DELFT UNIVERSITY OF TECHNOLOGY
FACULTY OF ELECTRICAL ENGINEERING, MATHEMATICS & COMPUTER SCIENCE DISTRIBUTED SYSTEMS GROUP

TRUSTZERO - OPEN, VERIFIABLE AND SCALABLE ZERO-TRUST

ADRIAN-TUDOR DUMITRESCU

STUDENT NUMBER: 5810825

THESIS COMMITTEE:
DR. IR. J.A. POUWELSE (THESIS SUPERVISOR)
DR. R.R. VENKATESHA PRASAD

TO OBTAIN THE DEGREE OF MASTER OF SCIENCE IN COMPUTER SCIENCE SOFTWARE TECHNOLOGY TRACK
TO BE DEFENDED PUBLICLY ON FEBRUARY 20TH, 2025

TrustZero - open, verifiable and scalable zero-trust

Adrian-Tudor Dumitrescu
Delft University of Technology
Delft, The Netherlands
A.T.Dumitrescu@student.tudelft.nl

Johan Pouwelse
Delft University of Technology
Delft, The Netherlands
J.A.Pouwelse@tudelft.nl

***Abstract*—We present a passport-level trust token for Europe. In an era of escalating cyber threats fueled by global competition in economic, military, and technological domains, traditional security models are proving inadequate. The rise of advanced attacks exploiting zero-day vulnerabilities, supply chain infiltration, and system interdependencies underscores the need for a paradigm shift in cybersecurity. Zero Trust Architecture (ZTA) emerges as a transformative framework that replaces implicit trust with continuous verification of identity and granular access control. This thesis introduces TrustZero, a scalable layer of zero-trust security built around a universal ”trust token” - a non-revocable self-sovereign identity with cryptographic signatures to enable robust, mathematically grounded trust attestations. By integrating ZTA principles with cryptography, TrustZero establishes a secure web-of-trust framework adaptable to legacy systems and inter-organisational communication.**

I. Introduction

In an era marked by intense global competition across economic, military, and technological spheres, the digital landscape has become a critical battleground. Nations and organizations worldwide are investing heavily in cyber capabilities to gain a strategic advantage, leading to a rise in cyber threats that target both government and private sectors [43]. A major concern in this landscape is the presence of “zero-day” vulnerabilities—previously unknown security flaws in software or hardware that lack any available defenses [21]. These vulnerabilities are highly valuable, often traded on a global market and exploited by state actors and criminal groups to infiltrate systems, steal sensitive data, and disrupt operations. One such example is the Israeli NSO Group that used spyware Pegasus for remote zero-click surveillance of smartphones for goals ”aligned with the geopolitical interests” [30]. Zero Trust Architecture has emerged as a response to this escalating arms race in cybersecurity.

Traditional security models that rely on perimeter-based defenses, such as Virtual Private Networks (VPNs) or firewalls, are proving inadequate against advanced, multi-vector cyber threats. This shift is underscored by the European Systemic Risk Board’s findings [9], which highlight a persistently heightened cyber threat landscape in Europe with sabotage of underwater telecommunications cables and disruption to systems in major financial institutions. In this sector, cyber risks have evolved in tandem with these threats. Attackers have become adept at exploiting complex system interdependencies to maximize damage, compelling financial institutions to elevate their security stance. Programs like the TIBER-NL [10] (Threat Intelligence-based Ethical Red-teaming) in the Netherlands are part of a proactive approach where regulated firms undergo simulated, controlled cyberattacks based on real-world threat scenarios. At the European level, the European Central Bank runs a similar initiative to ensure systemic stability by exposing potential vulnerabilities through rigorous, scenario-based testing [18]. Even the United Nations Security Council acknowledges the digital world has become a favorable field for espionage and cyberterrorism that creates ”mistrust and paranoia between nations” [48].

The proliferation of hybrid warfare has steadily penetrated societal activities, with even elections becoming a primary target for digital disruption. These attacks have been studied since the 2014 Scottish elections where ”fictional accounts of conspiracy theories” [13] spread misinformation and fear among citizens. This pattern of interferences peaked in 2024 with the Romanian presidential elections where cyberattacks sought to exploit vulnerabilities in the election IT system and influence people through fake accounts. This triggered the EU to search for a solution regarding bot activity and fraud by giving social media company TikTok a ”retention order that concerns national elections” [1]. Secret documents have been declassified to the public [6] and presented 25,000 accounts as part of a network on TikTok that became very active in the two weeks before the elections. From those, around 800 had existed since 2016, the year TikTok was released, but with almost no activity until November of this year. The Romanian Secret Services also observed that each TikTok account was associated with a unique IP address, indicating a deliberate strategy to obscure the true scale of the attack. The European Commission followed the press release with a formal proceeding against the company to ”assess and mitigate systemic risks” with the commission’s president stating that ”foreign actors interfered in the Romanian presidential elections by using TikTok” [2].

Zero Trust Architecture (ZTA) has emerged as a modern security framework grounded in continuous verification of identity and contextual access requests. ZTA, proposed by Kindervag in 2010 [32], follows the core principle that trust must never be assumed and that every user or device must be verified regardless. This new vision of the Internet prevents lateral movement intrusion, a type of attack encountered by big tech companies, based on the assumption that threats are omnipresent and no traffic can be trusted, including from internal networks [50].

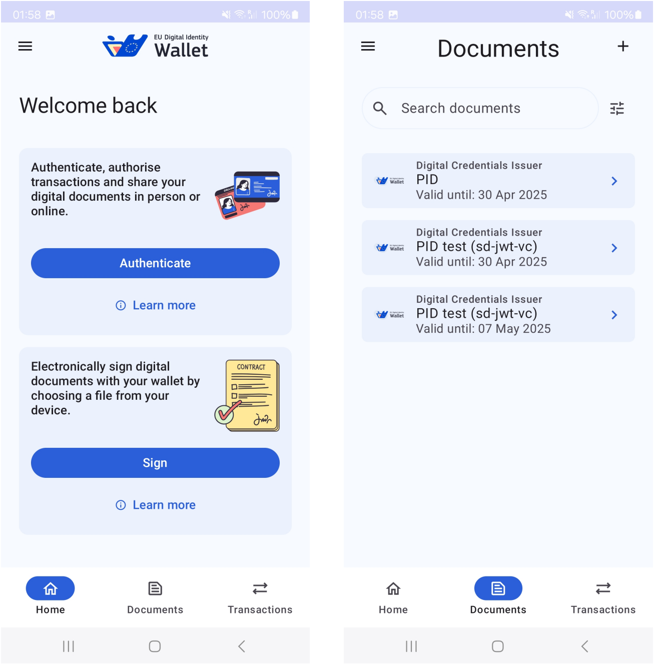
The criticality of a Zero Trust Architecture is further highlighted by high-profile breaches, such as Google's "Operation Aurora" incident in 2010. In this attack, Chinese state-sponsored actors exploited a zero-day vulnerability to infiltrate not only the original company but also Adobe and over 30 other major corporations [39]. The Operation Aurora case emphasized the risk posed by supply chain infiltration, where attackers compromise secondary suppliers of defense contractors to gain access to sensitive information. By exploiting the interconnected nature of global supply chains, adversaries can effectively bypass direct defenses, underscoring the need for a zero-trust approach that assumes no entity is trustworthy by default, regardless of its location within or outside the organization's network. Another such example of attack is Titan Rain, started in 2003 and targeting defense contractor computer networks in the US [46]. The attack source was identified as Guangdong, China, where perpetrators constantly changed IP addresses to make tracking their movements harder.

Current researches focus on moving from single-factor to multi-factor and continuous authentication, improving security while minimizing resource use [26], but most projects stop at the concept phase. This trend started with solutions developed by giant tech companies like Google's BeyondCorp. Motivated by the security breaches presented above, the company started a workflow transition to a protected zero-trust network, migration that proved to be harder than expected [24].

This thesis presents **TrustZero**, a layer of zero-trust security designed to be applied at large scale. As a proof-of-principle we created a universal ”trust token” that consists of a non-revocable self-sovereign identity with a list of trust attestations. The architecture relies on the ”trust token” flow between clients and servers during communication. We present a trust model/algorithm that is rigorously based on mathematical axioms with verifiable cryptographic signatures. By combining the zero-trust principles of continuous verification and cryptography we create a strong identity and web-of-trust framework that can serve as an upgrade for legacy communication infrastructure that can be applied between organisations. TrustZero is founded on the innovative principle of enabling trust to be both portable and verifiable across a global scale. In a world where the ”hybrid” war is escalating with denial of service and election interferences, we consider TrustZero would substantially make such attacks more difficult, costly and less effective.

II. The Fundaments of Zero Trust Architecture

Zero Trust Architecture fundamentally redefines cybersecurity by shifting from implicit trust to **continuous verification**. Every user and device must be authenticated and authorized before accessing network resources, reducing the risk of lateral threats. This model enhances data security and detects anomalies, making it a vital framework in today's cloud-driven, remote-access world. A key aspect of Zero Trust is its direct linkage to the first principles of digital identity, where the emphasis is placed on verifying the authenticity and integrity of digital identities in real-time. This means that identity is treated as the new perimeter, and digital identity verification becomes a cornerstone of security. ZTA helps ensure that only authenticated and authorized users or devices, with minimal privileges, can access critical resources, key requirements to provide passport-grade identity. TrustZero was developed to adhere to these requirements and to complement the European Digital Identity Wallet, presented in Figure 1, with a communicable, tamper-proof and verifiable token of trust.



Welcome back

Authenticate, authorise transactions and share your digital documents in person or online.

Electronically sign digital documents with your wallet by choosing a file from your device.

Documents

Digital Credentials Issuer
PID
Valid until: 30 Apr 2025

Digital Credentials Issuer
PID test (sd-jwt-vc)
Valid until: 30 Apr 2025

Digital Credentials Issuer
PID test (sd-jwt-vc)
Valid until: 07 May 2025

(a) EUDI home screen

(b) EUDI Digital identities

Fig. 1: EUDI wallet [40]

Building a zero-trust architecture starts from five basic assumptions that can solve the security problems encountered in large-scale supply-chain networks:

1) The network is constantly exposed to a hostile environment.
2) Threats, both internal and external nodes, persist throughout the network's operation.
3) A network's location alone cannot determine its trustworthiness.
4) Every device, user, and network traffic must be continuously authenticated and authorized.
5) Security policies need to be adaptable and dynamically recalculated based on a wide range of data inputs.

According to the National Institute of Standards and Technology (NIST), ZTA relies on three core logical components to enforce security policies. The Policy Enforcement Point (PEP) is the first of these, acting as an intermediary between the user and the server; it enables, monitors, and eventually terminates the connection between the subject and the resource, creating a boundary often referred to as the trust-zone. Closely collabo-

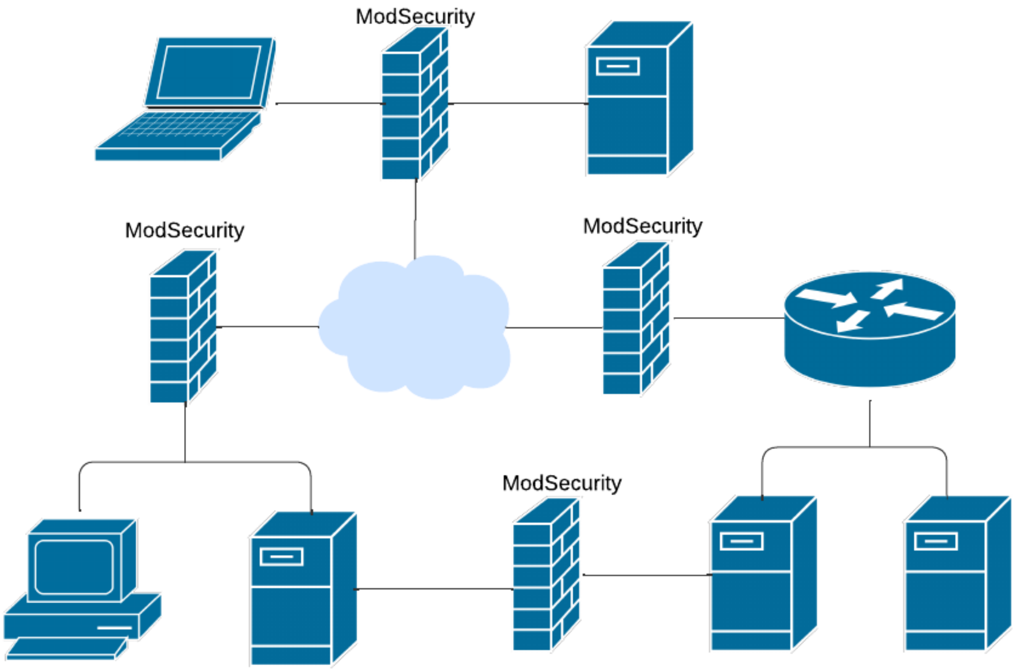


Fig. 2: TrustZero architecture overview

rating with the PEP is the Policy Administrator (PA), which is responsible for granting or denying access based on the PEP's assessments. Finally, the Policy Engine (PE) functions as the "brain" of the system, making access decisions by applying a trust algorithm to external inputs, in alignment with the organization's security policies [53].

In a digital identity-focused system, the policy engine (PE) plays a pivotal role in making access decisions according to business strategies. The trust computation is considered the cornerstone for access control in a zero-trust architecture and there is no universal algorithm agreed. Different approaches have been tried, from score-based/weight-based evaluations to fuzzy logic or graph theory model [54] but most of them prove to be too complex and hard to deploy on large-scale systems. Migrating to a zero-trust architecture is a complex task, as highlighted in [55]. Most trust computations rely on external information and when a provider encounters security or technical issues, finding a quick replacement can be challenging and costly. Requesting resources in a ZTA environment requires calculating a trust level based on credentials and information provided by the requester, then comparing it to the predetermined trust level required for accessing specific resources. Establishing an appropriate trust level for each resource is a complex task. The enterprise must find a balance in the trust levels—too high can make resources difficult to access, potentially hindering workflows, while too low may result in inadequate security. The trust algorithm represents the most common bottleneck regarding resource usage where complex dynamic scoring is based on multiple factors: login, network and operational behavior, user and device identities, user behavior, terminal security status and risks assessment ([14], [59], [57]).

One of the main concerns regarding ZTA is that, in fact, zero-trust is an impossible property to achieve for a system. As highlighted in a recent work [23], ”Zero Trust” is fundamentally unachievable for certain security properties in ”black box” devices—systems whose internal operations can not be fully inspected. Specifically, malware in such devices cannot be definitively ruled out, as verifications of security properties can only offer probabilistic assurances rather than certainties. Thus, complete trust elimination is impossible, and trust establishment becomes the practical alternative. Moreover, ZTA is considered impractical, demanding high assurance for all security properties of network devices, meaning their correctness must be proven rigorously. Instead, defenders focus on mitigating the cost of breaches rather than preventing them entirely, as low assurance and breaches are inevitable in real-world systems. This problem was reinforced by work in 2010 [49], expressing that, in security, trust can only be relocated not established. In relation to the impossibility of zero-trust, Lampson [34] stated that each part of prevention architecture is complex as a whole and there are always new threats, making security a fractal. Policy in such a system ”it has to be simple”, ”it has to minimize hassle for the user, at least most of the time” and ”it has to be true (given some assumptions)”, assumptions hard to achieve in ZTA.

III. Problem description

The field of zero trust security faces significant challenges related to transparency, openness, standardization, and reproducibility. Despite its growing adoption and the crucial role it plays in protecting sensitive infrastructures, the field lacks an open-source reference architecture or implementation that demonstrates exemplary zero trust security practices. This limitation extends to scientific research, where reproducibility is a cornerstone but remains underdeveloped in the context of zero trust systems. The absence of detailed descriptions of attack methodologies, defensive software architectures, security tooling, and policies contributes to an opaque landscape. This is true even in academia, where cybersecurity research and practice are often secluded. For instance, at institutions like Delft University, while other computer science faculties are accessible, cybersecurity departments maintain restricted access, highlighting an ingrained culture of secrecy. This is reflected in the methodologies in zero-trust where projects like OpenZiti [8] are using cryptographic principles such as low-cost hardware security modules (HSMs) [3] that leverage the laws of physics.

Moreover, the current body of scientific literature on zero trust is sparse in providing comprehensive, reproducible case studies or implementations. The seminal example of Google's response to the successful cyberattack Aurora illustrates this gap. Following the breach, a zero-trust model has been employed to isolate the legal, financial, engineering, and administrative networks. However, their documentation and public disclosures are described only at a high level([58], [17], [42]), lacking the granular details necessary for academic analysis and reproducibility.

Addressing this gap is critical not just for advancing zero trust methodologies and fostering an environment of shared knowledge but also to withstand the demands of modern cybersecurity. As stated before, the global competition is altering the digital landscape with attacks on critical infrastructure of finance and healthcare in vulnerable moments like the COVID pandemic [15] and started to interfere even in election systems. Moreover, the attack traffic is carried by many autonomous systems (AS) along the way, indirectly assisting illegitimate traffic, based on the assumption that all packets are trustworthy. There are multiple studies to detect and prevent malicious traffic through the network ([56], [19], [36]), but are based on specific patterns and scenarios(data-driven) and most require extensive computational time/power.

In contrast, Zero Trust Architecture emphasizes *continuous verification of every user and device, regardless* of their location within the network involving regularly reassessing the trustworthiness of all participants. Even though this architecture was introduced in 2010, to this day, no real-world open-source reference applications have been made [26]. Thus, the overarching challenge lies in developing a comprehensive, open-source zero trust framework with full transparency that is easy to reproduced in sensitive infrastructure.

IV. Our TrustZero design

To address the limitations of existing zero trust implementations, we designed TrustZero, a framework that leverages complexity as a strength rather than a vulnerability. This architecture is designed for large and complex supply chains bringing new trust between organisations. Traditional approaches often regard complexity as a source of fragility. This exposes systems to faults and weaknesses as demonstrated by critical incidents such as the exposed Log4j vulnerability [27]. Moreover, major disruptions can occur like the CrowdStrike 2024 global outage [41] determined by an intricate interaction with Windows systems. These incidents forced major organizations to focus ”more on proactive defense strategies, away from the traditional perimeter-based protection to continuous monitoring of the internal systems” [38]. In various high-stakes industries, including defense contracting, aerospace, and semiconductor manufacturing, complexity presents formidable challenges that can undermine system reliability. The semiconductor industry leader ASML, for example, relies on an intricate network of over 5,000 suppliers [5]—a testament to the depth and diversity inherent in modern supply chains. This complexity can also be observed in the automotive industry where companies like Audi recognize their responsibility to maintain a network of 14,000 of direct suppliers in over 60 countries [4].

TrustZero capitalizes on this intricate connectivity, redefining zero trust principles to operate beyond the confines of a single entity, organization, or government. Unlike traditional zero trust models that focus solely on securing an isolated system or entity, TrustZero is built on the novel principle of making trust portable and verifiable at a global scale. This approach facilitates trust exchange between various autonomous entities, enabling a unified, interoperable trust network. Collective intelligence is a unique and impactful way organizations work together, made possible by our architecture. For instance, approaches such as Byzantine-robust learning with compression [44] could be shielded by TrustZero from misinformation, deception, spam, and fraud. Exploring this direction is left as future work. More exactly, our key contributions are as follows:

1) We propose an architecture that is based on transparency and reproducibility of simple cryptographic functions to create trust between nodes. The protocol increases the security of communication to servers by adding client trust score evaluation.
2) We provide an open-source minimum-viable product of the architecture to demonstrate the correctness and functionality.

At the core of TrustZero is the concept of shared and transparent authentication histories, which support collaborative verification processes. By making successful interactions publicly traceable, systems develop greater resilience through emergent patterns and verifiable trust. This resilience is especially pronounced when critical components of the authentication network use high-assurance, passport-grade verification mechanisms, such as those compliant with eIDAS (Electronic Identification, Authentication and Trust Services) standards. This ensures that even machine-to-machine (M2M) communications are protected by strong, interoperable trust assurance. This links, as noted in a 2014 ASML co-authored publication [22], are traditionally implemented using protocols like MQTT that were not designed with inherent security and are known in being vulnerable [16].

As a proof-of-concept, we developed a universal trust token, featuring a self-sovereign identity with non-revocable trust attestations. We further enhanced a web application firewall, ModSecurity, with trust scoring, real-time threat signaling, and collaboration and placed it as a reverse proxy and gateway for each server present in the network to form a global web-of-trust. Lastly, we present a global trust model rooted in mathematical(cryptographic) principles, combining the zero-trust elements to create a robust identity and trust framework. The high-overview architecture of TrustZero is represented in Figure 2.

A. *Protocol*

The TrustZero Protocol exemplifies a decentralized approach to secure trust verification in multi-party systems, addressing critical challenges in integrity and authenticity. In this protocol, a user generates a public/private keys ($(\mathsf{pk}_u, \mathsf{sk}_u) \leftarrow \mathsf{KGen}$) and sequentially interacts with multiple servers, each generating its own pair ($(\mathsf{pk}_{sn}, \mathsf{sk}_{sn}) \leftarrow \mathsf{KGen}$). The user initiates communication by sending messages and their public key to the servers, where each server signs the user's public key using its private key, producing signatures ($sign_N \leftarrow \mathsf{Sign}(\mathsf{sk}_{sN}, \mathsf{pk}_u)$). Each server processes 2 sequential steps ensuring both correctness and accountability:

1) verifies all the previously generated signatures by other servers
2) issues its own signature if the previous step passed(or is renewing it)

The resulting "trust token", a concatenation of all signatures, serves as verifiable proof of trust, validated against the servers' public keys. By distributing the signature generation and verification processes, the protocol eliminates the reliance on a central authority, enhancing security and trustworthiness. The protocol's integration of cryptographic keys, digital signatures, and decentralized verification provides a robust framework for securing trust in systems vulnerable to adversarial threats, making it suitable for applications in distributed supply-chain networks and secure communications.

B. *Trust token*

In our zero trust architecture, the trust token is the key element that enables a non-revocable, self-sovereign identity with a list of trust attestations. In this identity model, users maintain full ownership and control of their digital identity without third-party oversight or the risk of revocation. This identity is accompanied by a collection of trust attestations—verified endorsements or credentials—that validate the user's identity and reputation.

In a self-sovereign, zero-trust architecture, users maintain complete control of their trust tokens without relying on external cloud storage. The core design choice lies between transparent trust communication and an explicit trust protocol. Trust is communicated passively by embedding a session request header and token signatures can be verified and added as a layer of security without altering existing server protocols.

The list of trust attestations present in a trust token is a series of signatures that the user received from past (adequate) interactions with other nodes. These signatures can be verified by any server with simple cryptographic functions against the public key of the user with the public key of the issuing server. In exchange for a "good" request, the server will issue a new trust token for the user with a new signature (or update the present one if they already interacted). The trust token sent in each request will be structured in the form:

$$token = sign_{server_1} || sign_{server_2} || ... || sign_{server_N} \quad (1)$$

The EUDI app from Figure 1 uses a traditional JSON web token to exchange information and verify identities for access. This approach proves to be rigid regarding token revocation [29] with a bad actor being able to use it even after detection. The trust token used by TrustZero can be revoked by any server after any abnormal access, lowering the user's reputation. Moreover, a server can change its key pair to invalidate all its signatures and indirectly signal the user to all other servers that will interact with it.

C. *Trust algorithm*

The trust algorithm implemented in TrustZero is score-based, evaluating the trustworthiness of entities and assigning a numerical score that reflects their reliability. The value is computed by verifying the number of valid signatures a user has on its public key. The crucial objective is to make it as expensive as possible to set up, maintain, and/or exit fake identities and give priority to trusted parties. To achieve this, the architecture exploits a resource that most attackers do not have: time. With TrustZero scoring system, trust is gained over time and interaction with unique servers, reducing the surprise of an attack (like denial of service for example).

The trust computation is constructed on basic cryptographic functions that require low resources and are easy to understand, from administration entities to simple users. This algorithm is executed before a request even gets to a server not interfering with its normal behavior. In the end, the TrustZero algorithm for a user (pk_U) that has signatures from n servers(S) can be summarized as:

$$\begin{aligned} trustscore = {} & Vf(sign_{S_1}, \mathsf{pk}_{S_1}, \mathsf{pk}_U) + \\ & + Vf(sign_{S_2}, \mathsf{pk}_{S_2}, \mathsf{pk}_U) + ... \\ & ... + Vf(sign_{S_N}, \mathsf{pk}_{S_N}, \mathsf{pk}_U) \end{aligned} \quad (2)$$

where $Vf()$ is a cryptographic function that returns 0 or 1 based on the validation of a signature using the public key of the originator server against the public key of the user.

D. ModSecurity

ModSecurity [45] is an open-source web application firewall (WAF) used to monitor, log, and filter HTTP traffic to prevent attacks on web applications. It acts as a security layer between users and web servers by inspecting requests and responses based on customizable rules, helping to detect and block threats such as SQL injection, cross-site scripting (XSS), and other vulnerabilities. ModSecurity can also be used for real-time monitoring, auditing, and compliance with security standards, making it a vital tool for enhancing web application security.

In TrustZero, ModSecurity is deployed in reverse proxy mode where it acts as an intermediary between clients and servers, inspecting all inbound and outbound traffic before it reaches the destination web server. This setup allows ModSecurity to enforce security rules, log traffic, and block attacks without modifying the web server itself, acting as a Policy enforcement point.

The capabilities of ModSecurity in detecting and mitigating different attacks have been extensively studied in state-of-the-art literature ([28], [51], [11], [33]). In our architecture, its most important characteristic is the non-disruptive nature of inspecting requests while enabling external scripting processing. The signatures present in a request are verified by ModSecurity which computes the score and can deny the communication (if signatures are wrong/compromised) or forward it to the servers. ModSecurity offers the perfect environment with its custom rules to adapt the security level(paranoia) for each request depending on its nature. While computing the trust score is studied in this thesis, its utilization in raising different security levels based on it is left for further research.

V. Related work

Although Zero Trust Architecture (ZTA) was first introduced over 10 years ago, no comprehensive, real-world implementation has emerged that fully addresses its potential. Most ZTA proposals remain in the design phase due to the complexity of their trust models and the need for substantial changes in infrastructure to accommodate them. Additionally, these solutions are often narrowly focused on specific business sectors, such as cloud computing, IoT etc, rather than providing a universal applicable framework applicable. This section will review these ZTA and security architecture proposals, highlighting the need for a more generalized, adaptable approach.

A. First steps towards Zero-Trust Architecture

TrustGuard [52] model serves as an intermediary security layer that enforces strict access controls, monitors communication between entities, and validates interactions in real-time. Designed to reduce the risk of unauthorized access and lateral movement, TrustGuard bridges the gap between traditional network models and the zero-trust paradigm by establishing micro-boundaries of trust that are dynamically managed. It introduces a flow-level reputation-based defense mechanism and it was proposed as early as 2005 as a first step towards reputation and trust management networks. Unlike traditional methods that typically focus on IP addresses or individual packet characteristics, TrustGuard evaluates the reputation of entire network flows. Over time, it evolved in more specific uses cases such as allowing for more precise identification and mitigation of Distributed Denial of Service (DDoS) attack traffic [35] while reducing the incidence of false positives.

The architecture of TrustGuard encompasses several integral components. The flow collector is responsible for gathering detailed flow-level data, encompassing both traffic characteristics and behavioral patterns. The reputation manager analyzes this data to compute reputation scores for each flow, leveraging historical behavior alongside real-time observations. The decision engine then utilizes these reputation scores to make informed traffic filtering decisions, effectively distinguishing between legitimate and malicious flows. Additionally, a feedback loop continuously refines the reputation scores based on observed behaviors, enabling the system to adapt dynamically.

By incorporating machine learning techniques, TrustGuard enhances its ability to adaptively modify reputation scores in response to shifting traffic patterns and evolving attack characteristics. This combination of advanced analytics and real-time data processing positions TrustGuard as a robust solution for modern network security challenges.

B. SDN and zero trust architecture

A novel solution is presented in combining Software-defined networks(SDN) with zero-trust principles([25], [60]), a security architecture designed to address the complex requirements of Industrial IoT systems, which include real-time operations, reliability, and decentralization. Traditional cybersecurity solutions struggle with the heterogeneity of IIoT devices. The proposed architectures leverages network micro-segmentation and integrates Software-Defined Networking (SDN) for policy enforcement, alongside a centralized security management layer for simplified control. A prototype demonstrates that this system ensures decentralized, resilient, and flexible security management while maintaining central oversight of security policies and network topology. One proposal [60] uses Nebula, a software-defined overlay network solution, in an abstraction layer for policy enforcement. This tool relies on a custom Public Key Infrastructure (PKI) system since it uses certificates that are not X.509 compliant.

Nebula introduces challenges in integrating with standard security frameworks and requires the development of a unique Certificate Authority (CA). Custom PKI solutions increase the complexity of managing certificate requests and key generation, already a demanding task, which may lead to security vulnerabilities. Additionally, isolating the Nebula network, while enhancing security, could introduce maintenance and scalability issues. Moreover, it is acknowledged that some devices might lack native support for Nebula and need to be integrated by introducing an additional device.

C. BeyondCorp

The "BeyondCorp" [58] model represents a paradigm shift in enterprise security, moving away from the traditional perimeter-centric approach. Developed by Google, it emphasizes user and device authentication regardless of location, allowing secure access to applications without a VPN. BeyondCorp relies on continuous verification through context-aware policies, integrating real-time monitoring and adaptive access controls to enhance security. The BeyondCorp model emphasizes secure device and user identification through a comprehensive management system. It maintains a Device Inventory Database to track managed devices, which are uniquely identified via device certificates stored in secure modules. User access is managed through a User and Group Database, integrated with HR processes, and authenticated via a Single Sign-On (SSO) system, which issues a session token for the access of a specific resource. Additionally, BeyondCorp establishes an unprivileged network that mimics an external network, enhancing security by minimizing trust in the internal network infrastructure.

As one of the few practical examples of ZTA, the model faced challenges during the later stages of Google's BeyondCorp migration, particularly regarding difficult use cases that did not fit the standard HTTPS-based workflow. Issues were signaled with specific applications that required IP-layer connectivity or could not easily integrate with the BeyondCorp access proxy [24].

D. Zero trust in cloud computing

Another discussed topic for ZTA is its use in cloud computing where services such as storage, processing power, databases, networking, software, and analytics are delivered over the internet. Thus, safeguarding such critical resources is key and the zero-trust design appears to satisfy the security requirements. For example, strategies with 9 principles of trust have been proposed but formulated just as a "conceptual model" [37] for further research.

A novel concept presented for cloud computing is "survivable zero trust". Unlike existing models, the proposed architecture [20] acknowledges that trusted components can be compromised. The novel survivable zero trust architecture ensures high security, robustness and can tolerate intrusions and recover from failures, making it suitable for cloud environments under specific conditions. The design is also based on a key pair and signature scheme that assists the communication against different attack scenarios. Even with a strong trust model, the paper recognizes that designing an effective protocol that ensures confidentiality while minimizing performance impacts and disruptions remains an open research challenge.

E. Zero-trust and Blockchain

Combining zero-trust and blockchain can enhance security in distributed systems addressing challenges such as identity management, secure data sharing, and ensuring compliance in decentralized environments. This movement materialized with projects like ZEBRA [12], a framework that focuses on securing Advanced Metering Infrastructure (AMI) using a Zero Trust Architecture combined with blockchain technology and Ring Oscillator Physical Unclonable Functions (ROPUFs). The design ensures robust device authentication and guarantees data integrity by leveraging the unique properties of ROPUFs, for generating unclonable keys, and blockchain for traceable and tamper-proof communication. This approach enhances the security of smart grid networks, providing resilience against cyber threats like unauthorized access, spoofing and data manipulation. Nevertheless, blockchain technology can introduce latency and require significant computational resources, which may be challenging for the resource-constrained devices used in AMI. Additionally, the reliance on ROPUFs for authentication, while secure, could be affected by environmental factors (such as temperature or voltage variations), impacting the reliability of the cryptographic keys generated. Managing these factors while maintaining system performance could pose challenges for real-world deployment.

Another solution, this time tailored for IoT is Amatista [47], a blockchain-based middleware designed for scalable management of IoT networks. The paper is the first to enumerate cryptography as an option in trust management but it incorporates it in the blockchain consensus algorithm. As IoT expands rapidly, the trustworthiness of millions of connected devices becomes a challenge. Amatista tackles this issue by employing a zero-trust approach, utilizing a novel hierarchical mining process to validate both the infrastructure and transactions at varying levels of trust. By leveraging blockchain features such as a distributed database, consensus mechanisms, smart contracts, and immutability, Amatista ensures reliable transactions without centralized validation nodes. The system is tested on Edison Arduino Boards, demonstrating how it can address trust concerns in IoT through decentralized validation mechanisms. While Amatista shows promise, potential issues include reliance on complex blockchain infrastructure, scalability challenges with numerous devices and, as stated by the authors, not yet tested "in a large scale loT deployment".

VI. Implementation and Experiments

Our experiments focus on an in-depth exploration of real-world systems applying zero trust principles, combined with a comprehensive performance analysis of our novel TrustZero token. The key contributions of this work lie in demystifying the opaque security practices of major corporations and identifying the practical implications of deploying an open, verifiable zero trust system. Given the absence of a comprehensive zero trust solution that embodies end-to-end openness, open-source availability, and the potential for uncompromised self-hosting, our experiments are necessarily exploratory and integrative, evaluating various components to construct a robust, open framework.

The protocol outlined has been implemented in Python as a proof of concept. The open-source code is available on GitHub [7]. The Cryptography library is utilized for public/private key generation, as well as for signing and verification. A simple login server has also been created to accept (or deny) requests

```
POST /resource HTTP/1.1
Host: api.example.com
Content-Type: application/json
User-Key-Signatures:
pk_user:1:5d41402abc4b2a76b9719d91017c592
Content-Length: 47
{
  "username": "John Doe",
  "password": "johndoe",
}
```

Fig. 3: POST Request example

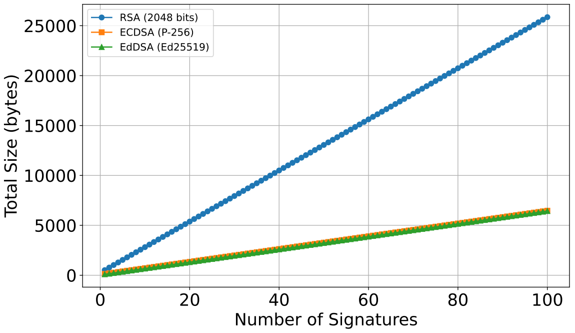


Fig. 4: Signatures size

from users deployed alongside a ModSecurity instance acting as a reverse proxy for it. An example of POST request that was used in the experiments against the deployed server is presented in Figure 3. The pair server-ModSecurity has been deployed with docker in multiple containers to simulate a distributed network of servers having their own ports and key pair. All traffic directed to the server is intercepted by ModSecurity who inspects the header of the request and searches for the trust token and public key. The rule related to this actions is defined in the following way:

```
SecRule &REQUEST_HEADERS:User-Key-Signatures \
    "eq 0" \
    "id:10009, \
     phase:1, \
     t:none, \
     msg:'Missing User-Key-Signatures header', \
     deny"
```

In the case of a 'cold start', a user who has no signatures yet will only send his public key and will be treated accordingly with a score of 0.

Once the header is detected, ModSecurity inspects it to determine the number of valid signatures present and to asses the trust score of the user. The rule makes use of a Python script with cryptographic functions for verification and in case of an error signals ModSecurity to terminate the request:

```
SecRule REQUEST_HEADERS:User-Key-Signatures \
    "@inspectFile /app/check_signatures.py" \
    "id:10010, \
     phase:1, \
     msg:'Error in signatures', \
     deny, \
     t:none"
```

If the request has all the correct signatures, it is passed to the server who can solve it accordingly. In the response header, the server attaches his signature to the token and passes it one more time through ModSecurity. After checking its presence, the response is forwarded to the user who can now store the signatures for further communication.

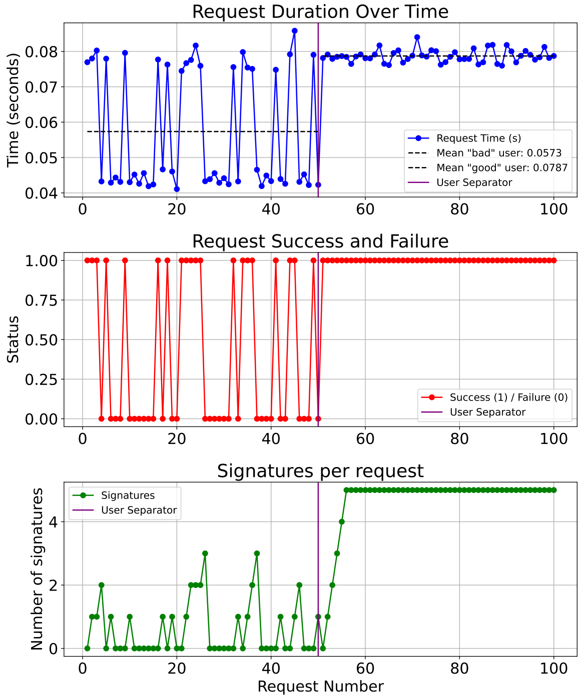


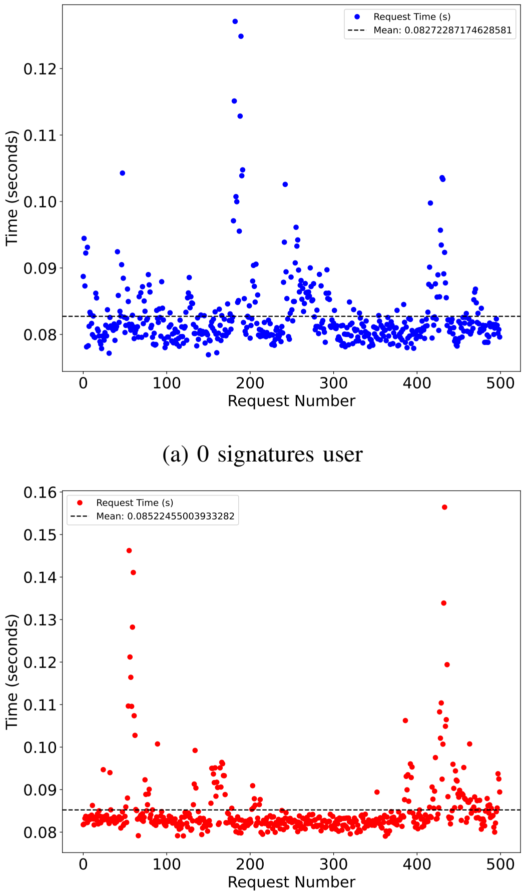
Fig. 5: First experiment

While the usage of the score is not implemented in this POC, it can be stored on every server as a table of public keys with their scores, every server having a snapshot of the network at a certain point I. ModSecurity has multiple score-based parameters(anomaly level, paranoia level) that can be set based on the request score to treat user based on their reputation.

Public Key	Score
PK1	5
PK2	2
PK3	3
PK4	0

TABLE I: Table of public key scores.

TrustZero uses the RSA cryptographic algorithm to create the public/private pair with a public key size of 256 bytes. The signature resulting from it inherits this dimension and creates a linear growth. To address a potential size issue, other signature algorithms such as Elliptic curve P-256(64 bytes public key) or EdDSA Ed25519(32 bytes public key) can be used. Both options are present in the Cryptography library in Python resulting in the same size of signature of 64 bytes. Adding the public key itself to the token dimension, the comparison,



(a) 0 signatures user

(b) 5 signatures user

Fig. 6: Signature latency

calculated up to 100 signatures, is presented in Figure 4. As highlighted in future experiments, the signatures did not present a significant overhead in communication and can also be limited to a maximum number to resolve the potential drawback. All the following requests and investigations are produced using simple RSA signatures.

For the experiments, 5 servers were deployed with docker, each container consuming as much CPU as it needed from the host. This setup was hosted on 2 different specifications: low resources (8 cores Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz) and high resources(16 cores AMD Ryzen 7 6800H 4.7 GHz). All the experiments were successfully run on both specifications showing the low overhead of TrustZero even with a low-grade CPU. The subsequent results were used from the best-performing setup. One of the first experiments was to test whether any attacker that is tampering with the signatures is detected and their requests are denied. Two users were created sequentially to communicate with all the servers sending 50 requests and the results collected from all instances are presented in 5. In the second half of the figure, a genuine user is building the maximum trust over time(5) and, even though having a higher average latency, does not encounter a high variability in times and better predictability. In contrast, once a user behaves abnormally, like in the first half of the plot, and sends corrupted signatures he loses his reputation and access to the resources and needs to build back the trust by interacting with the servers. In addition, every time a token is sent with modified signatures, the application firewall refuses immediately the call before it gets to the server, observation reflected in the low latency present in unsuccessful requests.

ModSecurity provides multiple automated defense mechanisms, starting from rate limiting to IP blacklisting integration. In a previous work regarding attack mitigations using WAFs [31], ModSecurity has been deployed to protect a server against a DDoS attack. Using his integrated IP blocking functionality based on a text file access, at the header level the overhead was measured at 509 microseconds. In TrustZero architecture, the trust token present in a header is parsed to an external script that signals ModSecurity the successful processing. This creates a higher amount of latency as there is no integrated functionality to process headers and the whole request is transferred to a python script. Based on the logging provided by ModSecurity in processing rules,

```
[/login][4] Operator completed in 79255 usec.
```

we observed that the number of signatures in a trust token is not influencing the processing time of the rule as presented in Table II.

Number of signatures	Processing time (in microseconds)
0	79255
1	87352
2	85477
3	73568
4	85917
5	81200

TABLE II: Processing time of signatures

To better understand the implication of signatures in response latency, measurements of time were taken from 2 specific cases: a user with all 5 server signatures and one with no reputation. 500 requests were created and resolved and the difference in computational time was around 0.025 seconds, presenting a low latency of adding signatures in the request.

A box plot 7 was created based on the latency set measured for better visibility of the data and its spreading. From the figure, both groups have similar distributions, but the user with more signatures shows a slightly higher median and fewer outliers compared to the plain one. In both cases, the number of values above the 95th percentile is small compared to the number of total requests reflecting a low variability over a large number of messages.

The largest experiment measured was deploying in the network up to 2000 users in different threads, adding every 2 seconds a new instance and measuring the latency of a legitimate user. This experiment can be considered as a ”DDoS” attack launched over time trying to collapse the servers with repetitive and identical requests. As all instances are created in

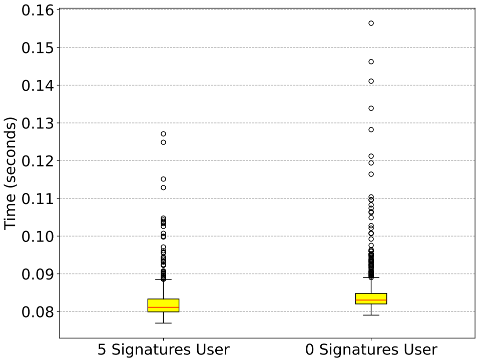


Fig. 7: Box plot signatures latency

the same computer(including servers), the CPU and memory usage might influence the request time as more users flood the network. The test measurements were split after every 200 users were added and 3 phases can be identified:

1) For the first 300 users the request times are relatively constant without big spikes of latency keeping the latency below the experiment mean
2) From 300 to 1800, the latencies fluctuate significantly more and increase in mean time exponentially
3) After 1800 users, the requests are resolved slowly (up to 35 seconds); the minimum latency is significantly higher than the other phases

Moreover, for the first 200 users, the request times reported stay consistent or lower with the measurements of previous experiments(of around 0.07 seconds). A considerable mass of low latencies is still present at up to 600 users. For better visualization, a moving average trend line was created to fit the growth tendency, being updated with a sliding window of 100 requests.

TrustZero was developed with a focus on transparency and ease of integration. Recognizing the importance of seamless interoperability between organizations, TrustZero introduces a streamlined way to enhance your security without adding unnecessary complexity.

To achieve this, we developed a proof-of-concept Android application designed to integrate effortlessly with any existing app. This tool allows you to automatically include your public key in the headers of outgoing traffic, ensuring secure identification and communication. TrustZero enables your application to manage received signatures effectively, including storing them securely and modifying them as needed to align with your security protocols and changes. The example of the integration app is presented in the annex A with the code available in the same repository [7]. Moreover, to test the actual integration of the keys in communication, the app was enhanced with an experiment where, after the containers(representing the servers) were started, a user could send requests and store the signatures received from different servers. This represents the final end-to-end open-source experiment towards passport-level trust including identity(key) generation and token exchange.

An initial performance analysis was measured with 9 Android phones starting to send requests to the servers over time. The results from Figure 8 show an overall higher average at 0.3 seconds but with a small interval distribution. This can be explained by the additional routing actions executed between the phone and the containerized servers running on the localhost machine. The UI of the experiment page alongside the resulting signatures received by 1 user are available in the annex at figure 11.

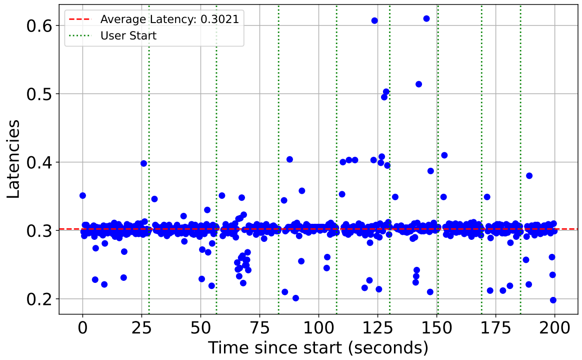


Fig. 8: Latency of 9 Android users

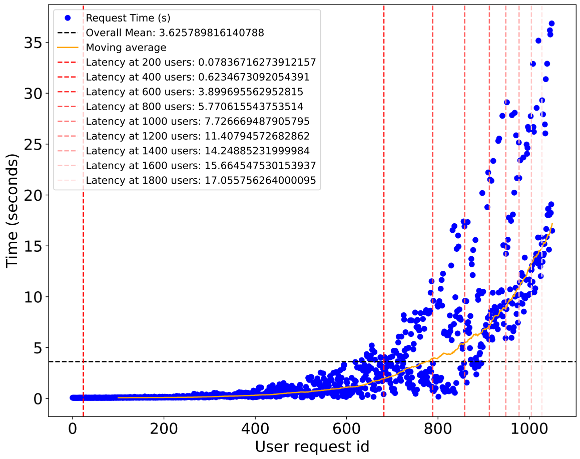


Fig. 9: Latency with up to 2000 users

VII. Conclusion

In an increasingly interconnected and volatile digital landscape, Zero Trust Architecture (ZTA) represents a vital paradigm shift in cybersecurity. This thesis introduced TrustZero, a scalable zero-trust security framework designed to address the limitations of traditional models. By leveraging a universal trust token and integrating robust cryptographic principles, TrustZero enhances trust portability, enables secure inter-organizational communication, and provides a resilient, mathematically grounded framework.

Our research underscores the transformative potential of combining zero-trust principles with lightweight cryptographic techniques to balance security and usability. Through rigorous testing, we demonstrated the feasibility of a distributed trust model that adapts to real-world complexities, such as supply chain interdependencies and evolving cyber threats. Experiments validated the efficiency of the trust scoring mechanism, showing minimal latency impacts and practical applicability even under simulated denial-of-service conditions.

TrustZero addresses the gap of implicit trust, as envisioned by ZTA, by focusing on mitigating the cost of breaches through continuous authentication and dynamic trust scoring. We implemented a proof-of-concept where we measured the latency of user requests and the resilience of the server in cases of HTTP floodings. Combining the zero-trust principle and cryptography in our design we create a strong identity and web-of-trust framework suitable to mitigate not only network attacks but also interferences in social activities.

Ultimately, TrustZero provides a foundational step toward an open, verifiable, and scalable zero-trust security ecosystem. By emphasizing transparency and reproducibility, this framework not only advances academic and industrial scenarios but also sets a precedent for future developments in cybersecurity.

References

[1] Commission, online platforms and civil society increase monitoring during Romanian elections — ec.europa.eu. https://ec.europa.eu/commission/presscorner/detail/en/ip_24_6243. [Accessed 10-12-2024].

[2] Commission opens formal proceedings against TikTok on election risks under the Digital Services Act — ec.europa.eu. https://ec.europa.eu/commission/presscorner/detail/en/ip_24_6487. [Accessed 17-12-2024].

[3] Hardware Security Modules — OpenZiti — openziti.io. https://openziti.io/docs/guides/hsm/#enabling-a-ziti-endpoint-using-an-hsm. [Accessed 04-10-2024].

[4] Responsibility in the supply chain — audi.com — audi.com. https://www.audi.com/en/sustainability/people-society/responsibility-in-the-supply-chain.html. [Accessed 01-12-2024].

[5] Responsible supply chain - Working with our suppliers to become a sustainable leader in our industry — asml.com. https://www.asml.com/en/company/sustainability/responsible-supply-chain. [Accessed 12-10-2024].

[6] Romania's presidential front-runner Georgescu benefited from Russia-style booster campaign, declassified docs say — politico.eu. https://www.politico.eu/article/romanias-presidential-frontrunner-benefited-from-russia-style-booster-campaign-declassified-docs-say/. [Accessed 16-12-2024].

[7] Trustzero. https://github.com/AdiDumi/TrustZero.

[8] What is OpenZiti? — OpenZiti — openziti.io. https://openziti.io/docs/learn/introduction/. [Accessed 04-10-2024].

[9] European Systemic Risk Board. *Advancing macroprudential tools for cyber resilience – Operational policy tools*. European Systemic Risk Board, 2024.

[10] AFM. Tiber-nl programme, 2016.

[11] Memen Akbar, Muhammad Arif Fadhly Ridha, et al. Sql injection and cross site scripting prevention using owasp modsecurity web application firewall. *JOIV: International Journal on Informatics Visualization*, 2(4):286–292, 2018.

[12] Faris Alsulami, Akshay R. Kulkarni, Noor Ahmad Hazari, and Mohammed Y. Niamat. Zebra: Zero trust architecture employing blockchain technology and ropuf for ami security. *IEEE Access*, 12:119868–119883, 2024.

[13] Sarah Birch and Fatma ElSafoury. Fraud, plot, or collective delusion? social media and perceptions of electoral misconduct in the 2014 scottish independence referendum. *Election Law Journal*, 16(4):470–484, 2017.

[14] Baozhan Chen, Siyuan Qiao, Jie Zhao, Dongqing Liu, Xiaobing Shi, Minzhao Lyu, Haotian Chen, Huimin Lu, and Yunkai Zhai. A security awareness and protection system for 5g smart healthcare based on zero-trust architecture. *IEEE internet of things journal*, 8(13):10248–10263, 2020.

[15] Joel Chigada and Rujeko Madzinga. Cyberattacks and threats during covid-19: A systematic literature review. *South African Journal of Information Management*, 23(1):1–11, 2021.

[16] Dan Dinculeană and Xiaochun Cheng. Vulnerabilities and limitations of mqtt protocol used between iot devices. *Applied Sciences*, 9(5):848, 2019.

[17] Victor Escobedo, Betsy Beyer, Max Saltonstall, and Filip Zyzniewski. Beyondcorp: The user experience. *Login*, 42(3):38–43, 2017.

[18] European Central Bank. Tiber-eu framework, 2018.

[19] Saman Feghhi and Douglas J Leith. A web traffic analysis attack using only timing information. *IEEE Transactions on Information Forensics and Security*, 11(8):1747–1759, 2016.

[20] Luca Ferretti, Federico Magnanini, Mauro Andreolini, and Michele Colajanni. Survivable zero trust for cloud computing environments. *Computers & Security*, 110:102419, 2021.

[21] Mailyn Fidler. Anarchy or regulation: Controlling the global trade in zero-day vulnerabilities. *PhD diss., Freeman Spogli Institute for International Studies, Stanford University*, 2014.

[22] Edielson P Frigieri, Daniel Mazzer, and LFCG Parreira. M2m protocols for constrained environments in the context of iot: A comparison of approaches. In *International Telecommunications Symposium*, page 5. sn, 2015.

[23] Virgil D Gligor. Zero trust in zero trust. Technical report, CMU CyLab Technical Report 22–002 December 17, 2022.

[24] Guilherme Gonçalves, Kyle O'Malley, Max Saltonstall, et al. Beyondcorp and the long tail of zero trust. 2023.

[25] Xian Guo, Hongbo Xian, Tao Feng, Yongbo Jiang, Di Zhang, and Junli Fang. An intelligent zero trust secure framework for software defined networking. *PeerJ Computer Science*, 9:e1674, 2023.

[26] Yuanhang He, Daochao Huang, Lei Chen, Yi Ni, and Xiangjie Ma. A survey on zero trust architecture: Challenges and future trends. *Wireless Communications and Mobile Computing*, 2022(1):6476274, 2022.

[27] Raphael Hiesgen, Marcin Nawrocki, Thomas C Schmidt, and Matthias Wählisch. The race to the vulnerable: Measuring the log4j shell incident. *arXiv preprint arXiv:2205.02544*, 2022.

[28] Trapti Jain and Nakul Jain. Framework for web application vulnerability discovery and mitigation by customizing rules through modsecurity. In *2019 6th International Conference on Signal Processing and Integrated Networks (SPIN)*, pages 643–648, 2019.

[29] László Viktor Jánoky, János Levendovszky, and Péter Ekler. An analysis on the revoking mechanisms for json web tokens. *International Journal of Distributed Sensor Networks*, 14(9):1550147718801535, 2018.

[30] Sean D Kaster and Prescott C Ensign. Privatized espionage: Nso group technologies and its pegasus spyware. *Thunderbird International Business Review*, 65(3):355–364, 2023.

[31] Julik S Keijer. Automated ddos mitigation based on known attacks using a web application firewall. B.S. thesis, University of Twente, 2019.

[32] John Kindervag et al. Build security into your network's dna: The zero trust network architecture. *Forrester Research Inc*, 27:1–16, 2010.

[33] V Lakhno, A Blozva, D Kasatkin, V Chubaievskyi, Y Shestak, D Tyshchenko, and R Brzhanov. Experimental studies of the features of using waf to protect internal services in the zero trust structure. *J Theor Appl Inf Technol*, 100(3):705–721, 2022.

[34] Butler Lampson. Privacy and security usable security: how to get it. *Communications of the ACM*, 52(11):25–27, 2009.

[35] Haiqin Liu, Yan Sun, Victor C Valgenti, and Min Sik Kim. Trustguard: A flow-level reputation-based ddos defense system. In *2011 IEEE Consumer Communications and Networking Conference (CCNC)*, pages 287–291. IEEE, 2011.

[36] Fares Meghdouri, Tanja Zseby, and Félix Iglesias. Analysis of lightweight feature vectors for attack detection in network traffic. *Applied Sciences*, 8(11):2196, 2018.

[37] Saima Mehraj and M. Tariq Banday. Establishing a zero trust strategy in cloud computing environment. In *2020 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–6, 2020.

[38] Iqra Naseer. The crowdstrike incident: Analysis and unveiling the intricacies of modern cybersecurity breaches. *World Journal of Advanced Engineering Technology and Sciences*, 10, 2024.

[39] TJ O'Connor. Chapter 4 - network traffic analysis with python. In TJ O'Connor, editor, *Violent Python*, pages 125–169. Syngress, 2013.

[40] GitHub Organization of the European Digital Identity project. European digital identity. https://github.com/eu-digital-identity-wallet/eudi-app-android-wallet-ui, 2024.

[41] Olugbenro Ogundipe and Tejiri Aweto. The shaky foundation of global technology: A case study of the 2024 crowdstrike outage. 2024.

[42] Jeff Peck, Betsy Beyer, Colin Beske, and Max Saltonstall. Migrating to beyondcorp: maintaining productivity while improving security. *Login*, 42(2):1–7, 2017.

[43] Pythagoras Petratos. Cybersecurity in europe: Cooperation and investment. *Cyber-Development, Cyber-Democracy and Cyber-Defense: Challenges, Opportunities and Implications for Theory, Policy and Practice*, pages 279–301, 2014.

[44] Ahmad Rammal, Kaja Gruntkowska, Nikita Fedin, Eduard Gorbunov, and Peter Richtárik. Communication compression for byzantine robust learning: New efficient algorithms and improved rates. In *International Conference on Artificial Intelligence and Statistics*, pages 1207–1215. PMLR, 2024.

[45] Ivan Ristic. *Modsecurity handbook*. Feisty Duck, 2010.

[46] Josh Rogin. Cyber officials: Chinese hackers attack 'anything and everything,'. *FCW. com, February*, 13:97658–1, 2007.

[47] Mayra Samaniego and Ralph Deters. Zero-trust hierarchical management in iot. In *2018 IEEE International Congress on Internet of Things (ICIOT)*, pages 88–95, 2018.

[48] Sebastian Santoni. The security council.

[49] Fred B. Schneider. Beyond hacking: an sos! In *2010 ACM/IEEE 32nd International Conference on Software Engineering*, volume 1, pages 2–2, 2010.

[50] Malcolm Shore, Sherali Zeadally, and Astha Keshariya. Zero trust: The what, how, why, and when. *Computer*, 54(11):26–35, 2021.

[51] Jatesh Jagraj Singh, Hamman Samuel, and Pavol Zavarsky. Impact of paranoia levels on the effectiveness of the modsecurity web application firewall. In *2018 1st International Conference on Data Intelligence and Security (ICDIS)*, pages 141–144. IEEE, 2018.

[52] Mudhakar Srivatsa, Li Xiong, and Ling Liu. Trustguard: countering vulnerabilities in reputation management for decentralized overlay networks. In *Proceedings of the 14th international conference on World Wide Web*, pages 422–431, 2005.

[53] V Stafford. Zero trust architecture. *NIST special publication*, 800:207, 2020.

[54] Naeem Firdous Syed, Syed W. Shah, Arash Shaghaghi, Adnan Anwar, Zubair Baig, and Robin Doss. Zero trust architecture (zta): A comprehensive survey. *IEEE Access*, 10:57143–57179, 2022.

[55] Songpon Teerakanok, Tetsutaro Uehara, and Atsuo Inomata. Migrating to zero trust architecture: Reviews and challenges. *Security and Communication Networks*, 2021(1):9947347, 2021.

[56] Theerasak Thapngam, Shui Yu, Wanlei Zhou, and Gleb Beliakov. Discriminating ddos attack traffic from flash crowd through packet arrival patterns. In *2011 IEEE conference on computer communications workshops (INFOCOM WKSHPS)*, pages 952–957. IEEE, 2011.

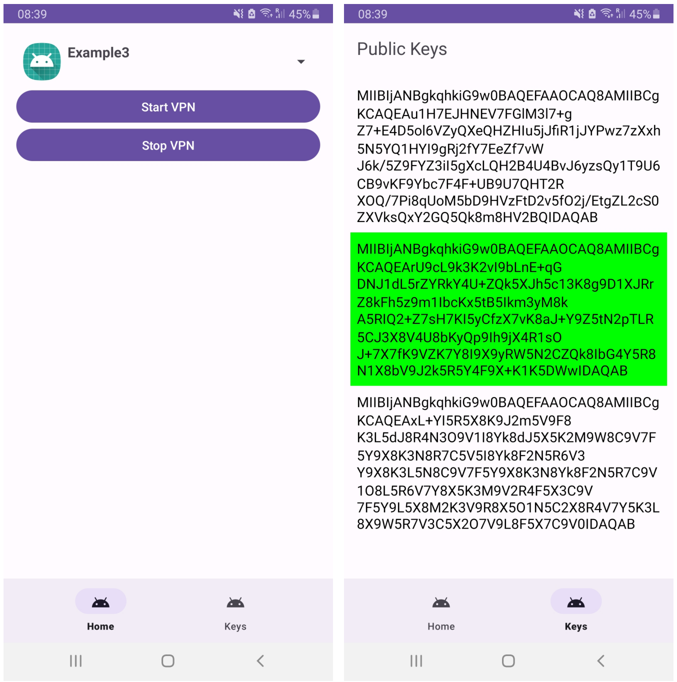
[57] Romans Vanickis, Paul Jacob, Sohelia Dehghanzadeh, and Brian Lee. Access control policy enforcement for zero-trust-networking. In *2018 29th Irish Signals and Systems Conference (ISSC)*, pages 1–6. IEEE, 2018.

[58] Rory Ward and Betsy Beyer. Beyondcorp: A new approach to enterprise security. *; login:: the magazine of USENIX & SAGE*, 39(6):6–11, 2014.

[59] Qigui Yao, Qi Wang, Xiaojian Zhang, and Jiaxuan Fei. Dynamic access control and authorization system based on zero-trust architecture. In *Proceedings of the 2020 1st international conference on control, robotics and intelligent system*, pages 123–127, 2020.

[60] Claudio Zanasi, Silvio Russo, and Michele Colajanni. Flexible zero trust architecture for the cybersecurity of industrial iot infrastructures. *Ad Hoc Networks*, 156:103414, 2024.

Appendix A
First Appendix



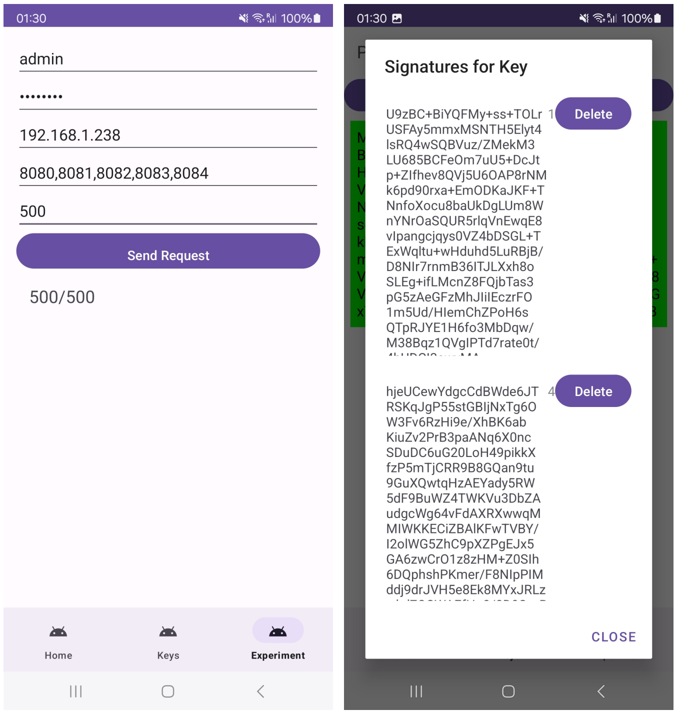
Public Keys

```
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCg
KCAQEAu1H7EJHNEV7FGlM3l7+g
Z7+E4D5ol6VZyQXeQHZHIu5jJfiR1jJYPwz7zXxh
5N5YQ1HYI9gRj2fY7EeZf7vW
J6k/5Z9FYZ3iI5gXcLQH2B4U4BvJ6yzsQy1T9U6
CB9vKF9Ybc7F4F+UB9U7QHT2R
XOQ/7Pi8qUoM5bD9HVzFtD2v5fO2j/EtgZL2cS0
ZXVksQxY2GQ5Qk8m8HV2BQIDAQAB
```

```
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCg
KCAQEArU9cL9k3K2vI9bLnE+qG
DNJ1dL5rZYRkY4U+ZQk5XJh5c13K8g9D1XJRr
Z8kFh5z9m1IbcKx5tB5Ikm3yM8k
A5RIQ2+Z7sH7KI5yCfzX7vK8aJ+Y9Z5tN2pTLR
5CJ3X8V4U8bKyQp9Ih9jX4R1sO
J+7X7fK9VZK7Y8I9X9yRW5N2CZQk8IbG4Y5R8
N1X8bV9J2k5R5Y4F9X+K1K5DWwIDAQAB
```

```
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCg
KCAQEAxL+YI5R5X8K9J2m5V9F8
K3L5dJ8R4N3O9V1I8Yk8dJ5X5K2M9W8C9V7F
5Y9X8K3N8R7C5V5I8Yk8F2N5R6V3
Y9X8K3L5N8C9V7F5Y9X8K3N8Yk8F2N5R7C9V
1O8L5R6V7Y8X5K3M9V2R4F5X3C9V
7F5Y9L5X8M2K3V9R8X5O1N5C2X8R4V7Y5K3L
8X9W5R7V3C5X2O7V9L8F5X7C9V0IDAQAB
```

Fig. 10: Android app for TrustZero integration



admin

••••••••

192.168.1.238

8080,8081,8082,8083,8084

500

500/500

Signatures for Key

```
U9zBC+BiYQFMy+ss+TOLr 1
USFAy5mmxMSNTH5Elyt4
lsRQ4wSQBVuz/ZMekM3
LU685BCFeOm7uU5+DcJt
p+ZIfhev8QVj5U6OAP8rNM
k6pd90rxa+EmODKaJKF+T
NnfoXocu8baUkDgLUm8W
nYNrOaSQUR5rlqVnEwqE8
vIpangcjqys0VZ4bDSGL+T
ExWqltu+wHduhd5LuRBjB/
D8NIr7rnmB36ITJLXxh8o
SLEg+ifLMcnZ8FQjbTas3
pG5zAeGFzMhJIiIEczrFO
1m5Ud/HIemChZPoH6s
QTpRJYE1H6fo3MbDqw/
M38Bqz1QVgIPTd7rate0t/
```

```
hjeUCewYdgcCdBWde6JT 4
RSKqJgP55stGBIjNxTg6O
W3Fv6RzHi9e/XhBK6ab
KiuZv2PrB3paANq6X0nc
SDuDC6uG20LoH49pikkX
fzP5mTjCRR9B8GQan9tu
9GuXQwtqHzAEYady5RW
5dF9BuWZ4TWKVu3DbZA
udgcWg64vFdAXRXwwqM
MIWKKECiZBAlKFwTVBY/
I2olWG5ZhC9pXZPgEJx5
GA6zwCrO1z8zHM+Z0SIh
6DQphshPKmer/F8NIpPIM
ddj9drJVH5e8Ek8MYxJRLz
```

Fig. 11: Android TrustZero experiment