Collectively Enhancing IoT Security: A Privacy-Aware Crowd-Sourcing Approach

Markus Dahlmanns[1], Roman Matzutt[1,2], Chris Dax[1], and Klaus Wehrle[1]

[1] Communication and Distributed Systems, RWTH Aachen University, Germany
[2] Data Protection and Sovereignty, Fraunhofer FIT, Aachen, Germany
{dahlmanns, matzutt, dax, wehrle}@comsys.rwth-aachen.de

Abstract. Security configurations remain challenging for trained administrators. Nowadays, due to the advent of the Internet of Things (IoT), untrained users operate numerous and heterogeneous Internet-facing services in manifold use case-specific scenarios. In this work, we close the growing gap between the complexity of IoT security configuration and the expertise of the affected users. To this end, we propose ColPSA, a platform for **col**lective and **p**rivacy-aware **s**ecurity **a**dvice that allows users to optimize their configuration by exchanging information about what security can be realized given their IoT deployment and scenario.

1 Introduction

The success of the Internet of Things (IoT) and its flavors, e.g., Smart Home and the Industrial IoT (IIoT), lead to a surge of Internet-connected devices and services handling sensitive data [14, 17, 27], which must be secured against unwanted access, eavesdropping, and overtakes. Yet, keeping networked devices secure remains a difficult and error-prone task even for trained system administrators [15]. In the IoT, the situation gets even worse as the intended users have less knowledge about IT security. Moreover, IoT deployments are increasingly complex and consist of numerous and heterogeneous components [17, 27]. Simultaneously, these components do not implement all security features [7], e.g., to reduce costs. Finally, IoT deployments are constrained by individual use cases, e.g., requiring or denying remote access. Overall, best practices for secure configurations cannot be transferred easily from one IoT deployment to another. As a result, IoT deployments have often proved to be notoriously insecure in the past [3, 4, 18, 30] or relying on compromised placeholder certificates [4, 5].

The goal of this work is to close this critical gap between the knowledge and effort required to securely configure IoT deployments and the end-users' capabilities to *realize* such a secure configuration based on their individual needs. To achieve this goal, we enable end-users to *exchange knowledge about realizable security configurations* in an automated fashion. In this paper, we present ColPSA, our central platform for **col**lective and **p**rivacy-aware **s**ecurity **a**dvice. ColPSA learns the most secure configuration for any specific IoT component in a particular scenario by regularly crowd-sourcing reports from participating end-users. Based on this knowledge, ColPSA then notifies users with improvable security configurations on how to optimize their IoT deployments.

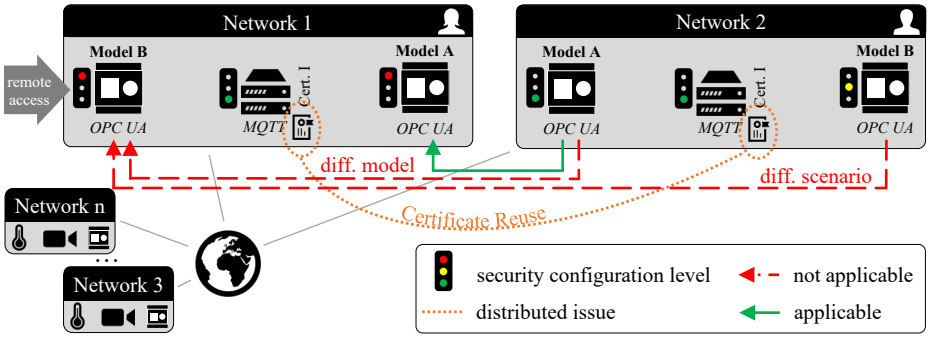


Fig. 1. Users operate their IoT deployments. Security configurations cannot be transferred between devices of different models or used in different scenarios. Additionally, deployments can suffer from distributed issues.

2 Security Configuration Goals and Pitfalls

Introducing *IoT components*, i.e., a potentially diverse subset of IoT devices and services, in their network requires users to actively maintain end-to-end security and access control. Here, users have to familiarize themselves with a multitude of available protocols. We give an overview of selected IoT protocols, i.e., TLS for secure communication, which can in turn be used by, e.g., MQTT, and OPC UA.

TLS and MQTT: TLS establishes confidential, integrity-protected, and authenticated connections [25]. Thus, many IoT protocols, e.g., MQTT, nowadays rely on TLS (or derivatives such as DTLS for UDP connections) as well [4]. However, *users are responsible for using up-to-date TLS configurations*, i.e., protocol version, cipher suite, and cryptographic primitives of their certificates [10, 29]. Additionally, to prevent unwanted access, *users need to manually configure secure credentials* and *manually enable the MQTT broker to enforce access control.*

OPC UA: OPC UA aims to homogenize IIoT deployments, e.g., by enabling cross-vendor communication [3, 8]. Additionally, it is the first IIoT protocol with built-in and attested security [8]. However, its security bases on custom protocol features instead of TLS [20]. Here, *users must actively enable confidentiality and authentication*, i.e., enable the correct security mode, and *select secure cryptographic primitives*, i.e., select a secure security policy.

3 Users Need IoT-specific Security Advice

The idea of supporting users with security advice has already been pursued for traditional IT services. However, the shift to the IoT comes with new challenges.

Figure 1 illustrates multiple users operating their IoT deployments that consist of multiple components each. These deployments incorporate both *heterogeneous IoT devices* with varying capabilities that satisfy *different, incompatible scenarios*, and have *varying degrees of realized security configurations.*

Heterogeneous IoT Devices: The heterogeneity of IoT devices highly complicates their (secure) configuration [28] as they typically do not implement all security features and cryptographic primitives defined for a protocol [7, 11],

e.g., to reduce development costs or due to hardware limitations. Instead, users have to keep the exact models of their IoT devices in mind when aiming for the most *realizable* security configuration.

Different Scenarios: Besides the device models, also the user's intended deployment scenario influences the most secure and *suitable* configuration. For example, deployments with components that are accessible from the outside may require other security measures than deployments that are only reachable from within the network. While externally reachable components handling sensitive data definitely should implement access control, other components are intentionally openly available, e.g., to share data.

Lax Current Level of Security: Previous research found various insecurely configured Internet-reachable IoT deployments, which put the confidentiality of sensitive data and the user's control over their components at risk [3–5, 18, 30]. Responsible disclosures related to these measurements regularly surprise affected users, which indicates they tend to be unaware of insecure configurations.

3.1 Requirements for IoT-specific Security Advice

Users need a reliable external source of expert knowledge that can give security advice that is tailored to the users' IoT components and needs. We identify the following six requirements for such recommendation systems.

R1: Technical Applicability: Any security advice given to the user must be realizable within their individual IoT deployment, i.e., take the user's specific IoT components and their capabilities into account. Giving advice solely based on protocol-specific best practices can aggravate to the user's burden when they cannot implement those recommendations or start to ignore inapplicable advice.

R2: Scenario-specific Adequacy: In addition to the technical applicability, the security advice must also fit the user's intended use case, i.e., the advice must be adequate for the given scenario. The scenario fundamentally influences the best-suited security configuration of an IoT deployment, e.g., a public weather camera does not require access control whereas a private surveillance camera should not be accessed by unauthorized parties. Thus, the security advice should not intimidate the user by making overly restrictive constraints and ignoring the user's requirements.

R3: Security: Any given advice should recommend the most secure configuration that is suitable after factoring in the individual device capabilities (R1) and the scenario (R2). As such, the advice must holistically consider the IoT deployment and detect configuration issues across multiple components. Notably, reusing secret values, e.g., copying example private keys from online tutorials or container images, can significantly impede the deployment's overall security [5].

R4: Generalizability: The IoT comprises vast and ever-growing amounts of different protocols and devices. Moreover, there is no clear boundary between different use cases and the applied technologies; for instance, PLCs are predominantly used in the IIoT, but can also occur in the context of smart homes. Thus, any system for assessing an IoT deployment's security configuration and

		Appl. (R1)	Adeq. (R2)	Sec. (R3)	Gen. (R4)	D. A. (R5)	Scal. (R6)
Security Guidelines (e.g. [9, 10, 22])		○	◔	◑	○	●	○
Local Assessment	Active (e.g. [1, 2, 31, 32])	◔	◔	◑	●	●	●
	Passive (e.g. [13, 26])	◔	◔	◑	●	●	◑
Remote Assessment (e.g. [23])		◔	○	●	◑	○	●

Table 1. Overview of related work. Currently, no approach meets all requirements, especially none achieves the desirable combination of applicability and adequacy.

advising for improvements should be generalizable to support established and upcoming IoT protocols as well as different network configurations.

R5: Data Avoidance: Potential configuration flaws in a user's IoT deployment bear high security risks and require manual intervention by the user. Hence, any externally available information on the user's security configuration is potentially valuable for adversaries. Users must be able to trust that any security assessment of their IoT deployments cannot benefit targeted attacks against them. As a result, any system for giving security advice must ensure the unlinkability between users and their configuration flaws.

R6: Scalability: Finally, due to the ever-growing size of the IoT and users' deployments, any advice-giving system must remain scalable. Namely, such systems must be able to handle numerous large and complex IoT deployments.

4 Related Work

To increase the level of security, users require advice matching to their components and needs, i.e., fulfills the requirements identified in Section 3.1. However, current approaches do not help completely. Table 1 summarizes our results.

Security Guidelines: Current best practices for security configurations on the protocol level are maintained in extensive *security guidelines*, e.g., for TLS [10] or OPC UA [22]. While this dissemination method for security advice can be scenario-agnostic (R2), typically up-to-date (R3), and requires no data to be disclosed (R5), it comes with significant downsides as well. Most notably, the guidelines are centered around single protocols and are not device-agnostic, which severely limits their generalizability (R4) and their applicability (R1). Likewise, users have to react manually to new recommendations, which limits the scalability (R6). Finally, security guidelines can only cover individual building blocks and cannot identify misconfigurations such as certificate reuse (R3).

Security Assessment Tools: When relying on *locally* deployed security assessment, users can choose between *active* and *passive* monitoring of their network. Here, active approaches, e.g., OpenVAS [24], actively scan deployments for issues and passive approaches, e.g., Snort [26], intercept ongoing communication to extract security-related information. Both approaches are signature-based and therefore avoid disclosing data to external parties (R5) but can be updated as the need arises (R4). However, while active tools can scale also to large network sizes [6], passive tools have to inspect every packet, which increases hardware

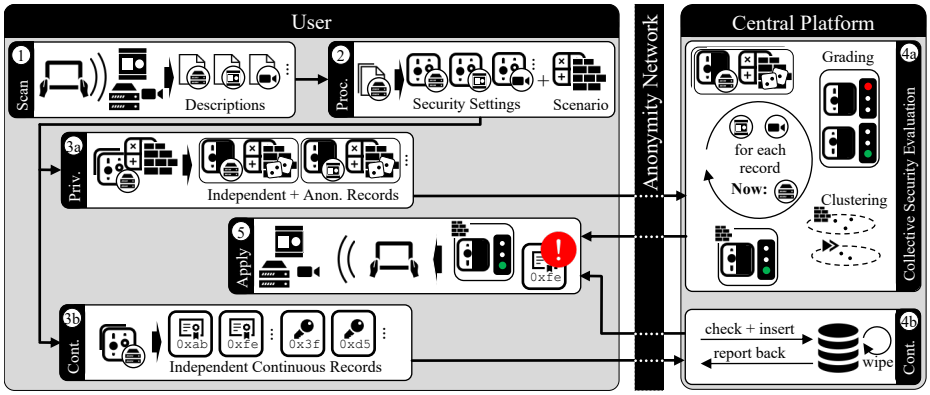


Fig. 2. ColPSA gives security advice for IoT deployments by accumulating knowledge about realizable and scenario-specific configurations from anonymous local scans.

requirements and limits scalability [21] (R6). Moreover, these approaches do not have a global view on security configurations (R3) and current approaches neither consider technical applicability (R1) nor scenario-specific adequacy (R2).

Instead of conducting their security assessments locally, users can instruct *remote* services to perform active scans of their deployments [23]. The service operators can scale up their infrastructure according to the growing needs of the IoT (R6) and combine the results of multiple deployments, i.e., they can detect security issues on a large scale (R3). However, precisely this external data collection poses significant risks regarding the disclosure of users' vulnerabilities (R5). Moreover, remote scans can only cover (directly) Internet-reachable IoT components and do not generalize to arbitrary deployments (R4). Finally, no remote scanning service can holistically consider the applicability (R1) or adequacy (R2) of its security advice for a particular IoT deployment.

5 Collectively Enhancing the IoT Security

Our discussion in Section 4 revealed that current security advice processes do not meet all requirements for improving the situation in today's IoT deployments. Accordingly, we now present ColPSA, our platform for **col**lective and **p**rivacy-aware **s**ecurity **a**dvice for the IoT that closes this gap. The core idea of ColPSA is to centrally derive applicable and adequate security advice for IoT deployments from its users' *combined* and *anonymized* local active network scans. This way, ColPSA can incrementally accumulate knowledge about the most secure and realizable security configuration for specific IoT devices and how different scenarios influence this assessment. Additionally, by only processing anonymized reports, ColPSA does not expose any vulnerabilities of their IoT deployments.

Figure 2 gives an overview of how users interact with ColPSA. Each user starts by ① locally scanning their network to generate descriptions of their IoT deployment, including the present IoT components, ports used, and Internet reachability. Next, the scanner ② processes the scan results, i.e., extract

their security configuration and scenario indicators, before ③ preparing them for upload to the ColPSA platform. Depending on the data types of individual records, the scanner either ③ᵃ fully anonymizes the record or ③ᵇ derives a fingerprint that allows for matching against other IoT deployments, e.g., to detect certificate reuse. The platform then ④ evaluates the submitted reports either by ④ᵃ determining the best known security configuration for the affected devices and scenario based on a protocol-specific grading function or by ④ᵇ checking submitted fingerprints against its database of known insecure credentials. The grading function is actively maintained by the platform operator based on the recommendations regarding cryptographic primitives and protocol options from security guidelines, effectively outsourcing the required security expertise to ColPSA's central platform. Finally, after anonymously receiving the resulting advice, the user can ⑤ apply it to their IoT deployment.

5.1 Network Scanning

When joining ColPSA to receive security advice, a new user first has to scan their IoT deployment (Step ① in Figure 2). The user first installs ColPSA's network scanner, either on one of their devices such as a tablet or as an additional service on their Internet router. The network scanner then collects information on (a) currently active IoT components, (b) each detected component's security configuration, and (c) the use-case context for each component, e.g., whether it is reachable from outside the network. To this end, the scanner combines *active* and *passive* network monitoring as we detail in the following.

Passive Communication Monitoring: As a first step, ColPSA's scanner performs a passive scan of ongoing communication within the user's IoT deployment to learn about active components and the contexts of their communication. When not running as a service on the user's Internet router, the scanner temporarily uses ARP spoofing to reroute all traffic for the required analyses (similarly to [13]). Then, the scanner extracts (a) active devices' internal IP addresses, (b) the number of communication partners for each component, and (c) whether a component is contacted from outside the user's network. ColPSA does not attempt to derive security configurations from the captured packets to keep the load on the scanner low and preserve scalability (R6).

Active Configuration Collection: ColPSA further learns about security configurations by also relying on a more performant active scan phase. While the short passive scan already unveils the majority of active IP addresses in the deployment, some device might only be active occasionally and not yet covered by the scanner. Thus, the scanner also performs an ARP scan, i.e., it requests a MAC address for each IP address considered inactive so far, which actively provokes responses from the remaining devices.

Knowing all relevant IP addresses, the scanner then performs protocol probing and application-layer scans to detect active services on each device. Specifically, the scanner performs full port scans for all identified IP addresses. Depending on the detected protocols, the scanner can gather more details, e.g., perform multiple handshakes to fully reveal a component's TLS configuration.

Finally, ColPSA performs anonymous Internet reachability checks to support the assessment of the user's security configuration based on their intended scenario. Here, the local scanner relies on an external proxy that reflects active scan requests to the device's IP address or the user's public IP address. To protect the confidentiality of the user's setup (R5), the proxy can either be run by the user, a trusted third party, or an exit node of an anonymity network, e.g., Tor. This way, ColPSA's platform operator cannot link any issues to a specific user.

5.2 Data Pre-Processing

After collecting the data, the scanner pre-processes it (Step ②) to retain the information relevant for the subsequent security assessment. Namely, the scanner extracts the types and relevant security settings for identified IoT components, and composes a scenario description per component based on this information.

Component Identification: ColPSA's scanner has to identify all components in an IoT deployment to associate security settings with these components. The scanner relies on established methods (e.g., [19]) that use already collected component information on, e.g., open ports, provided data, and mDNS queries.

Security Settings: Next, the scanner groups the configuration of the running services by device to derive a summary of the security settings and preprocess them for later transmission to ColPSA's platform. We distinguish between discrete settings that can only adopt pre-defined *options*, e.g., cipher suites or used cryptographic primitives, and continuously traceable *values*, such as public keys or certificate fingerprints, for enabling the detection of distributed issues.

Scenario Description: In addition to the configuration of each component, the scanner generates a description of the components' deployment scenarios. Here, the scanner relies on general information about the IoT deployment, e.g., device numbers and fluctuations, as well as per-component information, e.g., it's external reachability and the number of total distinct communication partners.

5.3 Privacy-Aware Data Submission

After the pre-processing step, the user *could* already submit their data to enhance the knowledge available to ColPSA's platform. However, doing so may leak sensitive information about their IoT deployment (R5), and so the user first applies privacy measures (Step ③) before and during transmission as follows.

Wiping Identifying Information: Before uploading data to ColPSA's platform, the scanner removes or replaces data from the security settings that are typically irrelevant for the security assessment and decouples the user's components from being linkable to their IoT deployment (Step ③ᵃ). Namely, the scanner wipes all identifiers, such as hostnames and IP addresses, from the security settings. Furthermore, the scanner creates an individual and independently submitted record for each component, which contains the component's security settings and scenario description. Furthermore, as continuous values, e.g., certificate or public key fingerprints, can still identify a single component, ColPSA separately handles all continuous values in different records (Step ③ᵇ).

Privacy-Preserving Data Submission: Since a direct transmission of the protected records would enable the platform to map records to IoT deployments and their owners, the records are sent via an anonymity network, e.g., Tor.

5.4 Collective Security Evaluation

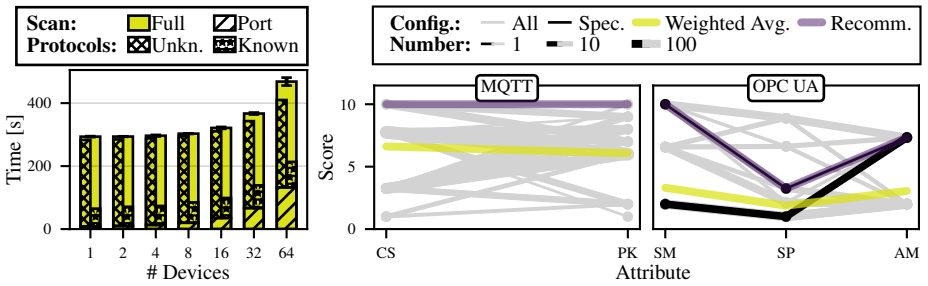
Based on the records received about users' IoT deployments, ColPSA's platform can continuously enhance its knowledge about the most secure realizable configuration and generate advice for users who can improve their security settings (Step ④). To this end, the platform needs to grade the security configuration, identify the component type, determine the scenario it is deployed in, and subsequently generate feedback. Finally, the platform keeps track of continuously traceable values to warn users about reused or otherwise insecure credentials.

Security Grading: When receiving a new record, ColPSA evaluates its security level (Step ④ᵃ) by grading all security options on a scale from *one* (no security, e.g., no TLS enabled) to *ten* (best security level, e.g., using TLS with the most secure, available cipher suite). The grading function is derived and actively maintained by security experts operating ColPSA based on information such as security guidelines (cf. Section 4) or CVE data; this way, the users do not have to individually keep up with these information sources.

Clustering by Type and Scenario: Before selecting the best configuration possible and sending advice to users, ColPSA ensures to only consider applicable (R1) and adequate settings (R2) for the given component type in the same scenario. Thus, the central platform tries to find a matching cluster amongst records received before. To this end, the platform uses the component type already included in the record (cf. Section 5.1) and only considers previously received records matching the same type. Subsequently, the platform clusters the new and remaining records using hierarchical clustering and the scenario information as features. However, ColPSA remains modular and the hierarchical clustering can be replaced by any other machine-learning clustering approach.

Realizable Feedback: After identifying records of similar components used in the same scenario, ColPSA determines the most secure option to create advice for the user. However, the particular protocol-device combination of the user may only allow for certain realizable configuration profiles, e.g., Security Mode and Security Policy for OPC UA (cf. Section 2). Thus, ColPSA cannot necessarily recommend the strongest option for each security setting, but has to find the most secure realizable combination. ColPSA compares the average of all grades in the relevant records to find a secure configuration that has been realized before as the basis for the advice sent back to the user.

Tracking Credential Usage: Besides continuously improving the security assessment for users' security configurations, ColPSA further utilizes the central information hub provided by its platform to keep track of the usage of insecure credentials, such as private keys or certificates (Step ④ᵇ). Namely, the platform keeps track of credentials that are reused across components and IoT deployments. To this end, it counts how often a specific credential is reported by all users. Upon request, users can request the counter values for used credentials in



(a) Active Scan Time (b) Security State and Estimated Improvement

Fig. 3. ColPSA is able to scan subnetworks of different sizes in a manageable time (left) and promises to improve the security configuration of IoT deployments significantly as shown for MQTT and OPC UA (right; Attributes: **C**ipher **S**uite, **P**ublic **K**ey, **S**ecure **M**ode, **S**ecure **P**olicy, **A**uthentication **M**ethod). For OPC UA, it only considers configurations of a specific example device to ensure applicability (R1).

their deployments and compare whether they are the only ones who submitted a specific credential. The platform occasionally wipes its database to reduce its footprint and to improve its users' privacy.

5.5 Security Configuration Improvement

Getting security advice for all components in their IoT deployment, the user now can refine their security configuration (Step ⑤) while being sure that the suggested improvements meet their components' capabilities and match their intended scenario. Additionally, the users get warned about distributed issues that might affect their components, e.g., reused certificates.

6 Evaluation

Processes for giving security advice for IoT deployments should be scalable and provide valid advice for improving security. In this section, we show that ColPSA meets the performance needs of the IoT and generates sensible security advice.

Component and Network Scan: We show that ColPSA scales well for larger IoT deployments (R6). To this end, evaluate the performance of a prototype of ColPSA's local scanner. We mainly implement the scanner in Python but rely on already established methods for active and passive scanning, i.e., we use `arp-scan` [12] for device detection, IoT Inspector [13] for passive, and `nmap` for active scanning to easily support a large number of protocols. We adapt IoT Inspector to offer a headless local operation mode, i.e., we remove its web interface and all cloud connections. Furthermore, we extend `nmap` with an OPC UA module on the basis of `pyopcua`. We simulate IoT deployments of different sizes using Mininet [16]. For each scenario, we add a growing number of devices that each run an OPC UA, HTTP, and HTTPS server, as well as dedicated AMQP and

MQTT brokers; all these components are identifiable by `nmap`. Moreover, each device also uses a protocol not identifiable by our `nmap` version to evaluate the performance impact of unknown protocols. To make our simulation more realistic, all links have a small latency of 0.2 ms. We deploy ColPSA's local scanner on Raspberry Pi 4 Model B connected to our simulation and measure the time required to scan the network. Specifically, we run each measurement ten times and report on the arithmetic mean and 99 % confidence interval (t-distribution).

While the performance of `arp-scan` and IoT Inspector is well-documented [12, 13] and feasible in IoT environments, the time required for the port scan and configuration retrieval during ColPSA's active scan is unknown so far. Figure 3a shows that this phase does only take minutes to complete, even in larger IoT deployments, and thus maintains scalability (R6). Specifically, the scan duration only increases linearly from 294 s with only one device, to 468 s with 64 devices. The reason for this is that the protocol identification after the initial port scans is parallelized, i.e., the scanner investigates all components at the same time to retrieve security configurations. Here, ColPSA requires the most time for components using unknown protocols, as `nmap`'s protocol detection probes for every known protocol (more than 2000) when having no success.

Security Improvement: To estimate the potential security improvement of ColPSA, we rely on configurations of real IoT deployments reachable via the Internet based on previous measurements [3,4]. More specifically, we run ColPSA on the configuration of 12 597 TLS-enabled MQTT brokers scanned on 2023-05-27, and 1651 OPC UA devices scanned on 2023-06-11. Figure 3b (left) details the security configurations of found MQTT brokers (gray) on ColPSA's score scale and also the average score over all configurations (yellow). Given that MQTT brokers typically run on IT hardware, ColPSA does not identify any specific type and recommends (purple) the overall strongest configuration ever seen. It recommends users to configure a significantly stronger cipher suite (some brokers use `TLS_RSA_WITH_RC4_128_SHA` despite RC4's insecurity) and a stronger public key to prevent attackers from performing Person-in-the-Middle attacks.

Figure 3b (right) shows the evaluation results for OPC UA. Given that these servers do not run on commodity hardware, not all theoretically possible configurations are feasible on all devices. Hence, ColPSA considers information on the device model (black) and only takes configurations into account that are known to be realizable in this context. For the example device model, the majority of users did not enforce secure communication. Only a single deployment disabled insecure communications and thereby unveiled a better score than comparable other deployments. Hence, ColPSA can recommend disabling insecure communications to all users in compatible contexts. Based on the configuration seen at a few deployments, ColPSA can help to increase the level of security at scale.

Since users are not identifiable in ColPSA, an adversary could *poison* the platform's gathered knowledge by sending too strong and inapplicable configurations. This attack would lead the platform to make false recommendations to other users. As a mitigation, ColPSA can incrementally roll out security advice and collect feedback about the advice's adoption.

7 Conclusion

The increasing gap between Internet of Things (IoT) deployments such as smart homes and the operating users' security expertise requires strong measures to support the users in configuring their components securely. Current approaches supporting security administration either do not scale to the effort required to maintain IoT deployments of increasingly growing sizes, do not generalize to the complexity of available protocols and devices, or only give theoretical advice without taking the capabilities of individual IoT devices into account. Hence, operating their IoT deployments securely remains a big challenge for end-users.

Our work, ColPSA, remedies this situation by automatically learning the best realizable and adequate security configurations by crowd-sourcing real configurations of IoT deployments. Based on this accumulated knowledge, ColPSA is capable of giving meaningful security advice to its users. Our evaluation of ColPSA shows that collectively gathering security information can help to improve the overall security across IoT deployments without requiring unrealistic security expertise from the users and without invading their privacy.

Acknowledgements. Funded by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) — Research Project VeN^{2}uS — 03EI6053K. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy — EXC-2023 Internet of Production — 390621612.

References

1. Amazon Web Services: AWS IoT Device Defender. https://aws.amazon.com/iot-device-defender/ (2023), (Accessed on 09/01/2023)
2. Avast: Avast Network Inspector. https://support.avast.com/en-ww/article/use-network-inspector (06 2022), (Accessed on 09/01/2023)
3. Dahlmanns, M., Lohmöller, J., et al.: Easing the Conscience with OPC UA: An Internet-Wide Study on Insecure Deployments. In: ACM IMC (2020)
4. Dahlmanns, M., Lohmöller, J., et al.: Missed Opportunities: Measuring the Untapped TLS Support in the Industrial Internet of Things. In: ACM ASIACCS (2022)
5. Dahlmanns, M., Sander, C., et al.: Secrets Revealed in Container Images: An Internet-wide Study on Occurrence and Impact. In: ACM ASIACCS (2023)
6. Durumeric, Z., Wustrow, E., et al.: ZMap: Fast Internet-wide Scanning and Its Security Applications. In: USENIX SEC (2013)
7. Erba, A., Müller, A., et al.: Security Analysis of Vendor Implementations of the OPC UA Protocol for Industrial Control Systems. In: ACM CPSIoTSec (2022)
8. Federal Office for Information Security: OPC UA Security Analysis (2017)
9. Federal Office for Information Security: Cryptographic Mechanisms: Recommendations and Key Lengths Part 4 – Use of Secure Shell (SSH). BSI TR-02102-4 (2021)

10. Federal Office for Information Security: Cryptographic Mechanisms: Recommendations and Key Lengths: Use of Transport Layer Security (TLS). BSI TR-02102-2 (2021)
11. Heer, T., Garcia-Morchon, O., et al.: Security Challenges in the IP-based Internet of Things. Wirel. Pers. Commun. **61**(3) (12 2011)
12. Hills, R.: arp-scan(1) - Linux man page (August 2016)
13. Huang, D.Y., Apthorpe, N., et al.: IoT Inspector: Crowdsourcing Labeled Network Traffic from Smart Home Devices at Scale. ACM IMWUT **4**(2) (2020)
14. Khan, M.A., Salah, K.: IoT security: Review, blockchain solutions, and open challenges. Future Gener. Comput. Syst. **82** (2018)
15. Krombholz, K., Mayer, W., et al.: "I Have No Idea What i'm Doing": On the Usability of Deploying HTTPS. In: USENIX SEC. SEC'17, USA (2017)
16. Lantz, B., Heller, B., et al.: A Network in a Laptop: Rapid Prototyping for Software-Defined Networks. In: ACM Hotnets. Hotnets-IX, New York, NY, USA (2010)
17. Madakam, S., Ramaswamy, R., et al.: Internet of Things (IoT): A Literature Review. J. Comput. Commun. **Vol.03No.05** (2015)
18. Maggi, F., Vosseler, R., et al.: The Fragility of Industrial IoT's Data Backbone: Security and Privacy Issues in MQTT and CoAP Protocols. Tech. rep., Trend Micro Inc. (2018)
19. Meidan, Y., Bohadana, M., et al.: ProfilIoT: A Machine Learning Approach for IoT Device Identification Based on Network Traffic Analysis. In: ACM SAC. SAC '17, New York, NY, USA (2017)
20. OPC Foundation: OPC Unified Architecture — Part 2: Security Model. OPC 10000-2: OPC Unified Architecture (2018)
21. Papadogiannakis, A., Vasiliadis, G., et al.: Improving the performance of passive network monitoring applications with memory locality enhancements. Comput. Commun. **35**(1) (2012)
22. Pohlmann, U., Sikora, A.: Practical Security Recommendations for building OPC UA Applications. Industrial Ethernet Book **106** (2018)
23. Qualys: SSL Server Test. https://www.ssllabs.com/ssltest/ (2023), (Accessed on 09/01/2023)
24. Rahalkar, S.: OpenVAS. Apress, Berkeley, CA (2019)
25. Rescorla, E., Dierks, T.: The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (2008)
26. Roesch, M.: Snort: Lightweight Intrusion Detection for Networks. In: USENIX LISA (November 1999)
27. Serror, M., Hack, S., et al.: Challenges and Opportunities in Securing the Industrial Internet of Things. IEEE Trans. Ind. Informat. **17**(5) (2021)
28. Sha, K., Wei, W., et al.: On security challenges and open issues in Internet of Things. Future Gener. Comput. Syst. **83** (2018)
29. Sheffer, Y., Holz, R., et al.: Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS). IETF RFC 7525 (2015)
30. Srinivasa, S., Pedersen, J.M., et al.: Open for Hire: Attack Trends and Misconfiguration Pitfalls of IoT Devices. In: ACM IMC (2021)
31. Testa, J.: ssh-audit. https://github.com/jtesta/ssh-audit (2023), (Accessed on 09/01/2023)
32. Wetter, D.: testssl.sh. https://testssl.sh/ (2022), (Accessed on 09/01/2023)