A Two-Way Security-by-Design Paradigm for Fortification of Industrial IoT

Anand K Bapatla[1*], Saiuditi Rout[2], Saraju P Mohanty[3*], Elias Kougianos[4]

[1]Department of Computer Science and Cybersecurity, University of Central Missouri, Warrensburg, MO, USA.
[2]Department of Electrical Engineering, Indian Institute of Technology, Madras, TN, India.
[3]Department of Computer Science and Engineering, University of North Texas, Denton, TX, USA.
[4]Department of Electrical Engineering, University of North Texas, Denton, TX, USA.

*Corresponding author(s). E-mail(s): bapatla@ucmo.edu; saraju.mohanty@unt.edu;
Contributing authors: ee22b139@smail.iitm.ac.in; elias.kougianos@unt.edu;

Abstract

Industrial Internet of Things (IIoT) is a network of things that are connected to form a system to monitor, communicate, and analyze data with collective intelligence. Internet of Things (IoT) has been a driving technology for many of the sectors, Industrial automated systems is one such sector that has been deeply influenced. Ensuring the security of IIoT devices presents significant challenges, particularly in critical infrastructure. The current work adavances Security-by-Design (SbD) paradigm in IIoT. This paper proposes a novel blockchain-based verified patch delivery system that also integrates Verifiable Delay Functions (VDFs) to provide a two-layer security. Blockchain provides a tamper-proof ledger for verifying the integrity of patches as well as identification of IIoT devices preventing rogue devices in the network. VDFs introduced help create a synchronized patching across the critical infrastructure devices which avoids patch exploitation by adversaries by reducing the attack surface.

Keywords: Security by Design (SbD), Industrial Internet of Things, Distributed Ledger, Patch Delivery System, Synchronized Patching

1 Introduction

Internet of Things (IoT) is a rapidly increasing technology that has shown promising applications in various sectors. It makes use of smart things that are connected to each other that sense, communicate, analyze, and act to create collective intelligence in the network. IoT applied to the critical infrastructure of industrial applications is called the Industrial Internet of Things (IIoT). It is estimated around 15.9 billion IoT devices are currently connected and the number will reach more than 32.1 billion in the next decade [1]. Even though the usage of IoT in critical infrastructure comes with great advantages, it also introduces various security threats. As IoT things or devices are resource-constrained, applying complex cryptography techniques or running security software with a large footprint is not viable. Also, the increasing landscape of threats needs continuous firmware updates for the things. Hence, IoT things undergo continuous Patching. As the things in the IoT network are geographically dispersed forming a distributed network, connecting intermittently, constrained by limited resources, ensuring timely patch deployments becomes complex [2]. Traditional patch management systems are designed for centralized smaller-scale systems and are often incapable of handling large IoT environments.

In a centralized patch management system, a centralized server handles the storage, management, and distribution of patches to the connected IoT devices. This centralized approach has a few significant issues that include a Single Point of Failure (SPOF), an adversary gaining access to these servers, or a server becoming unavailable can disrupt the patch distribution process. Secondly, large IIoT deployments can overload the centralized servers leading to slower patch distribution, or even failed updates. Typical Patch delivery system is shown in Figure 1.

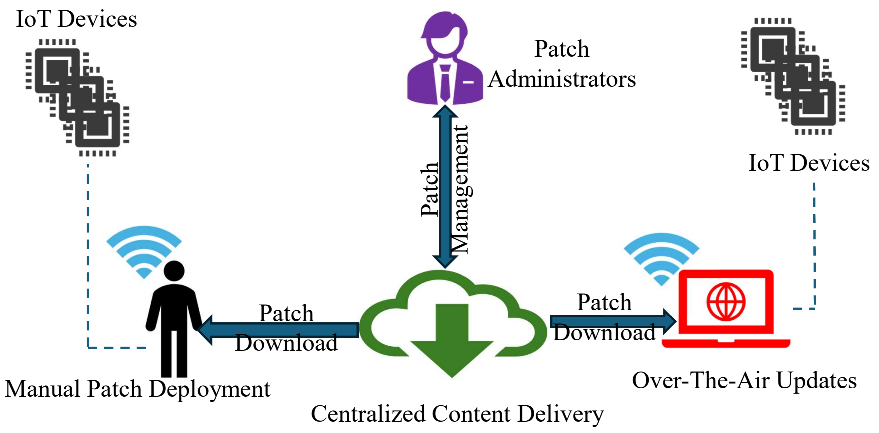


Fig. 1 Centralized Patch Delivery System

Over-the-Air (OTA) Updates are a way of delivering firmware and software updates wireless to the IoT devices [3]. In this method devices periodically check with a centralized cloud server for updates and patch automatically. Interception of patches while in the network, or man-in-the-middle attack can occur if the updates are not securely transmitted over the network. OTA updates also rely heavily on network connectivity, it will be a significant challenge to reach devices in remote areas with poor or intermittent connectivity leaving some devices vulnerable to security threats. IoT networks contain heterogeneous devices with varying resource capabilities that can create delays or failures in receiving critical updates. Manual patch deployment is also used in some of the Industrial IoT (IIoT) environments due to its critical nature, but this approach is laborious and time-consuming. Manual patching is also prone to human errors such as incorrect patch versions, missed devices, and incomplete patching.

Vendor-specific solutions, each IoT device manufacturer offers proprietary patch management systems that are specially designed for their devices. This approach comes with the advantage of devices being part of the manufacturer's ecosystem but poses significant downsides like vendor lock-in, lack of standardization, and security blind spots. Without industry-wide standards, these proprietary systems can create fragmented ecosystems that can make device interoperability harder.

Security-by-Design (SbD) is a proactive approach that integrates cybersecurity in both software and hardware design from the outset, making security a core component throughout the design process [4]. Proposed architecture follows SbD paradigm by leveraging blockchain and cryptography primitives Verifiable Delay Functions (VDFs). Blockchain is one of the latest technologies that is being explored in multiple sectors including IoT networks. The decentralized nature of blockchain can eliminate SPOF, ensure transparency, provide availability, and make patches tamper-resistant. Employing blockchain in a patch management system can provide IoT devices with a secure way to authenticate the patches before applying them. Also, going for a decentralized approach can remove network bottlenecks and can serve large IoT networks.

A second layer of security is employed in the proposed architecture that includes Verifiable Delay Functions (VDFs). Heterogeneous devices in the network with varied resources can introduce another significant challenge of reverse-engineering the patch [5]. Even automated tools are available to exploit these patches even before they reach the lower capable devices [6]. VDFs proposed in the current architecture help in synchronizing the patching mechanism of IIoT critical infrastructure by reducing the attack surface.

The rest of the paper is organized as follows: Section 2 discussed the current state-of-art systems proposed for patch delivery mechanisms. Section 3 discusses the novel contributions of proposed blockchain based approach with integrated VDFs. Section 4 provides preliminary knowledge and Section 5 describes the architecture of proposed system. Section 6 discusses proposed algorithms. Section 7 describes the implementation of proposed patch delivery system and Section 8 provides conclusions along with future research aspects.

2 Related Prior Works

Blockchain is one of the recent technologies which is growing rapidly and showing promising solutions in various sectors. IoT is one of such fields that is benefited and novel solutions leveraging blockchain are being explored. Even blockchains targeting resource-constrained systems like IoT are being designed [7]. This section discusses state-of-art proposed patch management systems.

CrowdPatching [8] proposed a decentralized protocol for delivering software updates to IoT devices using Blockchain and zero-knowledge proofs. The proposed architecture also includes a compensation mechanism for the distributors to enhance trust by rewarding honest behavior. The proposed model provides high scalability and efficient redistribution; however, this approach doesn't consider the security issues that can be introduced by the distributors, and also synchronization of patch application is not discussed.

Proposed architecture in [9] utilized Ciphertext-Policy Attribute-Based Encryption (CP-ABE) combined with blockchain to provide software updates to only authorized devices. However, the complexity of decryption for CP-ABE is high and requires large computations. Authors tried addressing this by offloading these computations to blockchain which are expensive and require large amounts of resources.

Another blockchain-based approach utilizing the Ethereum platform is proposed in [10]. The proposed peer-to-peer updates protocol leverages the blockchain ledger to store software update metadata and uses it to enhance the integrity of patches. This ensures the patch is not tampered with when in the network. However, the current solution along with considering metadata for patch integrity also provides a way to eliminate rogue IoT devices from the network.

Patch Transporter [11] discusses the challenges of delivering software updates in Wireless Sensor Networks (WSN). A decentralized approach based on incentives for faithfully transporting the patch is proposed in this paper. Patches are encrypted and provided to the transporters who are responsible for delivering it to the end devices. Keys are shared using blockchain using which the end IoT devices will decrypt the patch.

Device identification using hardware primitives like PUF is proposed in [12]. Another SbD approach utilizing hardware security primitives and blockchain for IoT device identity is explored in [13]. This proposed approach is an efficient way of creating an identity chain for the IoT devices ensuring rogue devices can be identified easily in the network. Another approach of creating device identities using cryptography keys is proposed in [14].

3 Novel Contributions of Proposed Model

Our novel blockchain-based patch delivery system with integrated Verifiable Delay Functions (VDFs) and device identification mitigates some of the previously discussed problems from centralized patch delivery systems. Below are the novel contributions:

- By utilizing a distributed architecture, the proposed system avoids single point of failure and centralized control issues.
- Distributed nature can also avoid overloading by enabling parallel operations that efficiently manage large IoT networks.
- The blockchain-based approach will provide a secure system by avoiding centralized breaches, making it resistant to tampering and cyber-attacks.

- Proposed architecture utilizes VDFs to enforce a calculated delay that prevents the devices from applying the patches prematurely and ensures synchronized patch deployment.
- By utilizing digital twinning, the identity of each IoT device is managed to mitigate the rogue devices in the network.
- Blockchain decentralized approach also provides a common platform for different vendors to work collectively and reduce the interoperability issues.

4 Preliminaries

Verifiable Delay Functions (VDFs) cryptography primitives that are designed for enforcing a predefined amount of time delay by requiring a certain number of sequential computations. These sequential computations cannot be parallelized, heterogeneous devices with varied computational capabilities roughly takes the same amount of time to do these computations. Even though, computation takes predefined delay, verifying the result is easy and efficient for VDFs.

VDF involves two main steps. Firstly, evaluation step in which the devices will compute the VDF that takes a pre-determined and verifiable amount of time. Secondly, verification stage in which the output of computed VDF can be quickly and easily verified. The main goal is to enforce a computation of at least T steps, where T is desired delay. This sequence computation must be sequential to ensure no amount of computational resources and large devices can compute the result faster than T.

One of the most common method to implement VDF is through repeated squaring [15] in a group where the discrete logarithm is hard to compute. Let us consider an example where VDFs are implemented using modular exponentiation. let x is the base of VDF, N is a large prime modulus which is chosen such that $N = pq$ where p and q are large primes, T is the number of sequential steps required. The VDF computation is defined as below:

$$y = x^{2^T} \mod N. \tag{1}$$

In the above expression:

- x is the base (starting value).
- N is a large prime modulus.
- T is the number of sequential squaring steps required.

The computation phase in which repeated squaring of the base x for T steps. As the result of next step depends on output of the previous step, computation is sequential and cannot be parallelized ensuring that it takes at least T number of steps to compute the final result y.

Second phase in the process is to verify the computation. The result y can be easily verified by using the following equation:

$$y^{2^T} \mod N = x \mod N. \tag{2}$$

If this equation holds, then y is a valid result of the VDF. The verification process uses efficient exponentiation algorithms and has a time complexity of $O(\log T \log N)$, making it significantly faster than the full VDF computation.

The security of the VDF relies on the assumption that the discrete logarithm problem is hard to solve in the group modulo N. The VDF's resistance to parallelization ensures that even with substantial computational resources, the function cannot be computed in fewer than T sequential steps. Specifically, the hardness assumptions are based on:

- The modular exponentiation problem, which is assumed to be hard to reverse.
- The sequentiality of the squaring operation, which prevents parallel computation.

These security assumptions ensure that the VDF introduces a verifiable time delay, regardless of the computational power available to the adversary. In the context of IoT patch management, VDFs ensure that patches are applied in a synchronized and controlled manner, reducing the risk of exploitation during the patching process.

5 Architectural Overview of the Proposed System

The proposed Patch management system architecture is shown in Figure 2. Core components of the proposed system consist of Blockchain, Smart Contracts, Decentralized File System, IoT devices, VDF Module, Device identity management, and Patch providers or administrators. Each of these components is discussed below:

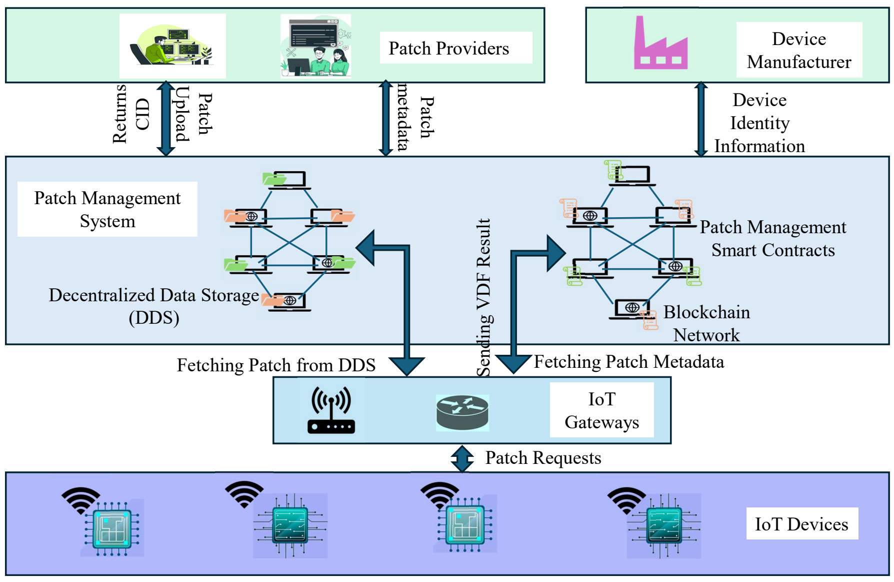


Fig. 2 High Level Architecture Overview

Patch providers or administrators are the entities that are responsible for developing new patches and ensuring they address the device's vulnerabilities. Once the patch is ready and tested, patch providers will upload the patch to Decentralized storage and receive the CID. Received CID along with patch metadata and VDF parameters will be uploaded to the blockchain. Information about the patch such as the Content ID (CID) of the patch from the Decentralized File System, version number, and release date will be uploaded by the patch provider to the ledger. Along with patch information, this ledger also ensures identity management by logging the status of the IoT device, information about the version of the patch, and whether the patch is applied to it or not. Results from the VDF will also be stored in the ledger that will be used during the verification phase. The characteristics of blockchain will ensure once the patch is submitted by the provider, it is accessible to all the devices in the network in a secure and immutable way.

Smart contracts are pieces of code that execute automatically when certain conditions are met. These reside on the blockchain and will be executed by all the nodes automatically. Patch Registration functions in Smart Contract are responsible for recording the patch metadata along with the other information regarding the patch. VDF functions are designed to verify each device's VDF computations. Device Identity function registers and authorizes only genuine IoT devices restricting access to rogue IoT devices.

IoT nodes act as the endpoints of the system, and their responsibilities include querying the blockchain to retrieve the latest patch metadata including VDF parameters. The device then starts the VDF computation, and the results will be submitted to the blockchain for verification, once the verification and device authentication are complete, the device will receive the CID using which it will download the actual patch from the decentralized. These devices will interact with the blockchain layer via lightweight blockchain clients as these are resource-constrained.

The VDF module is a crucial component in the proposed architecture which will enforce a verifiable time delay. This VDF computation ensures all devices in the network wait for a predetermined time before they can apply the patch as prematurely applying patches can increase the risk of attackers. Repeated squaring is used in the proposed architecture for VDF computations.

Another key feature of the system is managing device identities and states using smart contracts. Maintaining a ledger of the authorized devices will avoid unauthorized access of patches by the rogue devices. Device manufacturer, once every IoT device is manufactured, device-associated details like Media Access Control (MAC) ID and other device information are registered to the blockchain. This information will be used for authorization of genuine devices and blockchain along with maintaining the identity, it will also store the current state of the device. This information will help during audits to check if there are any unpatched devices in the network and take prompt action.

6 Proposed Algorithms

Uploading the patch by patch administrator is clearly shown in Algorithm 1. Once the patch is created that addresses the vulnerabilities of the IoT devices, the patch administrator will compute the hash of the patch and then sends the actual patch file to decentralized storage system. CID from Decentralized storage system along with hash of the patch, patch metadata will be uploaded to the blockchain ledger by calling smart contract functions.

Algorithm 1 Patch Upload Process

```
1:  Input: Patch data $P$, Metadata $M$ including version, release date, etc.
2:  Output: Patch registered on blockchain with cryptographic hash $H$
3:  Admin generates the patch $P$ and associated metadata $M$.
4:  Admin computes the cryptographic hash $H$ of the patch:
5:      $H = Hash(P)$
6:  Store the actual patch file $P$ in an external decentralized storage system.
7:  Create a smart contract transaction with:
8:      a. Cryptographic hash $H$
9:      b. Patch version and metadata $M$
10:     c. Link to patch $CID$ from external storage
11:     d. Verifiable Delay Function (VDF) parameters $T$
12: Register the patch metadata, hash $H$, and storage reference on the blockchain.
13: Blockchain stores the hash $H$ and metadata $M$ in an immutable ledger.
14: Confirm registration and patch availability.
```

During the patch download process, IoT devices follow the following steps described in Algorithm 2. The first step is the IoT device querying the patch data from the blockchain using a lightweight client. Once, the patch metadata is received the device will start the VDF computation and the computed result will be sent back to the smart contract for verification. On successful verification, the patch will be applied by the device.

Device registration and authorization steps are shown in the Algorithm 3. When the manufacturer manufactures a device, details about the device like MAC address, Device Type, and other information will be registered by the manufacturer by invoking the smart contract. These details will be stored in an immutable ledger and later used for authorizing the device.

7 Implementation and Validation

To validate the proposed patch delivery system architecture, it is designed using the Ethereum platform. Ethereum is a blockchain platform designed for Decentralized Application (DApp) development which will abstract the complexity of using blockchain. Ethereum supports smart contracts which are used in our design to perform essential functions of device registration and verification, patch upload and delivery, and VDF computations and verification. These smart contracts are written in solidity language.

Truffle suite is used for developing the required DApp and Ganache local blockchain is used for testing the application. Ganache is a local blockchain that mimics the operation of the main chain but runs locally. Ganache provides free accounts with test

Algorithm 2 Patch Download Process

```
1: **Input:** IoT Device ID, Patch Metadata $M$, Patch Hash $H$
2: **Output:** Patch downloaded, VDF computed, and applied
3: IoT device queries blockchain for available patches.
4: Device retrieves patch metadata $M$ and hash $H$ from the blockchain.
5: Device retrieves the patch file $P$ from the external storage using the metadata link.
6: Verify patch integrity:
7:     $H' = Hash(P)$
8: **if** $H' \neq H$ **then**
9:     Reject the patch and abort the process.
10: **end if**
11: Compute the Verifiable Delay Function (VDF):
12:     $y = x^{2^T} \mod N$
13: Submit the VDF result to the blockchain for verification.
14: **if** VDF result is verified **then**
15:     Apply the patch.
16:     Update the device state on the blockchain as patched.
17: **else**
18:     Abort and log the failure.
19: **end if**
```

Ether which is the native currency of Ethereum for testing purposes. This facilitates an inexpensive way of testing our DApp's during development instead of spending real currency on the MAINNET. The same is used in our implementation for functional testing. Client programs using JavaScript are created that act as IoT Nodes for requesting and processing the patches.

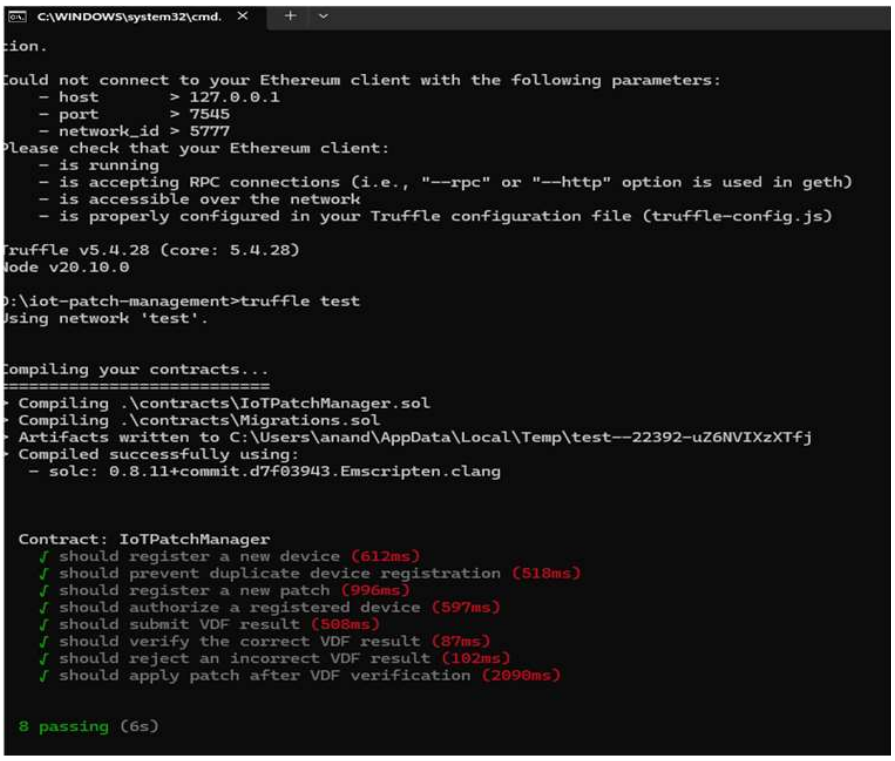
A functional analysis is performed using the Chai assertion library for Node.js to verify the expected outcomes of the tests. Different test cases are designed to capture the functions of the smart contract. Results from the analysis can be seen in Figure 3. A comparative analysis between the state-of-art and proposed architecture is shown in Table 1.

8 Conclusions and Future Research

In this work, we have proposed a novel Blockchain-based IoT Patch delivery mechanism integrated with VDF to ensure timely delivery and deployment. The proposed architecture is robust and reliable to address the challenges of securing critical IoT infrastructure. By leveraging a tamper-proof ledger, this system ensures verifiable patch management. The inclusion of VDFs adds an additional layer of security by enforcing a verifiable time delay in the patching, preventing premature and unauthorized updates. By employing a device identity mechanism using smart contracts, we were able to mitigate the unauthorized entities gaining access to patches.

Algorithm 3 Device Identity Management Process

```
 1: Input: Device details $D$ (MAC address, device type)
 2: Output: Device registered on blockchain and authorized for patch downloads
 3: Device manufacturer generates device details $D$ (MAC address).
 4: Manufacturer registers the device $D$ on the blockchain by invoking a smart
    contract.
 5: Smart contract stores:
 6:     a. Device MAC address
 7:     b. Device state (patched/unpatched)
 8:     c. Manufacturer details
 9: When an IoT device requests a patch, it submits its MAC address.
10: Blockchain smart contract checks the MAC against registered devices.
11: if MAC is valid then
12:     Authorize the device to download the patch.
13: else
14:     Reject the request and log the attempt.
15: end if
16: Once the patch is applied, update the device state as patched.
17: Administrators can audit device states to detect unpatched devices.
```



```
ion.

ould not connect to your Ethereum client with the following parameters:
   - host       > 127.0.0.1
   - port       > 7545
   - network_id > 5777
lease check that your Ethereum client:
   - is running
   - is accepting RPC connections (i.e., "--rpc" or "--http" option is used in geth)
   - is accessible over the network
   - is properly configured in your Truffle configuration file (truffle-config.js)

ruffle v5.4.28 (core: 5.4.28)
ode v20.10.0

:\iot-patch-management>truffle test
sing network 'test'.


ompiling your contracts...
==========================
 Compiling .\contracts\IoTPatchManager.sol
 Compiling .\contracts\Migrations.sol
 Artifacts written to C:\Users\anand\AppData\Local\Temp\test--22392-uZ6NVIXzXTfj
 Compiled successfully using:
  - solc: 0.8.11+commit.d7f03943.Emscripten.clang



 Contract: IoTPatchManager
   ✓ should register a new device (612ms)
   ✓ should prevent duplicate device registration (518ms)
   ✓ should register a new patch (996ms)
   ✓ should authorize a registered device (597ms)
   ✓ should submit VDF result (508ms)
   ✓ should verify the correct VDF result (87ms)
   ✓ should reject an incorrect VDF result (102ms)
   ✓ should apply patch after VDF verification (2090ms)


 8 passing (6s)
```

Fig. 3 Functional Test Results from Deployed Proposed System on Ganache

	Blockchain Platform	**Patch Storage**	**Patch Integrity Verification**	**Device Identity**	**Device State**	**Synchronized Patch Application**
Puggioni, et al. [8]	Ethereum	Decentralized Storage	✓	Public & Private Keys	✗	✗
Solomon, et al. [9]	Ethereum	Centralized Cloud Server	✓	CP-ABE encryption	✗	✗
Witanto, et al. [10]	Ethereum	Centralized Manufacturer Server	✓	✗	✗	✗
Lee, et al. [11]	Bitcoin	Centralized Provider Server	✓	Targeted Set of Devices	✗	✗
Proposed	Ethereum	Decentralized Storage	✓	Manufacturer Device Metadata	✓	✓

Table 1 Comparative Analysis of Proposed Architecture with State-of-Art

In future work, we will perform performance optimization of VDF computations and explore lightweight VDF implementations for resource-constrained devices. Layer-1 blockchains come with limitations on scalability and throughput, so we will explore Layer-2 solutions to accommodate higher traffic from large IoT networks.

References

[1] Vailshery, L.S.: Number of IoT Connections Worldwide 2022-2033, with Forecasts to 2030. Last Accessed: 2024-09-01. https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/

[2] Maroof, U., Shaghaghi, A., Michelin, R., Jha, S.: iRECOVer: Patch your IoT on-the-fly. Future Generation Computer Systems **132**, 178–193 (2022) https://doi.org/10.1016/j.future.2022.02.014

[3] Villegas, M.M., Orellana, C., Astudillo, H.: A study of over-the-air (OTA) update systems for CPS and IoT operating systems. In: Proc: 13th European Conference on Software Architecture. ECSA, vol. 2, pp. 269–272. ACM, Paris, France (2019). https://doi.org/10.1145/3344948.3344972

[4] Pescador, F., Mohanty, S.P.: Guest Editorial Security-by-Design for Electronic Systems. IEEE Transactions on Consumer Electronics **68**(1), 2–4 (2022) https://doi.org/10.1109/tce.2022.3147005

[5] Barnett, R.: The Race to Patch: Attackers Leverage Sample Exploit Code in WordPress Plug-in. Last Accessed: 2024-09-01. https://www.akamai.com/blog/security-research/attackers-leverage-sample-exploit-wordpress-plugin

[6] Letian, S., Jing, C., Jianming, F., Guojun, P.: PVDF: an automatic patch-based vulnerability description and fuzzing method. In: Proc: Communications Security Conference, pp. 1–8. Institution of Engineering and Technology, Beijing (2014). https://doi.org/10.1049/cp.2014.0733

[7] Bapatla, A.K., Puthal, D., Mohanty, S.P., Yanambaka, V.P., Kougianos, E.: Easy-Chain: an IoT-friendly blockchain for robust and energy-efficient authentication. Frontiers in Blockchain **6** (2023) https://doi.org/10.3389/fbloc.2023.1194883

[8] Puggioni, E., Shaghaghi, A., Doss, R., Kanhere, S.S.: Towards Decentralized IoT Updates Delivery Leveraging Blockchain and Zero-Knowledge Proofs. In: Proc: 19th International Symposium on Network Computing and Applications (NCA), pp. 1–10. IEEE, Cambridge, MA, USA (2020). https://doi.org/10.1109/nca51143.2020.9306689

[9] Solomon, G., Zhang, P., Brooks, R., Liu, Y.: A Secure and Cost-Efficient Blockchain Facilitated IoT Software Update Framework. IEEE Access **11**, 44879–44894 (2023) https://doi.org/10.1109/access.2023.3272899

[10] Witanto, E.N., Oktian, Y.E., Lee, S.-G., Lee, J.-H.: A Blockchain-Based OCF Firmware Update for IoT Devices. Applied Sciences **10**(19), 6744 (2020) https://doi.org/10.3390/app10196744

[11] Lee, J.: Patch Transporter: Incentivized, Decentralized Software Patch System for WSN and IoT Environments. Sensors **18**(2), 574 (2018) https://doi.org/10.3390/s18020574

[12] Li, Z., Chu, Y., Liu, X., Zhang, Y., Feng, J., Xiang, X.: Physical Unclonable Function Based Identity Management for IoT with Blockchain. Procedia Computer Science **198**, 454–459 (2022) https://doi.org/10.1016/j.procs.2021.12.269

[13] Bathalapalli, V.K.V.V., Mohanty, S.P., Kougianos, E., Iyer, V., Rout, B.: PUFchain 3.0: Hardware-Assisted Distributed Ledger for Robust Authentication in Healthcare Cyber–Physical Systems. Sensors **24**(3), 938 (2024) https://doi.org/10.3390/s24030938

[14] Ghosh, U., Das, D., Banerjee, S., Mohanty, S.: Blockchain-Based Device Identity Management and Authentication in Cyber-Physical Systems. In: Proc: Consumer Communications and Networking Conference, pp. 1–6. IEEE, Las Vegas, NV, USA (2024). https://doi.org/10.1109/ccnc51664.2024.10454888

[15] Wesolowski, B.: Efficient Verifiable Delay Functions. In: Ishai, Y., Rijmen, V. (eds) Advances in Cryptology – EUROCRYPT 2019, vol. 11478, pp. 379–407. Springer, Darmastadt, Germany (2019). https://doi.org/10.1007/978-3-030-17659-4_13