concerns

Despite the strong performance of LLMs in many areas, their accuracy and reliability remain ongoing concerns, especially in sensitive applications like fact-checking. Sometimes they generate false but apparently right information, such as fabricating sources or incorrect claims. It is also common for certain claims to be misleading rather than completely untrue but misleading due to selective presentation of facts, which is equally challenging to identify. Another challenging issue is that, despite some language models and systems having the capacity to validate claims in real time, there are still clear issues with some rapidly evolving scenarios, like political events or health crises.

One of the most widely reported issues is the tendency of LLMs to generate hallucinations. This kind of output is fluent and confident but factually incorrect. These hallucinations often include fabricated sources, misquoted claims, or outdated facts, which can seriously impact the trustworthiness of automated fact-checking systems. Recent studies have tried to better understand and measure

this problem. For instance, researchers from [11] provide a comprehensive survey on the factuality of LLMs, outlining the main causes of hallucinations and evaluating different strategies for detecting and reducing them. The paper highlights that hallucinations often stem from the model training data and the way it predicts text token-by-token without grounded verification.

Another study by [36] emphasizes that even when LLMs are fine-tuned or prompted carefully, they still tend to produce errors in high-stakes or rapidly evolving domains, such as political news or public health crises. In these areas, up-to-date and precise information is crucial and necessary. These errors are not always obvious lies. Sometimes models present true facts in a misleading way, such as ignoring important context or quoting information selectively. This kind of implicit misinformation is especially difficult to detect and correct.

To address this, several evaluation frameworks have been proposed. Researchers [35] introduced MONITOR, a method to assess the factual reliability of model outputs more directly, aiming for low computational overhead. Meanwhile, the Long-Fact benchmark [38] provides tools to evaluate LLM outputs on multi-sentence answers by checking each statement against evidence from web searches. These tools are useful not only for assessing factuality but also for improving the design of fact-checking systems. FactTest [23] is a statistical method that tests whether a model's factual outputs meet reliability thresholds with high confidence. This approach gives a formal way to assess and compare model performance under strict requirements. To check the fact in LLMs, another metric FACTSCORE [22] is estimated by an automated system using retrieval and a strong language model. This new evaluation method breaks a generation into a series of atomic facts and computes the percentage of atomic facts that are supported by a reliable source of knowledge. Such kind of evaluation method to check the facts on LLM generations before the LLMs check the facts on news and articles helped lay the foundation for the latter to a certain extent.

To summarize, while LLMs have achieved significant progress in language understanding, ensuring factual accuracy, particularly in real-time, high-volume settings, remains a challenging area to address. Addressing these concerns will require not just the development of enhanced models but also the implementation of robust evaluation methods, external verification tools, and careful deployment practices.

8.7.2 Ethical and Bias Considerations

While LLMs have brought significant progress to automated fact-checking, these models may have various kinds of ethical and bias issues. Fact-checking models typically prioritize mainstream sources of information, which may reinforce the dominant narrative at the expense of credible but less conspicuous sources. LLMs may reinforce misinformation if the training data for the LLM is biased or if the LLM generates fact-checks based on faulty reasoning. Many fact-checking models perform poorly in less-representative languages and dialects, leading to differences in accuracy.

Using LLMs on fact-checking may also raise important ethical concerns. One major issue is the presence of real-world bias in these models. Since LLMs are trained on massive text corpora collected from the internet, they often inherit and reproduce the societal, cultural, and political biases embedded in those sources [4]. This means that even if a model appears to be neutral, it may unknowingly support some particular worldviews or points. In this case, the interpretation of claims and the response to certain claims are influenced. As a result, these systems may unintentionally reinforce stereotypes or marginalize less dominant views.

Another concern is that many fact-checking systems built with LLMs tend to prioritize information from popular or widely recognized sources. While this may improve consistency and reliability in some cases, it can also marginalize credible but less prominent sources, reinforcing dominant narratives and limiting diversity in viewpoints [39]. In politically sensitive or controversial contexts, over-reliance on mainstream media sources could lead to biased fact-checking that ignores minority viewpoints or other evidence. This could affect public trust in fact-checking tools and raise questions about whose version of the truth the system supports.

Language bias is also a pressing challenge in the development of fact-checking models. Currently, most LLMs perform best on high-resource languages, such as English, and often struggle with low-resource languages and dialects [44]. This difference leads to variations in accuracy, making fact-checking tools less reliable for minority language speakers. In multilingual or international environments, this can disadvantage large segments of the population and widen the digital divide. Efforts to develop multilingual fact-checking models are increasing, but gaps remain large.

Moreover, LLMs can still generate fact-checks that are flawed or misleading, especially if their reasoning is based on biased, outdated, or insufficient evidence. In some cases, they may even reinforce misinformation unintentionally if the original training data contained inaccurate or harmful content [4]. Because these models generate text with high fluency and confidence, users may mistakenly trust fact-checks that are incorrect or misleading. Researchers are exploring mitigation strategies, such as filtering training data, improving prompt design, and adding post-processing bias detection layers, to reduce these risks [8]. However, ensuring ethical and fair behavior in LLM-based fact-checking systems remains an open challenge.

In short, while LLMs are powerful tools for enhancing the scale and speed of fact-checking, it is essential to be aware of their ethical limitations. Bias, unequal language coverage, and overreliance on dominant sources are significant risks that need to be carefully managed. Ongoing research, transparency in system design, and inclusive data practices will be key to building more equitable and trustworthy fact-checking technologies.

8.7.3 Transparency and Explainability Challenges

Automated LLM-based fact-checking systems often provide verdicts without clear reasoning. They often produce clear verdicts labeling claims as true or false, but without showing how they arrived at those conclusions. Even if CoT and ToT make the reasoning steps more visible, the explanation of the decision still can be irrelevant, inaccurate, or ambiguous. Users and researchers alike have raised concerns that such "black-box" behavior undermines trust, since a verdict without evidence leaves readers unable to judge the quality of the fact-check [40].

Efforts to make LLM fact-checkers more transparent have focused on adding citations or highlighting the evidence used. However, studies show that models still generate inaccurate or irrelevant references. For example, a recent evaluation study found that many top-performing LLMs attribute facts to wrong sources or invent citations entirely, reducing the usefulness of links meant to back up their claims [40].

One promising approach is that the model first identifies relevant evidence and then explains how that evidence supports or refutes the claim. CorXFact [31] adopts this strategy by explicitly modeling the relationship between each piece of evidence and the claim, producing explanations that are closer to human reasoning. In experiments, CorXFact showed higher alignment with expert-written rationales compared to end-to-end generation methods. This method helps the model provide more transparent explanations.

Another line of work frames explainability as a question-answering task. By generating questions and answers from claims and then answering the same questions from evidence, systems proposed by [41] create a chain of reasoning that can be followed step by step. Leveraging question answering as a proxy, they break down automated fact-checking into several sub-processes. This format exposes the underlying logic while allowing users to verify each answer independently, addressing the explainability challenges to some extent.

Beyond individual models, the survey of explainable fact-checking methods [14] also highlights the need for standardized benchmarks and clear evaluation metrics. The formats of existing datasets vary widely. Some of them focus on highlighting supporting sentences, while others require full paragraph summaries, making it difficult to directly compare various methods. The survey asks for shared tasks that combine verdict accuracy with explainability scores to ensure that future systems can both get the right answer and present their work in a human-readable way.

In summary, while transparency and explainability remain challenging, recent research offers concrete solutions: better evidence attribution protocols, claim-evidence correlation models, and question-driven explanations. Adopting these methods and evaluating them on common benchmarks will be key to building LLM fact-checkers that are accurate, transparent, and trustworthy.

8.8 Summary

This seminar paper explored the development, methodologies, strengths, and challenges of automated fact-checking systems, with a particular focus on the growing role of LLMs. It established clear definitions of fact-checking and the nature of what a fact is. Followed by a detailed breakdown of the fact-checking pipeline, ranging from detecting check-worthy claims to evidence retrieval and final verification. The paper then outlined key evaluation metrics used to assess the performance and reliability of such systems. Different fact-checking methods were compared, showing how LLMs have significantly advanced the field.

The superiority of LLMs lies in their improved ability to retrieve relevant information, handle nuanced logical reasoning, process multimodal content, and support real-time, large-scale applications. The challenges section highlighted several ongoing limitations of LLMs, including their tendency to generate convincing but incorrect information, biases stemming from training data, and the lack of transparency in how they arrive at their conclusions. These issues pose ethical and practical concerns, especially in high-stakes domains like politics and public health. However, current research has been working on proposing useful strategies to face the challenges.

In conclusion, while LLMs represent a powerful step forward for automated fact-checking, realizing their full potential requires addressing these limitations through more robust evaluation, improved model transparency, and greater inclusivity across languages and cultures. Our report offers a basis for comprehending the potential and drawbacks of LLM-based fact-checking systems in the current digital information environment.

References

[1] Alimohammad Beigi et al. “LRQ-Fact: LLM-Generated Relevant Questions for Multimodal Fact-Checking”. In: *CoRR* abs/2410.04616 (2024). URL: %5Curl%7Bhttps://doi.org/10.48550/arXiv.2410.04616%7D.

[2] Recep Firat Cekinel and Çağrı Çöltekin. “Multimodal Fact-Checking with Vision Language Models: A Probing Classifier based Solution with Embedding Strategies”. In: *Proceedings of the 31st International Conference on Computational Linguistics*. Ed. by Owen Rambow. Abu Dhabi, UAE: Association for Computational Linguistics, Jan. 2025, pp. 4622–4633. URL: %5Curl%7Bhttps://aclanthology.org/2025.coling-main.310/%7D.

[3] E. Fadeeva et al. “Fact-checking the output of large language models via token-level uncertainty quantification”. In: *Proceedings of the ACL Findings*. 2024, pp. 9367–9385.

[4] Isabel O. Gallegos et al. *Bias and Fairness in Large Language Models: A Survey*. 2024. arXiv: 2309.00770 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2309.00770%7D.

[5] Jiahui Geng et al. *Multimodal Large Language Models to Support Real-World Fact-Checking*. 2024. arXiv: 2403.03627 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2403.03627%7D.

[6] Lucas Graves. *Deciding What's True: The Rise of Political Fact-Checking in American Journalism*. Princeton, NJ: Princeton University Press, 2016.

[7] Chuan Guo et al. "On Calibration of Modern Neural Networks". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. PMLR. 2017, pp. 1321–1330.

[8] Yufei Guo et al. *Bias in Large Language Models: Origin, Evaluation, and Mitigation*. 2024. arXiv: 2411.10915 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2411.10915%7D.

[9] R. Hassan et al. "ClaimBuster: The first-ever system for check-worthiness detection". In: *Proc. of the NAACL Workshop on Fact Extraction and VERification (FEVER)*. 2017.

[10] International Fact-Checking Network (IFCN). *Code of Principles*. 2019. URL: %5Curl%7Bhttps://www.poynter.org/ifcn/%7D.

[11] Ziwei Ji et al. "Survey of Hallucination in Natural Language Generation". In: *ACM Computing Surveys* 55.12 (Mar. 2023), pp. 1–38. ISSN: 1557-7341. DOI: 10.1145/3571730. URL: http://dx.doi.org/10.1145/3571730.

[12] D. Jurafsky and J. H. Martin. *Speech and Language Processing (3rd ed., draft)*. Pearson, 2020.

[13] K. Kakizaki, Y. Matsunaga, and R. Furukawa. "MAFT: Multimodal Automated Fact-Checking via Textualization". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 39. 28. 2025, pp. 29646–29648. DOI: 10.1609/aaai.v39i28.35354.

[14] Neema Kotonya and Francesca Toni. *Explainable Automated Fact-Checking: A Survey*. 2020. arXiv: 2011.03870 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2011.03870%7D.

[15] David M. J. Lazer et al. "The science of fake news". In: *Science* 359.6380 (2018), pp. 1094–1096.

[16] Markus Leippold et al. *Automated Fact-Checking of Climate Change Claims with Large Language Models*. 2024. arXiv: 2401.12566 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2401.12566%7D.

[17] M. Li et al. "Self-Checker: Plug-and-play modules for fact-checking with large language models". In: *Proceedings of NAACL Findings*. 2024, pp. 163–181.

[18] Miaoran Li et al. *Self-Checker: Plug-and-Play Modules for Fact-Checking with Large Language Models*. 2024. arXiv: 2305.14623 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2305.14623%7D.

[19] Gionnieve Lim and Simon T. Perrault. *Evaluation of an LLM in Identifying Logical Fallacies: A Call for Rigor When Adopting LLMs in HCI Research*. 2024. arXiv: 2404.05213 [cs.HC]. URL: %5Curl%7Bhttps://arxiv.org/abs/2404.05213%7D.

[20] Hanmeng Liu et al. *Logical Reasoning in Large Language Models: A Survey*. 2025. arXiv: 2502.09100 [cs.AI]. URL: %5Curl%7Bhttps://arxiv.org/abs/2502.09100%7D.

[21] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge, UK: Cambridge University Press, 2008.

[22] Sewon Min et al. *FActScore: Fine-grained Atomic Evaluation of Factual Precision in Long Form Text Generation*. 2023. arXiv: 2305.14251 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2305.14251%7D.

[23] Fan Nie et al. *FactTest: Factuality Testing in Large Language Models with Finite-Sample and Distribution-Free Guarantees*. 2024. arXiv: 2411.02603 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2411.02603%7D.

[24] Xirui Peng et al. *Automatic News Generation and Fact-Checking System Based on Language Processing*. 2024. arXiv: 2405.10492 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2405.10492%7D.

[25] Aske Plaat et al. *Reasoning with Large Language Models, a Survey*. 2024. arXiv: 2407.11511 [cs.AI]. URL: %5Curl%7Bhttps://arxiv.org/abs/2407.11511%7D.

[26] D. Quelle and A. Bovet. "The perils & promises of fact-checking with large language models". In: *Frontiers in Artificial Intelligence* 7 (2024), p. 1341697.

[27] Dorian Quelle and Alexandre Bovet. "The perils and promises of fact-checking with large language models". In: *Frontiers in Artificial Intelligence* 7 (Feb. 2024). ISSN: 2624-8212. DOI: 10.3389/frai.2024.1341697. URL: http://dx.doi.org/10.3389/frai.2024.1341697.

[28] K. Shu et al. "Fake News Detection on Social Media: A Data Mining Perspective". In: *ACM SIGKDD Explorations Newsletter* 19.1 (2017), pp. 22–36. URL: %5Curl%7Bhttps://dl.acm.org/doi/abs/10.1145/3137597.3137600%7D.

[29] C. Silverman. *Verification Handbook: A Definitive Guide to Verifying Digital Content for Emergency Coverage*. European Journalism Centre, 2016.

[30] X. Tan, B. Zou, and A. A. Ti. "Evidence-based interpretable open-domain fact-checking with large language models". In: *arXiv preprint* (2023). Preprint.

[31] Xin Tan. "Improving Explainable Fact-Checking with Claim-Evidence Correlations". In: *Proceedings of the 31st International Conference on Computational Linguistics*. Ed. by Owen Rambow. Abu Dhabi, UAE: Association for Computational Linguistics, Jan. 2025, pp. 1600–1612. URL: %5Curl%7Bhttps://aclanthology.org/2025.coling-main.108/%7D.

[32] Liyan Tang, Philippe Laban, and Greg Durrett. *MiniCheck: Efficient Fact-Checking of LLMs on Grounding Documents*. 2024. arXiv: 2404.10774 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2404.10774%7D.

[33] James Thorne et al. "FEVER: a Large-scale Dataset for Fact Extraction and VERification". In: *Proceedings of NAACL-HLT 2018*. 2018, pp. 809–819. URL: %5Curl%7Bhttps://aclanthology.org/N18-1074/%7D.

[34] I. Vykopal et al. "Generative large language models in automated fact-checking: A survey". In: *arXiv preprint* (2024). Preprint.

[35] Weixuan Wang. "Assessing Factual Reliability of Large Language Model Knowledge". In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Ed. by Kevin Duh. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 805–819. DOI: 10.18653/v1/2024.naacl-long.46. URL: %5Curl%7Bhttps://aclanthology.org/2024.naacl-long.46/%7D.

[36] Yuxia Wang et al. *Factuality of Large Language Models: A Survey*. 2024. arXiv: 2402.02420 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2402.02420%7D.

[37] Claire Wardle and Hossein Derakhshan. *Information Disorder: Toward an Interdisciplinary Framework for Research and Policy Making*. Tech. rep. Council of Europe, 2017. URL: %5Curl%7Bhttps://rm.coe.int/information-disorder-toward-an-interdisciplinary-framework-for-researc/168076277c%7D.

[38] Jerry Wei et al. *Long-form factuality in large language models*. 2024. arXiv: 2403.18802 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2403.18802%7D.

[39] Laura Weidinger et al. *Ethical and social risks of harm from Language Models*. 2021. arXiv: 2112.04359 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2112.04359%7D.

[40] Rui Xing, Timothy Baldwin, and Jey Han Lau. *Evaluating Evidence Attribution in Generated Fact Checking Explanations*. 2025. arXiv: 2406.12645 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2406.12645%7D.

[41] Jing Yang et al. "Explainable Fact-Checking Through Question Answering". In: *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, May 2022, pp. 8952–8956. DOI: 10.1109/icassp43922.2022.9747214. URL: http://dx.doi.org/10.1109/ICASSP43922.2022.9747214.

[42] Barry Menglong Yao et al. "End-to-End Multimodal Fact-Checking and Explanation Generation: A Challenging Dataset and Models". In: *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '23. Taipei, Taiwan: Association for Computing Machinery, 2023, pp. 2733–2743. DOI: 10.1145/3539618.3591879.

[43] Shunyu Yao et al. *ReAct: Synergizing Reasoning and Acting in Language Models*. 2023. arXiv: 2210.03629 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2210.03629%7D.

[44] Caiqi Zhang, Zhijiang Guo, and Andreas Vlachos. *Do We Need Language-Specific Fact-Checking Models? The Case of Chinese*. 2024. arXiv: 2401.15498 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2401.15498%7D.

[45] Xuan Zhang and Wei Gao. *Towards LLM-based Fact Verification on News Claims with a Hierarchical Step-by-Step Prompting Method*. 2023. arXiv: 2310.00305 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2310.00305%7D.

[46] Yutao Zhu et al. *Large Language Models for Information Retrieval: A Survey*. 2024. arXiv: 2308.07107 [cs.CL]. URL: %5Curl%7Bhttps://arxiv.org/abs/2308.07107%7D.

Chapter 9

Machine Learning in 5G Security – An Overview

Ajeong Shin, Dora Silva, Nasim Nezhadsistani

The fifth generation (5G) is a huge help to stay connected and productive in our modern world. It provides huge advancements to its predecessors, such as lower latency and enhanced connectivity whilst maintaining higher speeds. This is achieved through a higher bandwidth, network slicing, and edge computing. This allows us to have real-time data exchange and automation, creating the possibility for autonomous vehicles and "Smart Cities". Despite its impressive capabilities, 5G does have some security breaches, such as protocol vulnerabilities through Denial of Service attacks and privacy concerns through weak cryptographic implementations. This paper should give an overview of different Machine Learning (ML) techniques and how they will solve security issues. For example, Reinforcement Learning can optimize authentication processes, minimizing the risk of an attack.

Contents

9.1 Introduction and Problem Statement 197
9.2 5G Security Architecture and Vulnerabilities 198
9.2.1 5G Security Requirements and Protocol Design 198
9.2.2 Security Domains in 5G 198
9.2.3 Common Attack Vectors in 5G Networks 200
9.2.4 Current Security Challenges 200
9.3 AI and ML Techniques for Cybersecurity 200
9.3.1 Types of Machine Learning 200
9.3.1.1 Supervised Learning 200
9.3.1.2 Unsupervised Learning 201
9.3.1.3 Reinforcement Learning 201
9.3.1.4 Federated Learning 201
9.3.2 ML in 5G Security 202
9.3.2.1 Authentication and Access Control 202
9.3.2.2 Network Anomaly and Intrusion Detection . . 202
9.3.2.3 Threat Intelligence and Analysis 202
9.3.2.4 Privacy Protection and Preservation 203
9.3.2.5 Security Automation and Orchestration 203
9.3.2.6 AI in Cybersecurity 203
9.3.2.7 Challenges and Limitations 204
9.4 Evaluation and Discussion of the case study 204
9.4.1 Approach 205
9.4.2 Findings 205
9.4.3 Evaluation 206
9.4.4 Conclusion 207
9.5 Future Directions 207
9.5.1 ML integration in 6G 207
9.5.2 Research Opportunities 208
9.6 Conclusion 208

9.1 Introduction and Problem Statement

5G connects almost one million devices per square kilometer and has a data rate of up to 20 gigabits per second [1]. It is essential to connect to our modern world. With a vast number of different devices connected to the network, self-driving cars, or cell phones, 5G security has reached its limits. The needed cryptographic methods are costly and cannot be performed on every device, creating latency and communication overhead. Furthermore, they lack the adaptability required for the dynamic network environments that 5G offers. This creates a difficulty in creating precise authentication models and detecting compromised security keys [3]. Therefore, the main objective of this paper is to investigate the applicability of Machine Learning (ML) in enhancing 5G security.

The network structure difference between 4G and 5G is one of the main reasons for the security difficulty in 5G. Unlike the hardware-centric 4G environment, the 5G environment consists of a software-based network. With the vast increase in demand for personal devices and server computers, the overhead forward edge side became heavier than ever. To mitigate this problem, 5G allows network slicing to increase capacity to handle several devices over shared network resources, enhancing the communication speed in a bunch of devices world. However, it results in new security weaknesses in the 5G as well, such as Software Define Networks(SDN), Network Function Virtualization(NFV), cloud, and edge computing [1].

As a promising solution, ML is actively discussed in the communication field. Given the diverse types of ML and their strengths, it is expected that ML will be used as the core 5G security methodology, optimized to address the unique security needs and vulnerabilities of edge devices. ML can foresee the expected security issues based on the learning data from real-world history and allows us to avoid similar threats. Moreover, it classifies new types of threats and applies the solution based on prediction results. Artificial Intelligence (AI) also reinforces ML performance, especially in the aspects of threat hunting and investigation[8].

In this paper, we will introduce multiple problems that come with a 5G network environment. After that, we will inspect the types and strategies of each ML technology and discuss how to implement this methodology into the 5G security scene. In the evaluation and discussion of the case study section, we assess the practicality of Federated Learning, one of the ML technologies, as a 5G security solution. Finally, we wrap up this paper with ideas on preparing for the upcoming 6G network environment with ML, finding research opportunities, and concluding with a concise summary.

9.2 5G Security Architecture and Vulnerabilities

9.2.1 5G Security Requirements and Protocol Design

IMT-2020 systems (5G) allow more flexibility, security, and reliability than IMT-advanced (4G). While previous generations were centralized, 5G introduced a distributed SBA (service-based architecture). This architecture enables 5G to provide diverse services, including three main parts: eMBB (enhanced Mobile Broad-Band), mMTC (massive Machine-Type Communication), and URLLC (Ultra-Reliable Low-Latency Communication). While this makes our lives easy, security issues do also occur. For instance, eMBB needs confidentiality and integrity based on security transmission. mMTC requires authentication of the devices, limiting authenticated nodes and enabling End-to-End security. URLLC demands availability and resilience as well[7, 3].

Key 5G services such as eMBB, mMTC, and URLLC influence protocol design, making it dynamic and software-centric. To handle the increasing amount of edge devices and ensure fast and reliable communication, 5G adopts SDN(Software-Defined Networking) and NFV(Network Function Virtualization)[1]. It allows the network to be more flexible, modular, and virtualized to distribute limited network resources to diverse edge devices. On the other hand, the 5G architecture of 3GPP introduces a network slicing feature, literally "slicing" the bandwidth to multi-edge users. To keep a software-based, expandable network, several authentication frameworks like 5G-AKA (Authentication and Key Agreement) and expansion based on EAP (Extensible Authentication Protocol) should also be made[3]. API-based service calling is also one of the key design differences in 5G. Following the 5G protocol design, diverse security features are implemented. With expanded 3GPP-defined security features, 5G reinforced End-to-End security features based on slicing [9, 1]. Each slice can adopt a separate specific security policy, enabling not only optimized edge devices' usage in the network, but also ensuring security. These slices are isolated, preventing unexpected spread effects when other slices are attacked from outside. SUCI(Subscription Concealed Identifier), one of the 5G key security mechanisms, is used for encrypting user identification information[9, 3]. For an API-based communication environment, OAuth 2.0 supports a unified token authentication system for every network feature[10]. This feature ensures high versatility and expandability by synchronizing the authentication method, fitted with HTTP/RESTful API-based SBA[12].

9.2.2 Security Domains in 5G

3GPP defines six security domains: Network Access Security, Network Domain Security, User Domain Security, Application Domain Security, SBA Domain Security, Security Visibility, and Configurability[9]. Network Domain focuses on authentication and protection for the wireless channels between devices and networks like SUCI[3]. On the other hand, Network Domain Security is related to

interior network protection between communication, for instance, TLS (Transport Layer Security)[1]. Moving closer to the user side, User Domain Security is protecting the device itself, user identification, or access at the edge node with USIM or PIN[9]. Application domain security is a perspective on application security communication that uses end-to-end encryption like HTTPS. SBA Domain Security defines API security between NF (network features) in the SBA with OAuth2.0[1]. Finally, Security Visibility and Configuration supply controllable features to users, such as alarms or Zero Trust[10].

Table 9.1: 5G Security Domains and Example Features (3GPP TS 33.501)

Security Domain	**Description**	**Example Security Features**
Network Access Security	Protects communication between UE and network through authentication and secure channel establishment	SUCI, EAP-AKA', key agreement
Network Domain Security	Secures communication between internal network functions and across transport interfaces	IPsec, TLS (Transport Layer Security), GTP-U integrity
User Domain Security	Ensures security of the user equipment and local user authentication	USIM, PIN, biometric authentication
Application Domain Security	Protects communication between user devices and external application services	End-to-end encryption, HTTPS, TLS
SBA Domain Security	Secures communication between network functions in SBA using service-based interfaces	OAuth2, API access control, JSON Web Token
Security Visibility & Configurability	Provides transparency and configurability of security settings to the user	Security status indication, user policies, Zero Trust-based visibility

9.2.3 Common Attack Vectors in 5G Networks

One key change to 5G architecture is making network components programmable and configurable with SBA [1]. This software-centric architect can result in severe network risk not only to network security but also to the user side by imposing malware into weak software. It can lead to DoS (Denial of Service) attacks or cybersecurity threats[4]. Massive device communication with wireless networks is vulnerable to wireless interference like jamming[10]. Today, people highly depend on wireless networks on every device, including life-related machines such as auto-driving cars or surgery machines based on 5G's URLLC service. This use case can have unrecoverable results when it is exposed to a threat. As an extension of URLLC, sensitive data is frequently transmitted via the network. During the transmission, several attacks attempt to extort money by holding valuable data hostage. These threats keep increasing as the network is developed, heading towards virtualization, cloudification, and decentralization to handle the booming demand for IoT(Internet of Technology)[1].

9.2.4 Current Security Challenges

While preventing every attack is important, human-targeted attacks are the most notorious. Furthermore, these attacks are considered only to be targeted by people who are unfamiliar with technology, but nowadays, we can easily see that even engineers can fall victim to cybersecurity crimes such as phishing and spoofing [1]. 5G's dynamic configuration is also a security challenge, as it is hard to adapt security policies appropriately in real time. For instance, cloud or edge computing creates and removes the network in real-time; it needs dynamic policy, not static policy, and requires transition time. This time-gap can be targeted by an attacker. Lightweight security is the main challenge in 5G security. Devices that have light arithmetic performance or battery, such as sensors, cannot be applied to heavyweight traditional encryption algorithms [8]. In a 5G environment, every type of edge device should be considered, and the authentication type should be selected[3]. For that, authentication and lightweight security methods are required.

9.3 AI and ML Techniques for Cybersecurity

9.3.1 Types of Machine Learning

9.3.1.1 Supervised Learning

Supervised learning uses labeled data sets to map input information into output functions. To achieve this, different techniques are used such as Decision Trees, Support Vector Machine(SVM) or K-Nearest Neighbors(KNN)[3]. Unlike other types of learners (unsupervised learning and reinforcement), supervised learning

assumes the availability of input-output pairs (x, y)[8]. With a large amount of data sets, this learning method shows high accuracy. The more number of instance with labeled data, the model becomes more robust and accurate through the learning by minimizing the gap between the estimated result and the target of labeled data. In cases where the labeled data is limited, Transfer Learning, a method within the supervised learning paradigm, can improve model performance by leveraging knowledge from a pre-trained supervised model [8]. In 5G network, classification ML algorithms can be used to detect anomalies by monitoring network parameters [1].

9.3.1.2 Unsupervised Learning

In contrast to supervised learning, unsupervised learning uses unlabeled data. With unlabeled data collection, the model attempts to figure out boundaries between related datasets. This learning algorithm can help uncover hidden patterns and structures by using techniques like K-means Clustering for grouping data and Principal Component Analysis (PCA) by reducing dimension [3]. The dominant application of unsupervised learning in a 5G environment is clustering. Clustering is often deployed for grouping from immensely large traffic data patterns as well as identifying suspicious activities, which is especially beneficial in software-centric and virtualization environments [1].

9.3.1.3 Reinforcement Learning

In Reinforcement Learning (RL), the model is formulated using a Markov Decision Process (MDP) based on a mathematical formula and optimized for the desired object by using a reward system [8]. In a given state, each action of the model is evaluated and rewarded by the system to identify the most rewarded policy across multiple scenarios under specific conditions. While supervised shows powerful performance in "pre-learned" environment, Deep Reinforcement Learning (DRL) performs well in different conditions based on generalized experience from interaction with an environment. This algorithm can recognize and prevent original attacks like obfuscation, polymorphism, or impersonation, reducing 5G security threats.

9.3.1.4 Federated Learning

Federated Learning (FL) is a decentralized approach, unlike previously addressed ML types, supervised learning, unsupervised learning, and Reinforcement learning, which are centralized methodologies. The difference between other ML methods is that FL enables training on data from diverse devices called clients. Each device sends local parameters trained on its own data to a central server. After aggregation of the parameters in the server, it goes back to the client, the local server, to update local cores identically [**Perifanis2023**]. This algorithm is well-fitted with

edge devices in a 5G network environment, especially in terms of securing privacy. Today, FL shows performance in multiple purposes on anomaly detection and time-series forecasting for 5G security.

9.3.2 ML in 5G Security

The integration of ML in 5G security has opened new possibilities for prevention, detection, and reaction to threats. It solves the issue of evolving attacks by shifting from static, rule-based detection to dynamic and adaptive security mechanisms.

9.3.2.1 Authentication and Access Control

Traditional methods for authentication are at risk for spoofing and credential theft. The physical layer could be combined with ML to improve authentication. By leveraging signal characteristics like RSSI, CFO, and CIR, it would enable device fingerprinting and strengthen access control without relying solely on cryptographic credentials. This ML model would analyze whether values of the signal are within reason and are not from an adversary edge device. By using different ML techniques, most prominent unsupervised learning, user behavior is recognized, and the model is trained on them. Unauthorized authentication can be detected if they deviate from normal user patterns. This is extremely helpful as there are a lot of unpredictable user-device interactions, which ML can detect and researchers do not have to predefine the data. Additionally, through situation-aware authentication, where the user's and network's temporal and spatial dynamics are taken into consideration, ML is able to create risk-based access control [5, 3].

9.3.2.2 Network Anomaly and Intrusion Detection

Through ML-powered Network Intrusion Detection Systems (NIDS) and its combination with the dynamic structure of 5G, we can analyze large volumes of traffic data. This can be achieved, for example, through unsupervised learning methods such as K-Means Clustering. Hidden patterns in unlabeled data become apparent, facilitating the detection of anomalies that are potential cyber threats [5]. Gudepu et al. also indicates that advanced approaches using Deep Transfer Learning (DTL) can introduce here a new perspective. By training a model on one network's data to detect threats, we are able to analyze other network's data, overcoming the scarcity of good, labeled datasets [5].

9.3.2.3 Threat Intelligence and Analysis

Saha et al. proposes a new framework that integrates machine learning with constraint satisfaction programming (CSP) to predict new exploit combinations. The

ML-approach used in his studies is an automatically constructed attack graph analysis. There, multiple known exploits are modeled to analyze potential attack paths through the network. This intelligence framework demonstrates how ML can detect vulnerable patches by going through the graph [10].

9.3.2.4 Privacy Protection and Preservation

Federated Learning can address privacy concerns by moving the computation closer to edge-devices. Privacy preservation would be achieved by only exchanging ML model parameters, as FL allows on-device training. This mitigates the risk of data theft and manipulation during transmission [1].

9.3.2.5 Security Automation and Orchestration

ML models at different layers in 5G network can coordinate to provide automated security, allowing real-time threat detection and decision-making without human intervention [1].

This is also supported by research by Afaq et al. The study emphasizes a layered approach. They especially propose different kinds of ML applications for different kinds of threats. Supervised Learning is best suited for anomaly classification and DDoS attacks, whilst unsupervised learning, as mentioned before, is great for clustering traffic data and detecting unknown threats. Deep Learning would be best for intrusion detection and malware recognition. He proposes Federated Learning as a privacy-preserving model for training at scale. Learning would not be limited to the center core but can be extended to edge devices. Through reinforcement learning, adaptive attack mitigation strategies can be created. Automated actions based on the threats can be achieved, when they optimize their countermeasures through continuous interaction with the environment. RL would orchestrate the responses and use the insights gained from the other ML models. This would create an adaptive autonomous system that would increase their resilience over time by combining different models together. [1].

9.3.2.6 AI in Cybersecurity

Generative AI models can simulate potential attack traffic, allowing ML models to adapt and train themselves further, before deployment into the network. Gudepu et al. proposes Gen AI, specifically Variational Autoencoders (VAEs) to create synthetic traffic data that represents real-world observations. Through this, additional data simulating attacks becomes available. The goal is to provide not only additional data for training, but also data that can be fed to the model to prevent degeneration by users performing complex behavior [5].

9.3.2.7 Challenges and Limitations

Despite its effectiveness, ML analysis of threats in 5G networks can have several challenges and limitations.

First of all, datasets in high-quality regarding 5G security incidents are limited due to privacy restrictions and lack of instrumentation to create the data. High-quality labeled datasets that are needed to train ML security models are rare. This restricts the learning approaches and potentially also the output of the frameworks [10, 8, 5].

Second, a real-time analysis of the 5G environments requires substantial computational resources, potentially impacting overall network performance. The creation and analysis of, for example, attack graphs, especially in real-time procedures, is limited by resources. This demonstrates how ML has scalability issues when applied across the network. The computational overhead of ML systems can create severe problems, especially in latency-sensitive applications such as URLLC in autonomous driving cars, where it must operate within milliseconds to be usable [10, 8]. As always, there is a trade-off between resources: quality and time. Models optimized for accuracy suffer in real-time execution or scalability. Faster ones often have lower detection rates [5]. This creates not only a technical challenge but also an ethical one, as resources are constrained.

Third, ML and AI-based threat prediction are known to work behind a black-box. Often, these new models are highly independent, which introduces interpretability challenges for us humans. This could be solved through explainable AI techniques, which are still an emerging field. In the future, ML models should have the ability to explain their operations and decisions, as for humans to be able to trust them completely. Going a step further, the black-boxing can hide tempering, making it difficult to detect if a model has been compromised [10, 1, 12, 8]. ML models themselves can become targets for attackers. The input to the ML can be crafted to mislead the model, creating loopholes in the whole system. This can be achieved through tainting the training dataset, but also by misleading it to ignore specific traffic patterns, creating an adversarial ML attack where security measures are compromised. [10, 1]. Lastly, ML models are not generalizable as of now. When trained in a specific 5G slice, it may perform poorly in another due to variations in traffic patterns and attack types [5].

9.4 Evaluation and Discussion of the case study

To be able to comprehend and utilize the previously gained knowledge, we are going to take a deeper look at the paper ”DDoS attack detection using unsupervised federated learning for 5G networks and beyond” by Saied Sheikhi and Panos Kostakos[11]. The paper shows how unsupervised federated learning allows multiple devices to collaboratively detect DDoS attacks on 5G networks while preserving

data pricacy[11]. For that, they created a 5G architecture that simulates a public network and focused on two specific DDoS attacks: SYN flood and UDP flood[11].

9.4.1 Approach

In Chapter 3, AI and ML Techniques for Cybersecurity, we examined various AI and ML methodologies within the scope of 5G security. The knowledge will be used to understand the paper and evaluate it according to other existing research. We will dissect the methodology, experiment, result analysis, and evaluation. The goal of the chapter is to review the current usage of FL in 5G security and evaluate its practicality by comparing it with other techniques addressed.

Currently, centralized ML is widely used for 5G security; however, detecting diverse types of attacks from edge to center is difficult on a large scale. As a solution to the problem, attempts are being made to develop AI-based methodologies considering the 5G architecture[8]. Federated Learning (FL) is especially well suited to the software-centric architecture of 5G, which includes computing on the edge devices from the user side. We chose to examine this paper as the authors created a safe testing environment that was used to experiment with a prominent threat. This chapter demonstrates to us how essential and promising ML is in 5G security, but also that there are still tests and improvements to be done.

9.4.2 Findings

This paper verifies the effectiveness of an FL-based approach for anomaly detection, particularly Distributed Denial of Service (DDoS) attacks, in the 5G domain[11]. The author implements two types of testbeds, each mocking a SYN flood and a UDP flood. Those are two common types of DDoS attacks that frequently occur in a network. In the first case, a SYN flood is created that targets the TCP protocol, sending fake requests without response to make the network overuse its resources. In the second case, a UDP flood attacks the UDP protocol, sending numerous small fake packets to multiple ports. It causes the network to exhaust its resources and be overwhelmed by the large traffic load in the system, creating a Denial of Service attack[11].

Each 5G core learns its DDoS attack pattern locally[11]. The local model sends learning parameters9.2 to the global model. The FL model executes federated aggregation by using autoencoders in unsupervised learning as a form of ML. After aggregation, the global model is redistributed to clients (5G cores). According to the paper, the federated model shows 99% accuracy in detecting DDoS attacks[11]. The key insight of the paper is that distributed learning using autoencoders within an FL structure effectively detects different types of DDoS attacks. Furthermore, the author expects the usage of ML in edge devices in a 5G network environment, where each local network can be used as learning material that will then be reflected on every device from the global core [11].

Table 9.2: List of extracted features from the network [11]

Feature name	**Feature name**
ip.len	frame.time_relative
ip.flags.df	frame.time_delta
ip.flags.mf	tcp.time_relative
tcp.port	tcp.time_delta
tcp.window_size	gtp.ext_hdr
tcp.ack_raw	gtp.ext_hdr.length
ip.fragment.count	gtp.ext_hdr.pdu_type
ip.ttl	gtp.ext_hdr.pdu
ip.proto	qos_flow_id
tcp.ack	pdu_ses_cont.ppp
tcp.seq	gtp.ext_hdr.pdu
tcp.len	gtp.flags
tcp.stream	gtp.flags.e
tcp.urgent_pointer	gtp.flags.payload
tcp.flags	gtp.flags.pn
tcp.analysis.ack_rtt	gtp.flags.reserved
tcp.segments	gtp.flags.s
tcp.reassembled.length	gtp.flags.version
http.request	gtp.length
udp.port	gtp.message
udp.length	gtp.teid

9.4.3 Evaluation

This paper aligns with the theory that Federated Learning is efficient for securing privacy and can be deployed for cyberattack detection by leveraging local training [**Alalyan2024**]. While most previous research has focused on the privacy preserving functionality of Federated Learning, this paper not only emphasizes the theoretical application in the security domain of the 5G network environment but also tests the performance of the learning methodology by applying it to two virtualized 5G core networks for realistic experiment simulation.

One of the biggest constraints is the lack of high-quality data. By using an ML model that uses unsupervised learning with autoencoders, they were not dependent on labeled data and were able to capture network traffic data directly from their own simulation. Through Federated Learning, the model was able to improve itself from 15-30% to 99%. The paper demonstrated the great potential of the model [11].

As a result, the study verifies that Federated Learning can be a practical solution for detecting DDoS attacks as well as privacy security[11]. However, the paper has some limitations. The experiment is executed on only two testbed environments; it is insufficient to represent the generality of the vest of 5G core network scenarios.

Although this paper shows the potential usage in diverse cybersecurity, the limited type of attacks of cybersecurity in 5G, DDoS, is not enough to evaluate the overall performance of Federated learning on the 5G network security.

9.4.4 Conclusion

"DDoS attack detection using unsupervised federated learning for 5G networks and beyond" contributes by verifying the utility of unsupervised federated learning in detecting DDoS attacks. Simulating the model on virtualized 5G cores enhances the reliability, but using only two clients limits the generalization of the result to large-scale cyberattacks. As a further development, diverse types of cyberattacks with a sufficient number of clients to represent a wide range of edge devices in 5G should be discussed.

9.5 Future Directions

9.5.1 ML integration in 6G

According to the Focus Group on Technologies for Network 2030[6] of International Telecommunication Union(ITU), 6G is forecasted to offer higher throughput, improved frequency usage, and support more advanced applications. This would facilitate, among other things, autonomous systems. AI and ML would extend all network layers, as argued by Kaur[7]. Looking at security, in the application layer, ML would enhance anomaly and intrusion detection through ubiquitous and continuous monitoring. However, AI and ML would not only be incorporated in this layer but in all of them. The result would be a network that is fully integrated and dependent on AI and ML[6]. Although it is a promising idea, there are some challenges to the integration of ML in 6G. One example would be the time complexity: since the goal with ML would be to develop real-time detection mechanisms, ML models would have to be able to respond faster. This could be solved with in-memory computation or edge computing[7].

Saad et al argues that the missing link for wireless revolution is the missing mimicking of human intelligence through artificial general intelligence (AGI). This would be possible through the metaverse. Through the emerging applications that are digital representations of the physical world, the missing link is provided. AI and ML models can train in their own world models, called digital twins, and even play scenarios out in them before acting. This will only be able in later states of 6G. He proposes gradual integration, where piece by piece ML gets integrated as its capacities grow. While these approaches are under research, they are still ideas[6].

9.5.2 Research Opportunities

Using ML algorithms, one could detect hidden correlations between features in 5G data to create a more efficient system. ML techniques not only enhance pattern recognition but also anomaly detection and automated threat intelligence. This could help developers find security gaps. Although already researched, these models still have limitations and need to be more researched before being completely integrated into 5G networks [1].

Furthermore, ML models should be more understandable for researchers to prevent black-boxing. The hidden layer in the thought process limits researchers' and developers' added insight into the models [1].

The biggest gap in the research represents the cooperation and coordination between the models themselves. As the models have their weaknesses and strengths and are deployed in different network layers, they present different issues. Hybrid ML frameworks would be able to resolve this issue but have to be created and tested first. At the same time, more research should focus on FL, as it is able to provide privacy-preserving and adaptive solutions [7, 1].

9.6 Conclusion

This paper provides a quick overview of state-of-the-art research on ML in 5G security. Throughout our seminar, we were able to examine both challenges and the potential that ML brings to the table.

The evolution from hardware-centric in 4G to software-centric in 5G environment has created a paradigm shift that extended capabilities but also introduced new security risks. ML offers promising approaches through different models to resolve said threats. Most notable was the added value of Reinforcement Learning, as it is able to adapt to evolving threats across different edge devices. Through our deeper analysis of a specific paper and its ML application, we were able to emphasize the significant potential such models can have in real-world scenarios. ML would imply not only a technical upgrade, but also a fundamental shift towards intelligent and adaptive networks.

The research suggests that one model alone cannot solve the security challenges, but instead a combined approach in which the models complement each other.

However, several challenges remain unsolved. To be able to deploy further ML models in our current 5G networks, problems such as black-boxing must be addressed and researched first. Researchers and developers must be able to understand a model in order to assess its security. Otherwise, ML integration could propose itself as a threat in 5G security. But not only through undetected malfunctioning, but also through undetected tempering. ML models can be fed incorrect or misleading data, creating a security loophole. Furthermore, ML introduces performance trade-offs. The models create a computational overhead, if performed

extensively, making URLLC applications useless. Therefore, ML used for now has to be balanced and have a lightweight architecture, as to fulfill complexity and performance requirements. The biggest problem still lies in the scarcity of high-quality, labeled security related datasets. As we saw in the discussed paper, a lot of research is done at scales below real-world 5G networks. The shortage limits the effectiveness of training models, perhaps even creating potential biases. Looking ahead, the integration of ML will be essential in 6G. Solving this limitations in the deployment of ML in 5G networks creates the pathway for 6G with fully integrated ML models at every layer. Although the research on Federated Learning in 5G networks seems promising, we would have liked to see more research on Reinforcement Learning and automation. We think that the decision-making of RL could orchestrate which ML model is needed at a specific time, to be able to keep the computational overhead as small as possible. It would be interesting to see empirical studies on whether RL keeps the ML integrated security models more lightweight and dynamic. We furthermore believe that future research should focus on ML integration protocols, as to minimize risks.

In conclusion, machine learning represents a necessary enhancement of 5G networks. Despite current limitations, they offer a promising path to resolve ever-evolving security threats.

References

[1] Amir Afaq et al. "Machine learning for 5G security: Architecture, recent advances, and challenges". In: *Ad Hoc Networks* 123 (2021), p. 102667. ISSN: 1570-8705. DOI: https://doi.org/10.1016/j.adhoc.2021.102667.

[2] Mohammed Al Asqah and Tarek Moulahi. "Federated Learning and Blockchain Integration for Privacy Protection in the Internet of Things: Challenges and Solutions". In: *Future Internet* 15.6 (2023), p. 203. DOI: 10.3390/fi15060203. URL: %5Curl%7Bhttps://doi.org/10.3390/fi15060203%7D.

[3] He Fang, Xianbin Wang, and Stefano Tomasin. "Machine Learning for Intelligent Authentication in 5G and Beyond Wireless Networks". In: *IEEE Wireless Communications* 26.5 (2019), pp. 55–61. DOI: 10.1109/MWC.001.1900054.

[4] Behnam Farzaneh et al. "DTL-5G: Deep transfer learning-based DDoS attack detection in 5G and beyond networks". In: *Computer Communications* 228 (2024), p. 107927. ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2024.107927.

[5] Venkateswarlu Gudepu et al. "Generative-AI for AI/ML Model Adaptive Retraining in Beyond 5G Networks". In: *arXiv preprint arXiv:2408.14827* (2024). URL: %5Curl%7Bhttps://arxiv.org/abs/2408.14827%7D.

[6] ITU-T Focus Group Technologies for Network 2030. *Network 2030 Architecture Framework*. Tech. rep. Accessed: 2025-05-25. International Telecommunication Union, 2020. URL: %5Curl%7Bhttps://www.itu.int/dms_pub/itu-t/opb/fg/T-FG-NET2030-2020-4-PDF-E.pdf%7D.

[7] Jasneet Kaur et al. "Machine Learning Techniques for 5G and Beyond". In: *IEEE Access* 9 (2021), pp. 23472–23488. DOI: 10.1109/ACCESS.2021.3051557.

[8] Manuel Eugenio Morocho-Cayamcela, Haeyoung Lee, and Wansu Lim. "Machine Learning for 5G/B5G Mobile and Wireless Communications: Potential, Limitations, and Future Directions". In: *IEEE Access* 7 (2019), pp. 137184–137206. DOI: 10.1109/ACCESS.2019.2942390.

[9] Roger Piqueras Jover and Vuk Marojevic. "Security and Protocol Exploit Analysis of the 5G Specifications". In: *IEEE Access* 7 (2019), pp. 24956–24963. DOI: 10.1109/ACCESS.2019.2899254.

[10] Tanujay Saha, Najwa Aaraj, and Niraj K. Jha. "Machine Learning Assisted Security Analysis of 5G-Network-Connected Systems". In: *IEEE Transactions on Emerging Topics in Computing* 10.4 (2022), pp. 2006–2024. DOI: 10.1109/TETC.2022.3147192.

[11] Saeid Sheikhi and Panos Kostakos. "DDoS attack detection using unsupervised federated learning for 5G networks and beyond". In: *2023 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*. 2023, pp. 442–447. DOI: 10.1109/EuCNC/6GSummit58263.2023.10188245.

[12] Jani Suomalainen et al. "Machine Learning Threatens 5G Security". In: *IEEE Access* 8 (2020), pp. 190822–190842. DOI: 10.1109/ACCESS.2020.3031966.

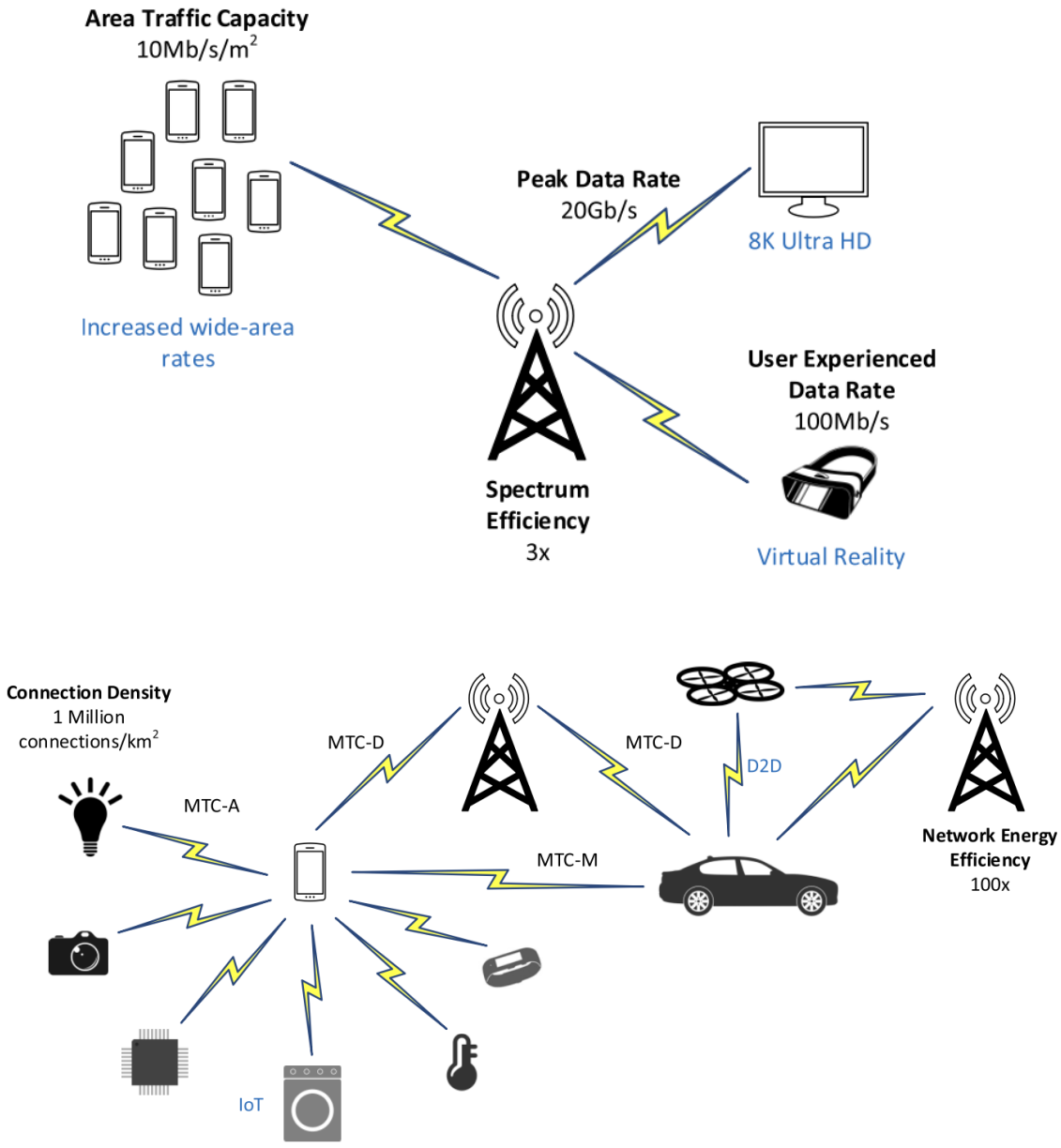


Figure 9.1: (Top) Enhanced Mobile Broadband (eMBB) applications; (Bottom) Massive Machine Type Communications (mMTC) applications [8]

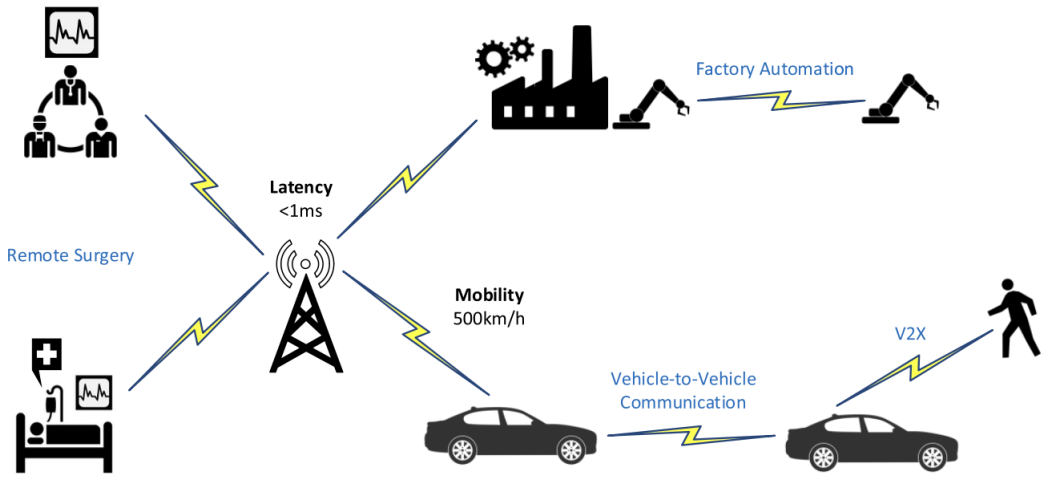


Figure 9.2: Ultra Reliable Low Latency (URLLC) applications [8]

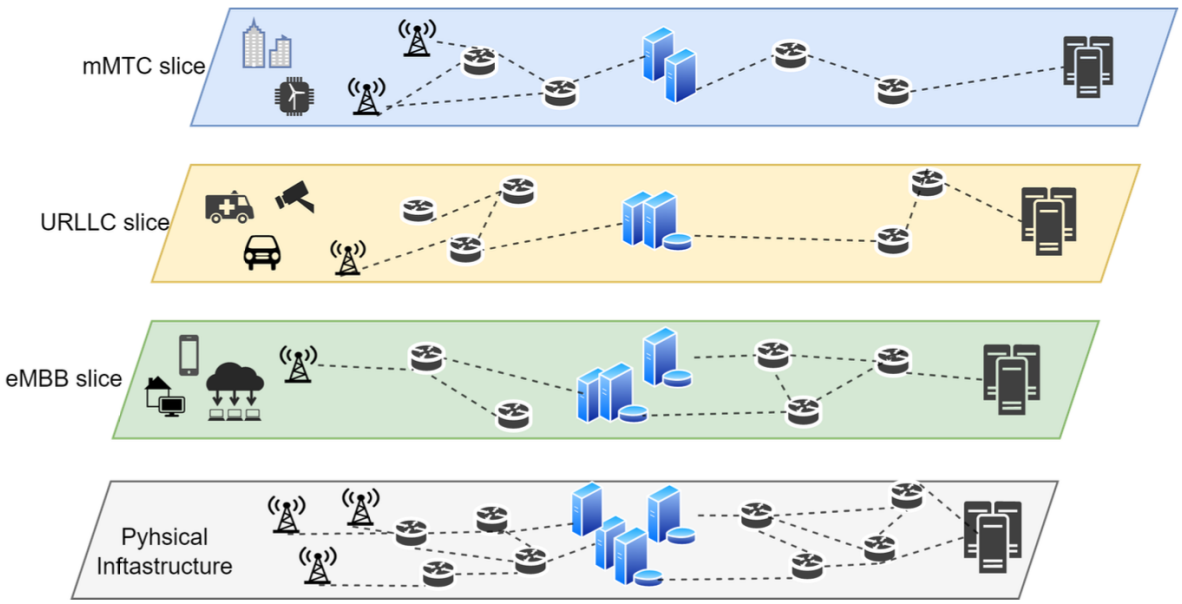


Figure 9.3: Simplified architecture of edge computing with network slicing [**Domeke2022**]

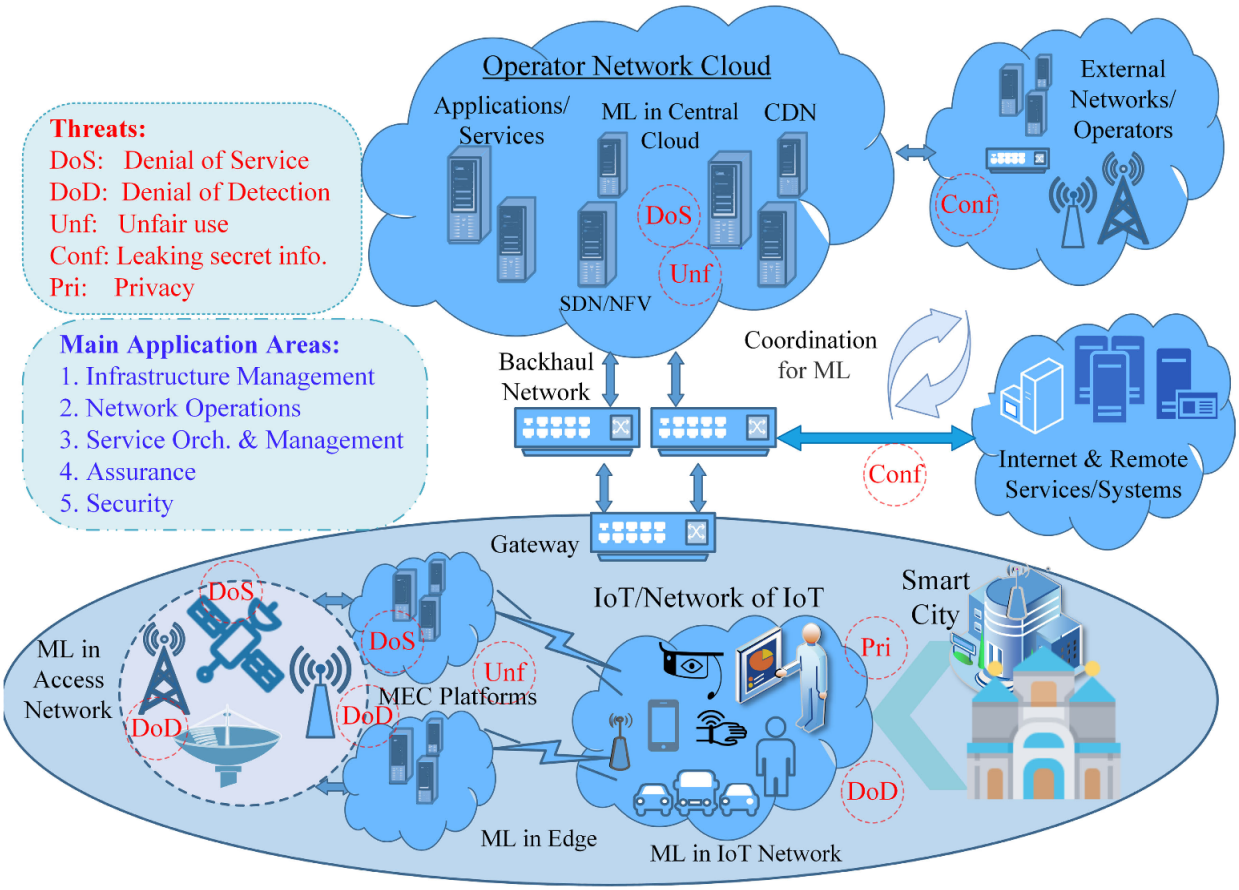


Figure 9.4: 5G Cyber Attack Surface [12]

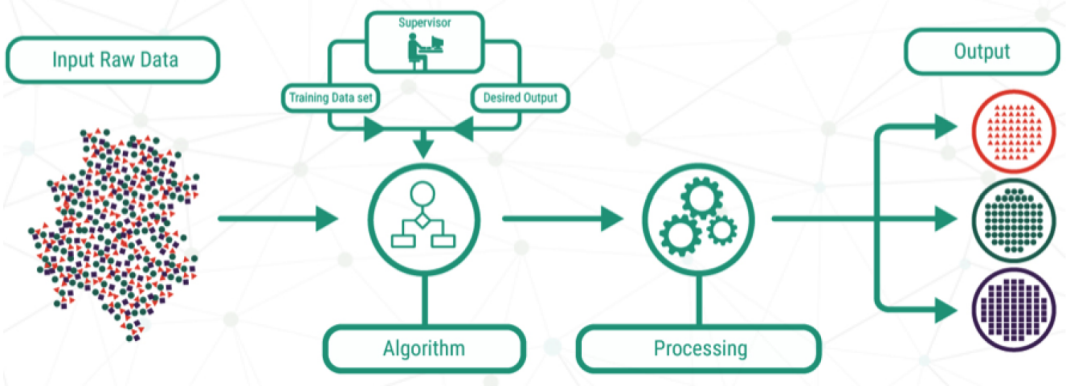


Figure 9.5: Supervised Learning Illustration [7]

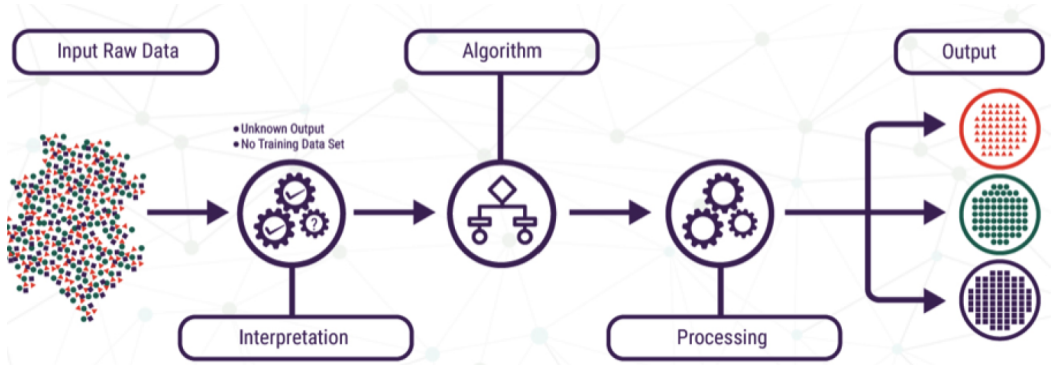


Figure 9.6: Unsupervised Learning Illustration [7]

Figure 9.7: Reinforcement Learning Illustration [7]

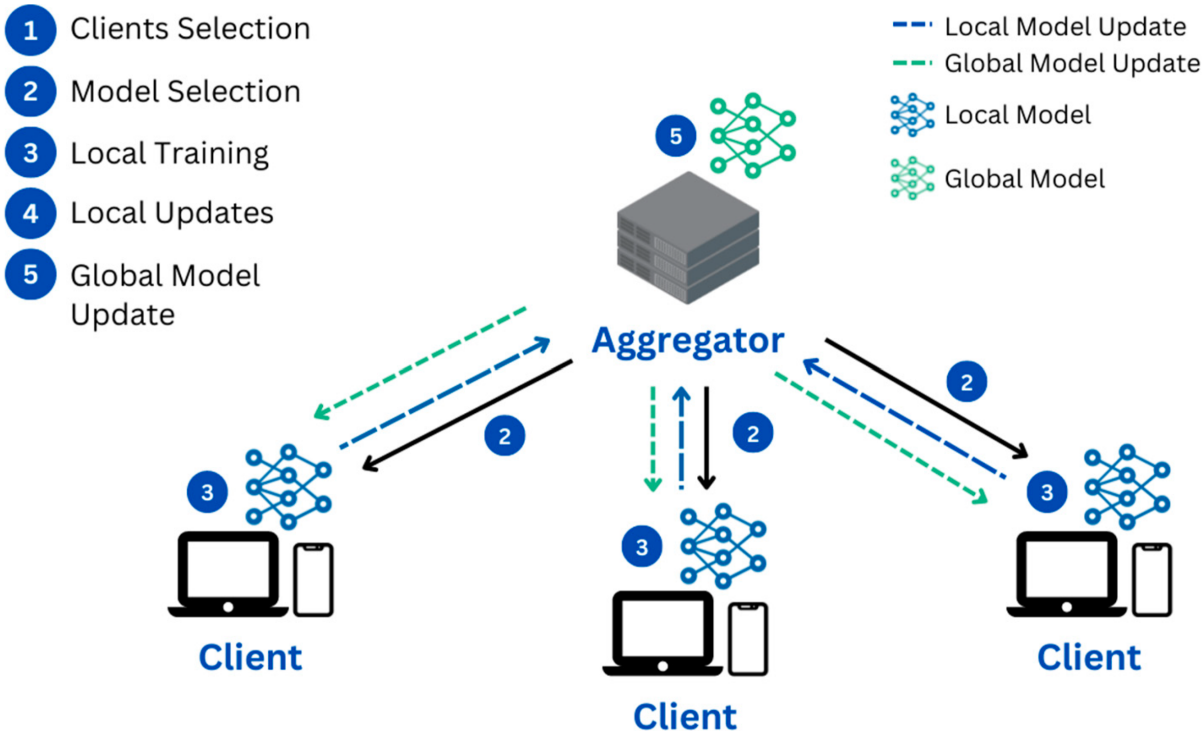


Figure 9.8: Federated Learning Illustration [2]

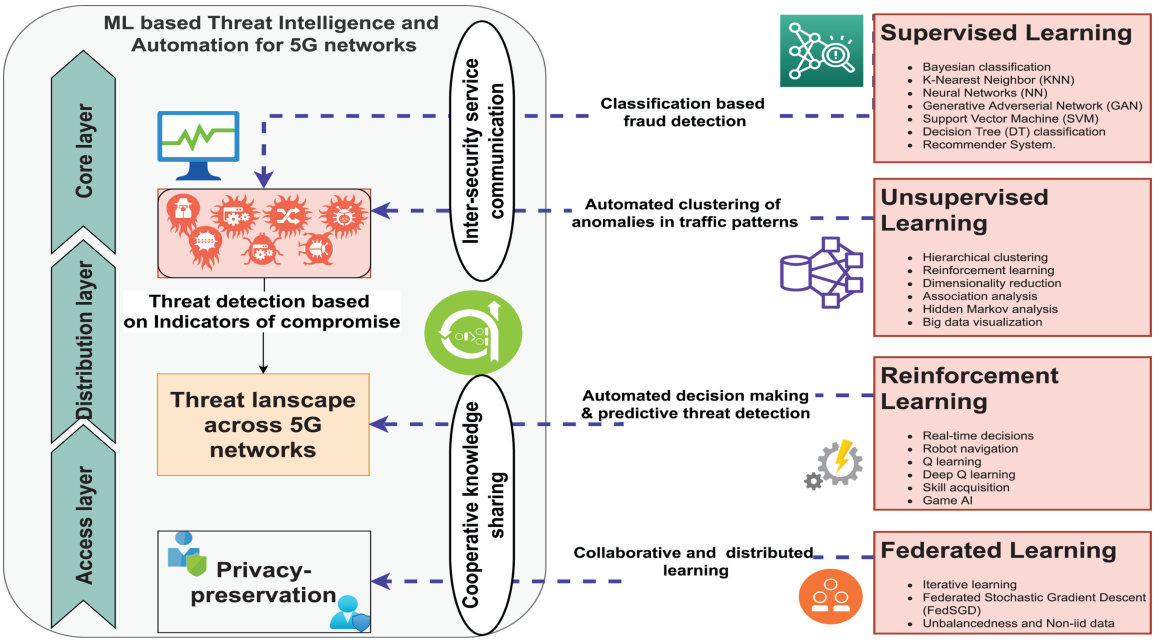


Figure 9.9: ML based Threat Intelligence and Automation [1]

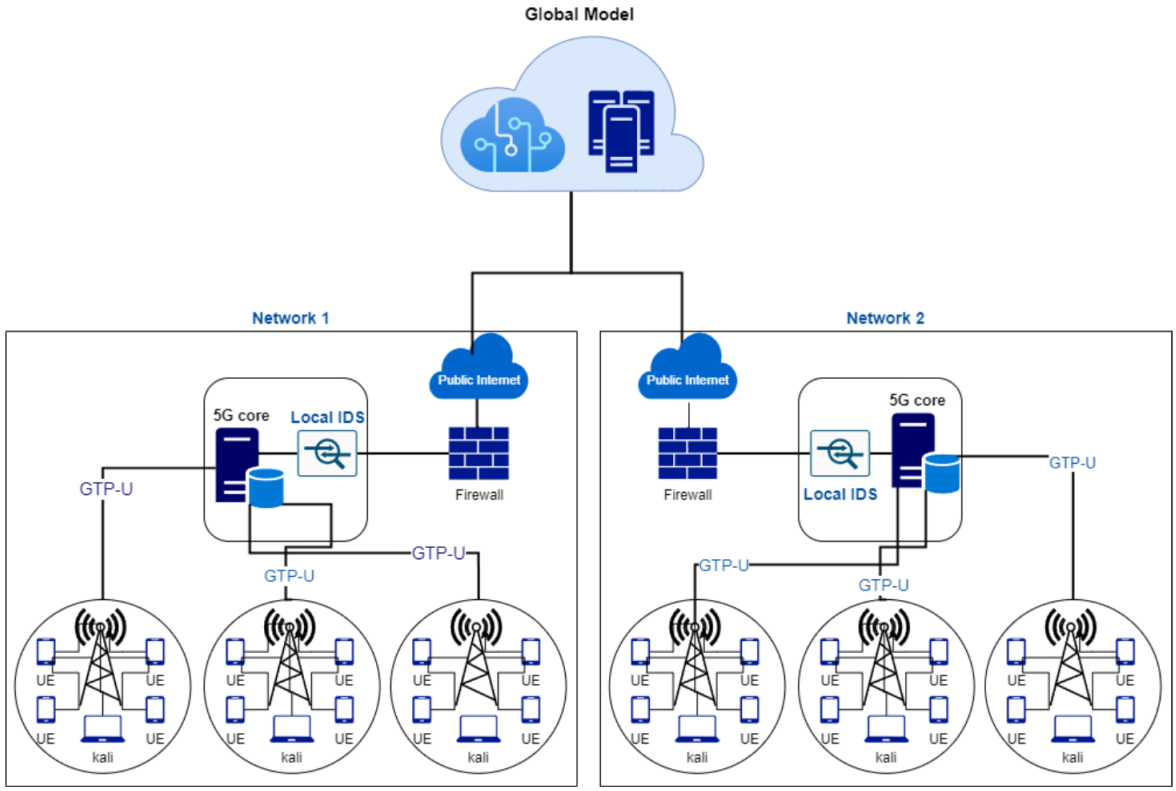
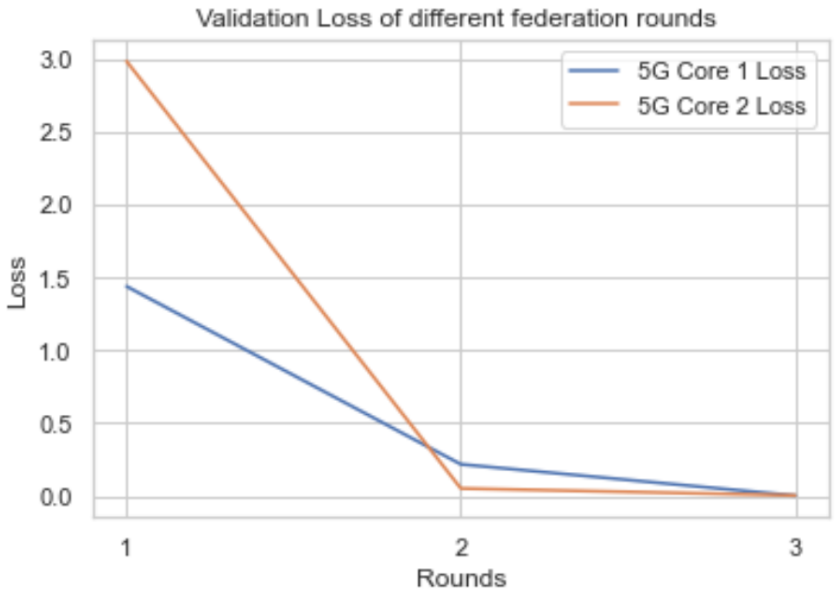


Figure 9.10: Architecture of testbed [11]



Client	Accuracy	F1-score
5G Core 1	0.9986	0.9988
5G Core 2	0.9988	0.9990

Figure 9.11: (Top) The loss of models in each round of federated evaluation; (Bottom) Performance evaluation of 5G core clients [11]

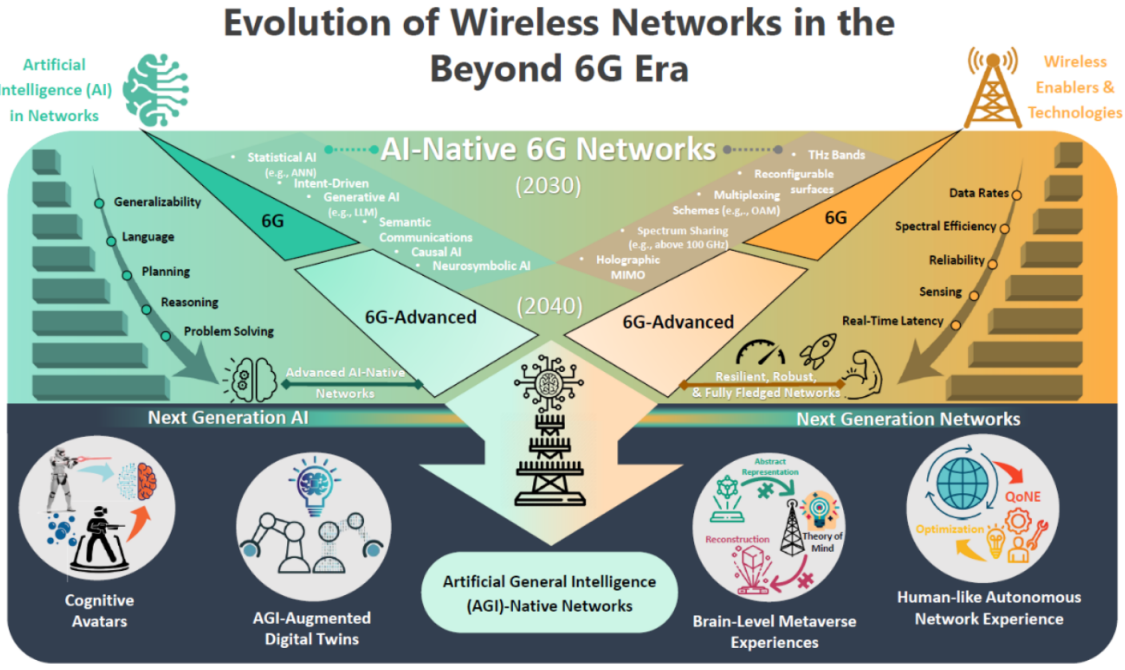


Figure 9.12: ML integration 6G [**VTNews2025**]

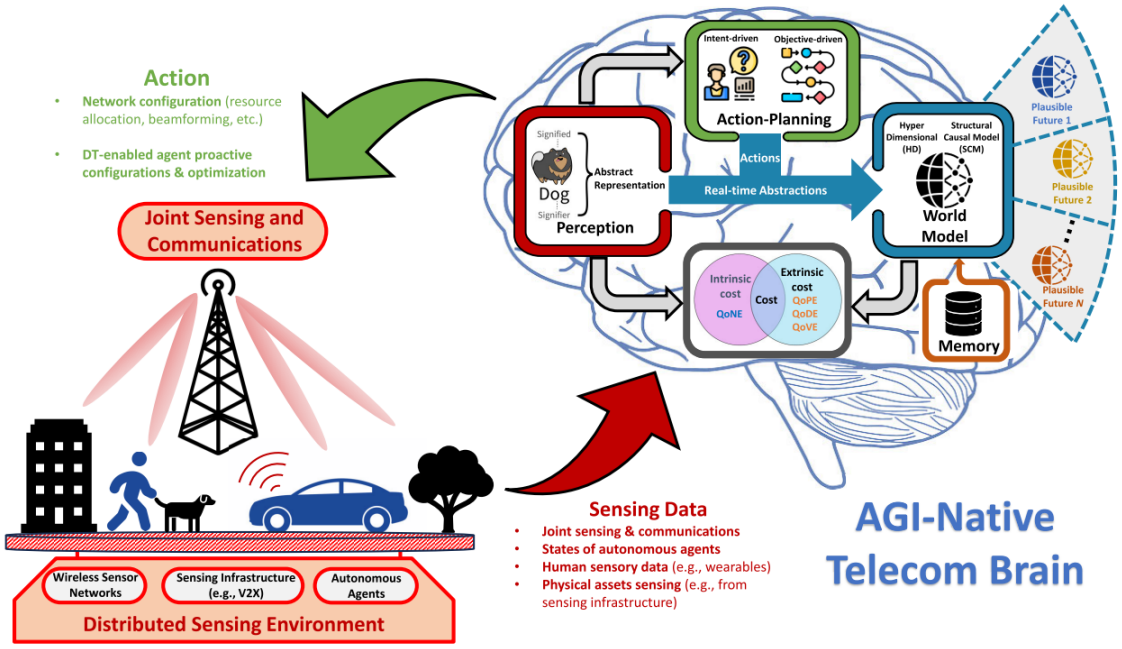


Figure 9.13: Artificial General Intelligence[**VTNews2025**]