Securing Internet of Things Devices Using The Network Context

Michal Trnka, Jan Svacina, Tomas Cerny, *Member, IEEE*, Eunjee Song, *Member, IEEE*, Jiman Hong, and Miroslav Bures

***Abstract*—Internet of Things (IoT) devices have been widely adopted in recent years. Unlike conventional information systems, IoT solutions have greater access to real-world contextual data and are typically deployed in an environment that cannot be fully controlled, and these circumstances create new challenges and opportunities. In this article, we leverage the knowledge that an IoT device has about its network context to provide an additional security factor. The device periodically scans a network and reports a list of all devices in the network. The server analyzes movements in the network and subsequently reacts to suspicious events. This article describes how our method can detect network changes, retrieved only from scanning devices in the network. To demonstrate the proposed solution, we perform a multiweek case study on a network with hundreds of active devices and confirm that our method can detect network anomalies or changes.**

***Index Terms*—Context-awareness, anomaly detection, authentication, Internet of Things (IoT), security.**

I. Introduction

CURRENTLY, access to the Internet is available to many devices that we use in our everyday lives; such devices are described by the term "smart objects" [1]. The Internet of Things (IoT) consists of smart devices that cooperate with each to provide complex services [2]. It is expected that the number of IoT devices will significantly increase in the near future; predictions state that there will be 20.5 billion devices in 2020, with over three trillion U.S. dollars spent on hardware alone [3].

Authentication and authorization in an environment with many devices that are heterogeneous and that dynamically connect and disconnect from a network are challenging. For example, Gartner Inc. [3] states the following: *Security and risk concerns will continue to be the greatest impediment to IoT adoption. The market for IoT specific security solutions will dramatically expand in 2017 as existing security providers aggressively retool existing capabilities to address IoT security risks.* Another security survey [4]–[6] confirms that authentication is a major open-ended issue.

The IoT intensifies security challenges compared to traditional applications. Sensors are often deployed in an environment over which we do not have full control. Furthermore, devices, applications, and communication schemes are heterogeneous. All of this makes device authentication more difficult and gives potential attackers more opportunities. Another problem is that a correctly authenticated device can provide malicious data if attackers gain access to its environment (which can be virtual or an actual real-world place). We aim to extend the authentication model to take its surroundings into account.

One general idea of how to enhance security is to consider the context of the device [7], and attempts to leverage context information for security applications are more than 15 years old [8]–[10]. The context provides an explanation for the data provided by any participant and allows us to better understand the participant's situation. With more information about IoT devices, we can enhance an authentication process to improve its overall reliability. In this article, we propose a method of extracting the context from an IoT device and using it as an additional factor for authentication. Our core concept is that, together with traditional methods (e.g., credentials), the context of a device can play a role in its authentication. We particularly focus on the network neighborhood of a device. The idea is to create a virtual map of surrounding devices and to track their changes and patterns over time. If an unexpected and significant change occurs, a red flag can be raised, and further actions can be taken. A significant change is marked by the percentage of devices that have changed, and the method of determining the percentage is illustrated in Section V.

The contribution of the article is summarized as follows.

1) We present an additional authentication factor for usage in the IoT environment.
2) We illustrate how to set up our method in a network.
3) We discuss how various settings affect the proposed method and how to determine the ideal values.
4) We demonstrate the feasibility of the method based on a network with hundreds of unique devices and we

Manuscript received June 15, 2019; revised October 10, 2019; accepted November 11, 2019. Date of publication November 18, 2019; date of current version February 28, 2020. This work was supported in part by the National Science Foundation under Grant 1854049 and in part by the OP VVV funded Project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics." Paper no. TII-19-2530. *(Corresponding author: Michal Trnka, Jiman Hong and Tomas Cerny.)*

M. Trnka and M. Bures are with the Department of Computer Science, Faculty of Electrical Engineering, Czech Technical University in Prague, 121 35 Prague, Czech Republic (e-mail: trnkami1@fel.cvut.cz; buresm3@fel.cvut.cz).

J. Svacina, T. Cerny, and E. Song are with the Department of Computer Science, Baylor University, Waco, TX 76798 USA (e-mail: jan_svacina2@baylor.edu; tomas_cerny@baylor.edu; eunjee_song@baylor.edu).

J. Hong is with the School of Computing, Soongsil University, Seoul 156-743, South Korea (e-mail: jiman@ssu.ac.kr).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2019.2954100

provide experimental data allowing better insight into and applicability of the method.

This article is organized as follows. Section II summarizes the background of the problem. Section III provides an overview of related work. In Section IV, we describe our proposed method for context-aware IoT authentication using the network context of a device. Section V evaluates the proposed technique and describes our experimental results and experimental settings. Section VI elaborates possible threats to validity and facts that are mitigating them. Section VII discusses the results. Section VIII concludes this article.

II. Background

An extensive overview of IoT context-awareness research is provided in the survey by Perera *et al.* [11]. This article shows the existing approaches to the IoT and distributed environments and identifies various context types. Related work often considers different context categories, such as user, identity, location, time, activity, or even the environment with its available resources. Various sensors can extend awareness with highly dynamic information, e.g., measuring temperature or biometrical functions. Utilizing the device constellation for context awareness, which our research aims to do, is uncommon. The survey mentioned above [11] explicitly mentions that security and privacy issues in context-aware computing are not seriously evaluated and considered in many existing solutions. However, if security is addressed, then it is related to sensor data, locations, preferences, or user details. In the extensive overview above, the authors shows that approximately one-fifth of context-awareness proposals mention security and privacy as trending topics.

Initial proposals operated with limited context information. Technical restrictions allowed access only to either the context data that the user provided about him or herself or data about the server side. A particular user provided the application with the communication history and with the timestamp for every part of the interaction and his/her virtual address, represented by the IP address. The server side could enrich the known context with a few extra properties—for example, the CPU load, memory usage, operating system, and user activities. Nevertheless, those early proposals provide valuable architectures from which we can evolve. Covington *et al.* [8] described enhancing role-based access control with environmental roles. Architecture with four context elements (context owner, context provider, context broken, and context-aware service) was discussed by Bhatti *et al.* [9]. One of the earliest methods to dynamically calculate context-based properties used for authentication was proposed by Hulsebosch *et al.* [10].

The IoT consists of various elements that often utilize connections to the Internet. In particular, end devices are not connected to the Internet directly but through some kind of usually private subnetwork.[1] These networks are diverse—they can be wireless or wired, their size can vary from a few devices (e.g., a smart home) to thousands (a smart city), and the participants in the network can be simple sensors or sophisticated services providing complex functions. The devices involved are often provided by various vendors and have a heterogeneous set of properties.

Networks do not vary only between themselves; a single network may have different structures and devices at various times. This is especially true in the dynamic IoT environment, where devices regularly connect and disconnect. The idea behind the proposed method is that there might exist recurring patterns in the network signature that can indicate regular behavior, based on which we can find anomalies. Some devices may connect or disconnect regularly at similar times; however, some devices may be connected all the time, while others may connect just once.

This property of IoT solutions can be leveraged for IoT device context-aware security. A device can regularly report the state of its network, and the server will determine the security of the device based on the recurring patterns in the network and the changes that have happened. Naturally, this method cannot provide the same authentication power as a key authentication or similar methods. However, as an additional factor, it can be an excellent complement. This method brings the ability to trace suspicious events and anomalies in the network. Such changes can be caused by, e.g., connecting unauthorized devices to the network, trying to eavesdrop on communication, performing any variation of a man-in-the-middle attack, sending spurious data, or even trying to hijack a device.

III. Related Work

Location-based authentication has been explored in the area of indoor smart environments [12], [13]. In general, such works attempt to identify a particular user by using proximity sensors and customize services according to user preferences. Al-Muhtadi *et al.* [12] introduced a confidence level for the authentication; their proposal requires having full knowledge about the environment retrieved from sensors and other elements in the smart environment. In contrast, our method uses only information that is available from the network. A similar topic is explored by [14] and [15], who predict future movements and locations. They do not explicitly mention using this prediction for authentication; however, comparing real with predicted locations can clearly be used as a factor for authentication.

Location awareness for authentication in an IoT involving smartphones is considered in [16] and is applicable to processes such as an online purchase or car unlocking. This article considers the possibility of modeling user location and user movements in the IoT. The authors suggest utilizing multiple devices of a given user to better prevent malicious activity, and they indicate that the state of the art is going in the direction of multifactor authentication to prevent fraud, e.g., by combining biometrical information with a secret password and a token. However, for performing routine operations, this approach is clearly not user-friendly. Location and movement can bring an additional factor to improve authentication, which the authors consider. Their solution involves location derived from multiple user things, e.g., smart wear, cars, body sensors, and mobile devices. They suggest that location can be derived not only

[1] In this article, we use the term "network" for the Internet subnetwork.

through GPS but also through Wi-Fi signals or proximity to other devices with known coordinates. They also suggest using access points and home hubs to assess the location and the use of connected devices. They highlight important privacy concerns and suggest that remote service providers cannot ask direct questions, such as where the user is; however, they could ask whether the user could be physically near a particular location. Their protocol does not reveal device location; rather, it considers relative distances. Interestingly, the article observes that 79% of people aged 18–44 have their cellphone with them 22 h a day. Their envisioned methodology is complicated; it would require a user to define a private hub, which would have to be linked to a dedicated device or a cloud service, that aggregates various devices in a hierarchy; the hub would collect device information but would not forward it directly to a third party. Furthermore, third-party services are expected to interact with the user's hub; upon receiving a notification signal, third parties (e.g., banks) would then query the hub for the user's location. This methodology would significantly change the interaction pattern for server-based systems. Moreover, the approach is data intensive and eager since devices interact with the hub even when no service is called upon for days, which could lead to congestion when there is large-scale use. Furthermore, the approach utilizes only user devices registered to the hub; thus, visiting a new environment does not expand the context awareness. In their evaluation, they analyze the prototype service at the Amazon EC2 cloud, revealing a 4–6% false incidence of service denial.

Multifactor authentication for Smart Homes using biometric information and hardware signature is described in [17]. It smartly combines face recognition with photo response nonuniformity, which is a hardware fingerprint that can be used to uniquely identify smartphones. Using this authentication, user takes a photo of his face and this image is used for authentication with IoT hub. In comparison to our method, this works only for user authentication and cannot be user to machine authentication.

The innovative usage of Bayesian decision theory [18] for authorization also considers context. It defines three trust parameters—the history of previous interactions, public and private knowledge about the peer (which includes contextual information), and the peer's reputation with other devices. By definition, biometric information is a subset of contextual data; thus, security systems that use such information [19], [20] are context aware. Abovementioned methods focus more on the trust between two participants than on changes in the context, as we propose.

Wi-Fi networks have been used to enhance security in an area, where some results have been obtained. All of the proposed methods share a common property—they do not require user cooperation. Some of them work even if the user does not carry a device (not even a Wi-Fi device such as a cell phone). There are various methods for acquiring a user's precise location based on the Wi-Fi signals of his or her devices [21], [22]. Such methods use a standard Wi-Fi router to measure the position of the user; the position is determined with median precision under 30 cm based on changes in the signal-strength, amplitude, etc. Accordingly, another IoT authentication perspective has been suggested by [23], who consider using Wi-Fi signals to assess identification. Their aim is to use a device-less approach; thus, the person profiled does not have to possess a smartphone or smart device. This approach utilizes Wi-Fi signals to capture unique human psychological and behavioral characteristics derived from the person's daily activities. They extract the channel state information (CSI) for the given Wi-Fi and develop a deep learning scheme to identify users. They test their concept in a small indoor environment with 11 subjects, achieving authentication accuracy of 91–94%. This approach can fit well with smart homes, preventing children from using the oven or operating potentially dangerous appliances. The method exploits the signal evaluations of individuals performing daily tasks, measuring the CSI amplitude and relative phase of signals among the devices in a room. For the same activities, different individuals exhibit different impacts on the wireless channel. First, the approach must recognize a particular activity, and then, it can start identifying a particular individual. Clearly, the major drawback is that for the system to identify users, it must learn about them first. Additionally, used at a large scale, it would have difficulty identifying a particular individual among many users; the evaluation was performed one person at a time. Similarly, in smart indoor environments, a recent study [24] utilizes features from CSI variations caused by the walking gait of humans. It uses signals to identify particular individuals, and only a few steps are needed to identify an individual. The researchers had an accuracy rate of 80–92%; however, the experiment involved only a small group of individuals. Various other CSI studies, including [25], have also been conducted primarily indoors. Another approach is to use the Wi-Fi device that users carry with them [26]. These devices can be used to assess the particular network and acquire knowledge about the devices connected to the network, with changes in the patterns of the devices being used as an additional factor during authentication. This research has been conducted on real users, and it is questionable whether it is transferable to the IoT environment for machine-to-machine authentication.

In our proposal, we try to use the network context to detect suspicious events in the IoT device neighborhood. In this regard, it is partially similar to network anomaly detection. Based on a comprehensive study focused on this topic [27], our proposal would be categorized as rule based, with network information being retrieved using probes that defend against network attacks and physical attacks. Because IoT networks and the environment are generally specific and slightly different from traditional networks, IoT-specific anomaly detection methods have been proposed. One of the oldest examples is the system called [28], which specifically focuses on 6LoWPAN [29] networks. Another promising method [30] involves using an artificial neural network to detect Distributed Denial of Service / Denial of Service. Possibly (Distributed) Denial of Service (DDoS/DoS) attacks. Our approach differs in two ways. First, we aim to enhance the security of a single device using knowledge about its surroundings, not the whole network. Second, we do not analyze network traffic, as doing so is resource demanding for any device; rather, we take snapshots of the network state and use them to compare changes in the network.

With respect to the related work, the closest approaches that we identify are as follows: Agadakos *et al.* [16], who consider

location awareness and multiple "registered" devices that help with authentication; Shi *et al.* [23], who consider Wi-Fi signals to recognize individuals with a device-less approach; and Trnka *et al.* [26], who consider context as an additional factor during authentication. We do not aim to replace the utilization of other contextual elements, such as those mentioned in [11]. Instead, we aim to augment the possibilities and to assess recurrent patterns in the behavior of networks and devices.

IV. Proposed Method

The idea of the proposed method is based on the regular network context reports provided by every IoT device. They retrieve a list of all devices discoverable in the network and send it to the server regularly. Ideally, that information is passed along during every server request. Due to the network bandwidth, storage and computation capabilities of the server and other limiting factors, it can be restricted to a certain reasonable time frame (e.g., every 15 min) to reduce communication overhead. The server subsequently stores the data for further use, evaluates the received data, and eventually proceeds with further actions. Such actions may include an additional authentication request to the suspicious device (which may or may not be the device that triggered the action), a notification to a network administrator or even a limitation to or removal of network access for the suspicious device. A network context scan is performed on end devices, and the server performs only a context evaluation, which results in great scalability.

The utilization of our approach and its full possibilities requires a significant amount of contextual data gathered over extended periods of time, preferably in various distinct physical locations, across multiple different networks and mainly with the knowledge of the security incidents that happened. Given such an extensive dataset, it can be analyzed using standard algorithms based on decision tree induction [31] or advanced adaptive fuzzy rule-based classification [32]. Once the patterns are recognized, they can be searched for in real time, and appropriate control mechanisms can be activated as needed. Unfortunately, we do not possess such a dataset. Therefore, in this article, we propose a method to analyze the network context of a particular device.

The method utilizes "recurring" devices for analyzing network context. A recurring device is a device that has been in the network in several consecutive days. For example, such a device is typically present in the network at particular time. Internet follows standard open systems interconnection (OSI) networking model [33], possibly with media access control (MAC) layer protocols adapted for IoT devices [34]. Therefore, MAC addresses are used as device identifiers because, by definition, they are unique. Problem of their potential counterfeit is not significant in most of the scenarios as multiple devices with spoofed MAC address would need to be introduced into the network. The possibility of the attacking device changing its MAC address does not affect our method more than any other device with spoofed MAC address, as this MAC address is treated as one of the addresses on the network. Potential successful attack targeting our method would lead to higher ratio of false positives, which would not affect user experience or security (in comparison not to use our method as additional factor at all).

The recurring device list is created specifically for a given network. While it is possible that a recurring device will be a recurring device in more than one network, this is rarely the case.

A. Illustration of The Proposed Approach

An example of such a situation is a personal device carried by a user; the device is in a network during the day when the user is at work but in a different "home" network at night.

Recurring devices are determined based on historical values that are stored by the server. If a device appears in the network at the same time over multiple consecutive days, it is marked as a recurring device. Recurring devices are determined from a limited historical time frame (e.g., the last five days), and therefore, the set of recurring devices can vary from one day to the next. When the process is started, recurring devices cannot be determined, as there is no reference point. A list of recurring devices can be made when the time frame passes (e.g., five days). Recurring devices are calculated every day given the historical values. The algorithm takes all devices from the first day and marks them as candidates. Every subsequent day, it removes devices that are not present during the day from the list of candidates. After all steps are completed, the candidate list is the final list of recurring devices.

Fig. 1 illustrates the three-step determination of recurring devices. The steps illustrate a network at the same time over three consecutive days. The sample network consists of 16 various devices; thus, we can easily visualize it. Real networks often contain hundreds of network elements. During step 1, all devices are considered recurring device candidates. In step 2, there are the same six devices as in step 1. These are new recurring device candidates. In the final step, four devices from the candidate list are present. This list is a new list of recurring devices and can be used on the following day. Each step represents a single day. After the first step, we cannot determine recurring devices because there is no reference point. On day two, we make a list of candidates with the devices that have been active during both days; on the following day, the list of candidates is reduced again. If the number of previous records is larger than that from the time frame used for determining a recurring device, then some devices can also be added.

During communication, a device sends the list of all reachable devices in the network. The same rules described above are applied for the determination of recurring devices; thus, the device does not need to obtain the list for every request. The server compares the sent list with the list of recurring devices (which we call the benchmark) for the given network for a roughly similar time frame. It also uses the provided list to modify and verify the benchmark for the following days. Our preliminary implementation of the approach can configure the desired recurring device match with the devices in the network. Fig. 2. illustrates a network with 16 devices and a set of recurring devices consisting of four devices from the previous figure—A, D, I, and L. In this example, the match

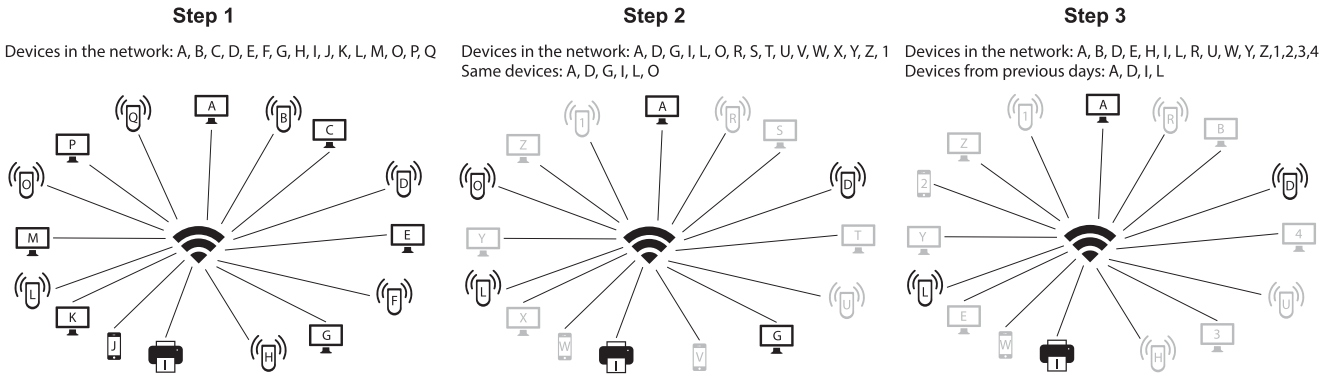

Fig. 1. Creation of recurring devices set in three steps.

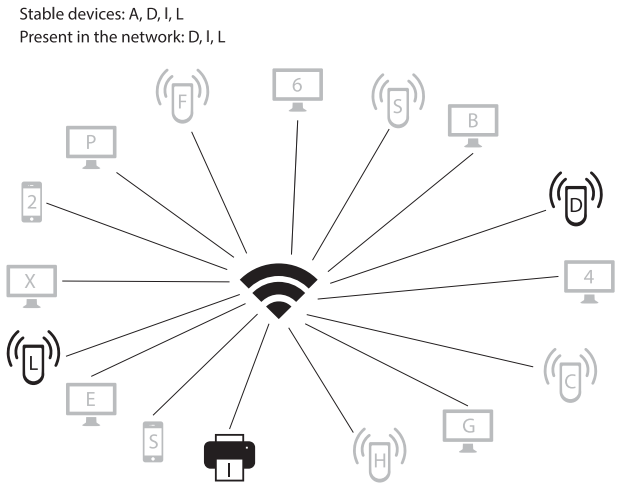

Fig. 2. Using network context to determine changes in the network.

is 75% (device A is missing). If the threshold is not met (e.g., 70% match), then the network context of the device is marked as suspicious, and further steps can be taken—the administrator is notified, an additional authentication factor can be invoked, or a more sophisticated network search for malicious devices can be triggered.

B. Problem Model and Proposed Algorithm

We model the analyzed network as a set of devices $N = \{n_1, n_2, \ldots, n_n\}$, where device n is every network element with MAC address. Time frame $t = (t_{\text{start}}, t_{\text{end}}), t_{\text{end}} - t_{\text{start}} < 1$ day is a time period during a single day. Times t_{start} and t_{end} can be equal; in such a case, the timeframe t is not an interval, but a time point. Age (denoted as age) is a number of consecutive days, for which the benchmark is created. We denote the day in which the analysis is performed as d.

Benchmark is $B(t, d, \text{age}) = \bigcap_{x=d-age-1}^{d-1} \text{devices}(N, t, x)$ where $\text{devices}(N, t, x)$ denotes set of devices present in the network in a randomly selected time from the time frame t during the day x.

We define $\text{match}(t, d, \text{age}) = \frac{B(t,d,age) \cap N(t,d)}{B(t,d,\text{age})}$ as the ratio between number of devices in the benchmark and number of devices in the benchmark present on the network, where t is a time frame and d is a day in which the analysis is performed.

Then, Threshold is a value of $\text{match}(t, d, \text{age})$ such that if $\text{Threshold} > \text{match}(t, d, \text{age})$ the authentication check (as introduced in Section IV-A) is passed.

In Algorithm IV-B, we describe the process to determine the threshold and age. The algorithm accepts the following inputs.

1) Set of all analyzed time frames T.
2) Analyzed network N.
3) Constant ε defining when to stop the algorithm.
4) Constant lim which is the number of days for which we run the algorithm.

The outputs of the algorithm are as follows.

1) Age_{opt}, which denotes the optimal age.
2) Threshold, which denotes maximal possible threshold for given lim.

The principle of the Algorithm IV-B is the following.

1) For network N, create a set of benchmarks for defined set of ages (2 to lim) (lines 6–24). During this process, two principal activities are conducted:
   a) Compare the last created benchmark with a previously determined best benchmark which of them the best value of match function. If the latest benchmark has better value of match function, consider this benchmark as the best one (line 15).
   b) When the value of match function of compared benchmarks starts converging to meet the algorithm stopping criteria defined by ε, return the best found age, denoted as Age_{opt}. (line 19)
2) For a time span from $\text{Age}_{\text{opt}} + 1$ to lim determine Threshold such that all match for each of the analyzed time spans are equal or higher than Threshold (lines 25 to 33)

V. Experimental Verification

To verify the proposed method using a real network, we conducted a case study described in this section. To demonstrate validity of the proposed approach, we performed: 1) evaluation using a real network and 2) simulation of the network with various possible events that could happen (e.g., recurring device disappearance or MAC address spoofing). Details are presented in the following sections.

Algorithm 1: getAgeAndThreshold(T, N).

Input : Timeframes T, Network N, ε, lim
Output: Age_{opt}, $Threshold$

$devices(N,t,d)$ = set of present devices in N for t and d, $t \in T$, d is day
$B(t,d,age) = \bigcap_{x=d-age-1}^{d-1} devices(N,t,x)$
$match(t,d,age) = \frac{B(t,d,age) \cap N(t,d)}{B(t,d,age)}$
$Age_{opt} \leftarrow 0$
$Match_{val} \leftarrow 1$
for $d = 3 \ldots lim$ **do**
  for $age = d-1 \ldots lim-1$ **do**
    $Match_{min} \leftarrow 1$
    for *each* $t \in T$ **do**
      $Match_{tmp} = match(t,d,age)$
      if ($Match_{min} > Match_{tmp}$) **then**
        $Match_{min} = Match_{tmp}$
      end
    end
    if ($Match_{min} > Match_{val}$) **then**
      $\Delta Match = Match_{min} - Match_{val}$
      $Match_{val} = Match_{min}$
      $Age_{opt} = age$
      if ($\Delta Match < \varepsilon$) **then**
        goto 25
      end
    end
  end
end
$Threshold \leftarrow 1$
for $d = Age_{opt} + 1 \ldots lim$ **do**
  for *each* $t \in T$ **do**
    $curMatch = match(t,d,Age_{opt})$
    if ($curMatch < threshold$) **then**
      $Threshold = curMatch$
    end
  end
end
return $Age_{opt}, Threshold$

A. Real-Network Evaluation

Initially, we determine relevant time frames for a benchmark. Then, we determine whether the exact same time of the day needs to be used for the measurements during various days or whether an alternatively approximate interval can be used. Once we have such values, we proceed to determine a threshold for the percentage of recurring devices in the network based on network historical data.

We perform five weeks of measurement in the same network and conduct six control measurements. As our network, we select a Baylor University Wi-Fi network in the Department of Computer Science with hundreds of unique devices. We choose this network for the experiment because it provides a considerable number of devices in which users periodically connect and disconnect (e.g., students' devices) with various schedules and devices that are always present (e.g., printers). We perform six analyses per day, evaluating the network only during weekdays. Three analyses are conducted at fixed times—08:00, 12:00, and 16:00—and three are conducted at random times within specific time intervals representing morning (07:30–10:00), midday (11:00–13:00), and afternoon (14:00–17:00).

TABLE I
DAY 11 BENCHMARK AGE DIFFERENCE TIMES: DIFFERENT BENCHMARKS FOR SPECIFIC DATE

	8:00	12:00	16:00	Morning	Noon	Afternoon
Devices count	272	620	931	309	581	560
2 day benchmark size	71	128	156	90	128	138
2 day recurring devices	47	77	86	58	69	84
2 day recurring devices	66%	60%	55%	64%	54%	61%
3 day benchmark size	51	70	90	46	70	80
3 day recurring devices	36	54	65	32	49	57
3 day recurring devices	71%	77%	72%	70%	70%	71%
4 day benchmark size	41	50	67	38	46	58
4 day recurring devices	30	39	53	28	33	45
4 day recurring devices	73%	78%	79%	74%	72%	78%
5 day benchmark size	35	41	54	30	37	44
5 day recurring devices	28	33	44	24	26	36
5 day recurring devices	80%	80%	81%	80%	70%	82%
6 day benchmark size	31	31	35	26	29	30
6 day recurring devices	26	26	31	21	20	25
6 day recurring devices	84%	84%	89%	81%	69%	83%
7 day benchmark size	25	27	30	22	24	25
7 day recurring devices	21	23	26	17	18	21
7 day recurring devices	84%	85%	87%	77%	75%	84%

Initially, we aim to determine how many days are needed for the benchmark. We run the algorithm for 11 days. We run it twice—once for the fixed time frames and once for the intervals. We show up to a 7-day benchmark for Day 11 in Table I with different benchmark periods. There is a gradual decrease in the benchmark size from over 100 devices in the two-day benchmark down to 25 devices in the 7-day benchmark. Theoretically, as these devices should be more stable, the percentage of recurring devices found increases, which is also generally the case on the example day. Note that for the last three days, the benchmark size and percentage do not vary considerably. This finding leads us to the conclusion that adding more than five days provides only limited benefits; thus, we choose five days as our benchmark period. Those findings are consistent across all measurement times, even for those taken randomly within an interval. We illustrate the percentage of recurring devices found for various benchmark periods for all days in Fig. 3, where the 12:00 and midday measurements are used. The randomized interval measurements are illustrated on the right graph, and they fluctuate significantly more than the measurements taken every day at the same time, which are on the left graph. Note that only weekdays are used; thus, day 6 corresponds to a Monday. The algorithm yields five-day benchmark that provides a percentage[2]

[2]Comparing the minimal value from the 25 days.

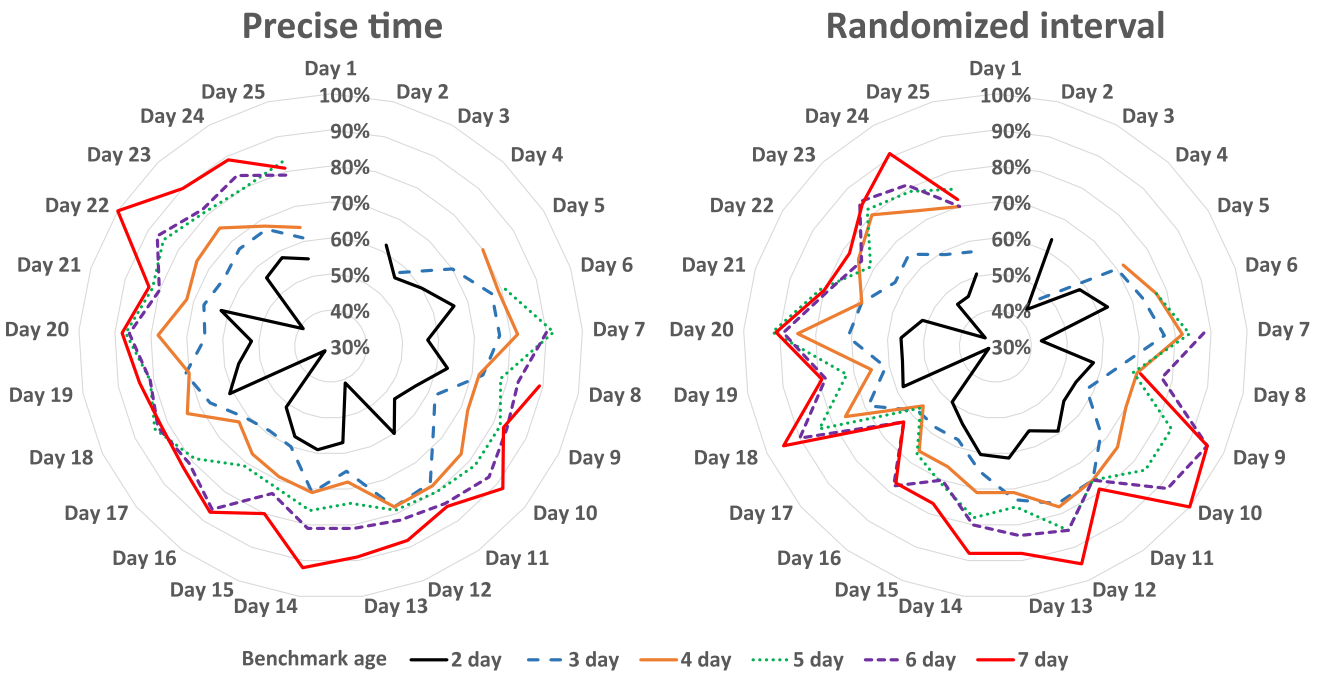

Fig. 3. Percentage of recurring devices for ever day using different benchmark age.

TABLE II
DAY 11 MEASUREMENT: DEVICES ON THE NETWORK IN THE SPECIFIC TIMES AND INTERVALS WITH 5-DAY BENCHMARK AGE

	Devices count	Benchmark size	Recurring devices count	Benchmark match
8:00	272	35	28	80%
12:00	620	41	33	80%
16:00	931	54	44	81%
Morning	309	30	24	80%
Noon	581	37	26	70%
Afternoon	560	44	36	82%

nearly as good as that of the benchmarks consisting of a longer period, with differences of only approximately 2% from the 6-day benchmark and 5% from the 7-day benchmark age, while also providing better stability than the 7-day benchmark.

The next unknown piece is the difference between the measurements taken at strictly the same time and those taken during the same interval. Randomized measurements decrease the possibility of intentionally spoofing the network and providing fictitious MAC addresses to inflate the set of recurring devices. As in the previous paragraph, we use day 11 to demonstrate our findings. However, we now choose only a 5-day benchmark and illustrate the number of devices in the network, the benchmark size, the recurring device count, and the benchmark match for every time in Table II. For every time or interval, we use the corresponding times or intervals on previous days to determine the benchmark. The table shows that there is a noticeable and randomly occurring decrease in the match percentage between the interval and corresponding fixed time measurement. We choose to continue the case study with fixed time measurements because they provide higher consistency. This higher consistency is also confirmed by the measurements in Table III, where the recurring devices for a specific time never drop below a 73% match, while the interval measurements can drop as low as a 65% match.

With the benchmark period set using fixed times for the measurement strategy, to run the control measurements, the only part that is missing is an optimal threshold for validation of the network context. Algorithm IV-B gave us output of 76% as maximal threshold, we choose to lower it to 70% to give us some safety margin. Table III presents the network evaluation for every day and time or interval during our study using the 5-day benchmark. Day 1 in the table corresponds to the Monday of the first week of the case study, with days 6, 11, 16, and 21 also being Mondays. The number of devices in the network varies from 250 to over 1000, with Fridays and parts of Monday being the days with the fewest devices and mornings being the time with the lowest number of active devices. However, the percentage of recurring devices is fairly consistent, never reaching below 73% across all days and times.

Five control measurements are conducted to verify the ability to detect changes in the context. The measurements are compared to the 5-day benchmark from previous days based on the base network at Baylor University. All measurements are taken at 12:00 to allow an exact match with the benchmark, which should give the highest similarity. The first control measurement is taken in a completely different environment to validate the capability of detecting an environment that significantly changes on day 6. This measurement is taken at a grocery store, and a match with single devices of only 3% is achieved. Another measurement is taken again in a completely distinct environment but with some devices from the base network regularly appearing

TABLE III
DAY OVERVIEW OF THE 5-DAY BENCHMARK: NUMBER OF DEVICES ON THE NETWORK AND RECURRING DEVICES FOR EACH MEASUREMENT DURING EVERY DAY

	8:00		12:00		16:00		Morning (07:30-10:00)		Noon (11:00-13:00)		Afternoon (14:00-17:00)	
Day	Devices	Recurring devices	Devices	Recurring devices	Devices	Recurring devices	Devices	Recurring devices	Devices	Recurring devices	Devices	Recurring devices
Day 1	343	N/A	769	N/A	729	N/A	568	N/A	599	N/A	568	N/A
Day 2	447	N/A	615	N/A	628	N/A	606	N/A	585	N/A	722	N/A
Day 3	349	N/A	645	N/A	753	N/A	337	N/A	629	N/A	781	N/A
Day 4	365	N/A	546	N/A	521	N/A	389	N/A	715	N/A	534	N/A
Day 5	245	N/A	456	N/A	557	N/A	191	N/A	537	N/A	580	N/A
Day 6	271	N/A	546	N/A	696	N/A	314	N/A	703	N/A	651	N/A
Day 7	429	96%	566	90%	573	76%	449	100%	518	88%	653	76%
Day 8	261	82%	691	83%	715	86%	182	69%	627	73%	705	82%
Day 9	416	100%	656	84%	514	82%	491	100%	578	89%	532	82%
Day 10	252	86%	540	87%	520	81%	280	96%	562	96%	446	76%
Day 11	272	84%	620	84%	931	89%	309	81%	581	69%	560	83%
Day 12	510	79%	772	82%	701	86%	198	79%	611	92%	577	83%
Day 13	316	79%	830	81%	981	82%	317	86%	765	83%	632	76%
Day 14	538	81%	689	81%	728	86%	162	79%	600	87%	746	88%
Day 15	320	77%	726	74%	709	78%	175	76%	677	73%	451	83%
Day 16	436	88%	811	86%	1166	76%	352	86%	600	84%	877	82%
Day 17	571	85%	765	81%	1004	79%	626	79%	720	66%	563	76%
Day 18	360	83%	1304	83%	1247	80%	351	88%	1206	83%	703	76%
Day 19	611	81%	938	81%	823	79%	549	81%	885	85%	664	75%
Day 20	304	82%	676	86%	739	81%	430	79%	966	89%	683	65%
Day 21	405	87%	1168	80%	928	73%	480	75%	1180	82%	829	77%
Day 22	564	79%	968	87%	723	77%	689	79%	994	81%	664	95%
Day 23	400	91%	818	82%	1147	78%	496	91%	1252	88%	986	72%
Day 24	602	92%	687	84%	708	75%	581	77%	740	81%	603	79%
Day 25	309	92%	576	79%	548	84%	378	78%	519	74%	386	75%

there. The place that is chosen is an apartment complex with a considerable number of Baylor students. However, there are zero matches, most likely because the network is segmented into smaller subnetworks that we were unable to scan. Two other measurements are taken in a partially similar environment where a high number of common devices can be expected. The chosen places are locations within Baylor University but outside of the base network, with a significant number of devices flowing between these networks. They provide a match of 5% and 0%, confirming that places with high fluctuation in the same devices are not matched. For the last control measurement, we choose our base network but during the weekend to verify that we can also detect a change in the context in the main network. During the analysis, there was less than one-fifth the normal number of devices, and the match was only 29%. All of the control measurements obtained values significantly lower than the threshold of 70% set in the previous paragraph. An overview of the results is presented in Table IV, including the benchmark size and the number of recurring devices found for every day of the measurements.

We evaluated the performance of this method in a network. address resolution protocol (ARP) scans are used to determine the devices available. Therefore, with our method, every device receives an ARP request. We evaluate the performance in a network with 254 addresses. With six devices scanning, the network simultaneously increases the latency in the network (measured between two other devices) from 2 ms to between 13 and 20 ms. A full scan of the network with 254 addresses takes slightly under 3 s.

TABLE IV
CONTROL MEASUREMENTS

Place	Devices	Bench.	Recurring devices	Percentage	Day
Supermarket	595	37	1	3%	6
Apartment	8	38	0	0%	9
Dinning hall	1095	38	0	0%	7
Commons	42	38	2	5%	9
Saturday	118	38	11	29%	10

This verification shows that we can detect anomalies in the network and provides data that illustrate this ability in a network with hundreds of users active at the same time. It demonstrates how 5-day benchmark was chosen as the ideal benchmark age, it explains when measurements taken at random times in an interval are better for analyzing networks than measurements taken at the same fixed times, and it describes the process for determining the optimal threshold value for this particular scenario. The control measurements demonstrate the ability to detect an unfamiliar context in numerous networks with different

TABLE V
SIMULATION WITH THRESHOLD 70%

Simulation	Simulation match	Original match	Threshold	Simulation classification
Failure same day	78.04%	80.48%	70%	True positive
Failure day before	80.00%	80.48%	70%	True positive
Adverse device	80.95%	80.48%	70%	True positive
Attack with 15 devs.	28.95%	49.06%	70%	True negative
Spoof attack	28.95%	31.57%	70%	True negative

characteristics or at a different times in the base network. This method alone cannot be used for device authentication, but it can serve as an additional factor during the authentication process. With an unfamiliar or suspicious network context, actions such as further authentication or time or resource-intensive network analysis can be taken. An example of a suspicious network is one involving the sudden appearance of a significant number of unknown devices.

B. Simulation

In this section, we simulate behavior of the network in potential situations that did not occur during our five weeks real-world evaluation, but are of a significant concern. For the simulation, we use the measurements from the real-world network and we adjust them to the particular scenarios by removing or adding the devices into the measured data. We explore cases that could potentially lead both to false negative and false positive classification. For initial simulations, we choose the day 11, time 12:00 from our measurements. For latter scenarios that could lead to false positives, we choose the Saturday following the day 10 and again 12:00 time as we have data for it in the control measurements. Results are summarized in Table V and described below.

The first simulated case is failure of the stable device. This can be divided into two events. The device can either fail before the measurement is taken, which means that it is not included in the current benchmark. Or it can fail on the same day and, therefore, is included in the benchmark. Failure on the same day decreases number of recurring devices from 33 to 32 and, therefore, match decreases from 80.48 to 78.04%, which is well above the threshold. Failure of the device in the preceding days decreases both benchmark size from 41 to 40 and number of recurring devices to 32, which leads to 80.00% match. Again above the threshold we set. The only measurement, where failure on the same day would lead to false negative is day 21 in 16:00 as it would decrease to match to 68.23% (failure on the day before would only decrease the match to 72.00%).

The second scenario is when an adversary is present on the network from the beginning. This leads to the increase of the benchmark and stable devices are found. In our simulation, it increases match from 80.48 to 80.95% with benchmark increase of one to 42 and number of recurring devices increase to 34.

Third case simulates wider attack to the network, with malicious 15 devices present on the network. This increases number of devices on the network to 133, benchmark size to 53, and number of recurring devices from 11 to 26. It leads to match of 49.06%, while the match without the attack was 28.95%. Given our network and the specific day, attack would need to consist of 52 devices to reach our threshold and, thus, lead to false positive.

Fourth case simulates an attack where the malicious devices spoofs the MAC address to one of the benchmark addresses not present on the network. Presence of the device increases number of recurring devices to 12 and the match from 28.95 to 31.57%. Overall, 11 devices in an coordinated attack would be needed to lead to false positive. Therefore, we identify this as the weakest part of our method, as 11 devices is considerably smaller than 53 devices from the previous scenario. Also, those 11 devices can be present on the network only during the attack and, therefore, they will more likely stay unnoticed by network administrators.

VI. THREATS TO VALIDITY

Experimental verification presented in this article is based on an experiment with one selected network and a simulation of various situations that can occur during network operation. This can be considered as a threat to validity. Although the network used in the experiment was sufficiently extensive, it cannot be assumed that other large networks will have a similar topology and characteristics.

However, this issue can be mitigated by adjustment of parameters of the proposed methods. In networks where devices do not fluctuate as much as they do in university networks or in networks where there is a high number of newcomers or irregularities, the values for the threshold, the optimal benchmark size or the measurement times may vary significantly.

Another concern may be raised regarding the fact that in the proposed method and in the experiments, we used MAC addresses as a device identifier. Generally, MAC addresses are easy to spoof, and if attackers determine the set of recurring devices, they can spoof them in the network, which would lead to false positive result.

To mitigate this issue, alternative device identification can be used. With an alternative identification of a device, principle of the method does not change.

VII. DISCUSSION

The *threshold* given by the Algorithm IV-B can be further adjusted to modify behavior of the method. Lowering the threshold decreases the amount of the false positive, while it may increase the number of false negative. Increasing the threshold has the opposite effect. Each percent we remove from the threshold determines percentage of devices that are allowed to fail without false negative. For instance, this could provide safety margin, while decreasing the accuracy of the method.

The *number of benchmark* days determines the adaptability to network changes. Networks with a higher number of fluctuating devices will have a lower value than networks where the same devices are present all the time. Those values can be modified to suit the particular network.

Time frames definition affects the behavior characteristics of the method. Basically the longer the time frame, the more devices fluctuate. While this can offer some extra protection against MAC spoofing, it decreases the threshold and, therefore, can lead to false positives.

The proposed method is dependent on the size of the network. At least tens of overall devices are needed to provide meaningful results and hundreds to achieve a consistent output.

Our approach provides an *additional authentication factor*, and, therefore, it would not be sufficient as a standalone authentication method. Also, the method does not detect changes in the behavior of the devices itself, but in its network neighborhood. Therefore, the proposed method does not detect device hijacking.

VIII. Conclusion

In the recent decade, IoT solutions have reached the point where they are being regularly deployed in the real world. However, the security of IoT devices (and, thus, of such solutions as a whole) still present numerous challenges that have to be addressed.

To date, various articles have described using existing security solutions in the IoT world or have proposed specific methods specifically tailored to the IoT. This article focused not on a standalone security method; rather, it proposed an additional authentication factor as a supplement to existing security solutions. It made decisions based on changes in the context in the network around devices, and therefore, it can detect suspicious or even malicious behavior. It is a simple mechanism in terms of device resources, and it can be deployed on every IoT device capable of communication over TCP/IP, allowing system operators to inspect the network and, if needed, to take appropriate actions to resolve an issue.

Performed experiments demonstrated the feasibility of our approach in a real-world network with a significant number of devices. The results indicated that our concept can provide valid results and increase the security of both the devices and the entire network. This sort of approach is especially fitting for secure locations, such as laboratories, energy sources or military bases, where the aim is to limit outside devices. However, this method might not be the best for locations where devices have a high rate of churn, such as shopping centers.

In future work, we seek to conduct more detailed and extensive testing of the method in various networks under different scenarios and observe whether it can detect security threats, e.g., monitoring when an external device (a potential threat) appears in the network. To evolve the proposed method further, in the future research, we are going to investigate options of general algorithm for threshold value determination using machine learning algorithms. We will focus on performance because a full network scan slows down the network if the scan is run from multiple devices. We seek to explore the possibilities of running the network scan for the given network from a single device and reusing it for others. We also seek to further refine our algorithm for detecting suspicious behavior and to apply various machine learning strategies.

References

[1] G. Wu, S. Talwar, K. Johnsson, N. Himayat, and K. D. Johnson, "M2M: From mobile to embedded Internet," *IEEE Commun. Mag.*, vol. 49, no. 4, pp. 36–43, Apr. 2011.

[2] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128610001568

[3] B. Lheureux, W. R. Schulte, A. Velosa, "Hype cycle for the Internet of Things," Gartner Inc., Tech. Rep., Jul. 2017. [Online]. Available: https://www.gartner.com/doc/3770369/hype-cycle-internet-things-

[4] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed Internet of Things," *Comput. Netw.*, vol. 57, no. 10, pp. 2266–2279, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128613000054

[5] K. Zhao and L. Ge, "A survey on the Internet of Things 3security," in *Proc. 9th Int. Conf. Comput. Intell. Secur.*, 2013, pp. 663–667. [Online]. Available: https://doi.org/10.1109/CIS.2013.145

[6] S. Li, L. D. Xu, and S. Zhao, "The Internet of Things: A survey," *Inf. Syst. Frontiers*, vol. 17, no. 2, pp. 243–259, Apr. 2015.

[7] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in *Handheld and Ubiquitous Computing*, H.-W. Gellersen, Ed. Berlin, Berlin, Germany: Springer, 1999, pp. 304–307.

[8] M. J. Covington, W. Long, S. Srinivasan, A. K. Dev, M. Ahamad, and G. D. Abowd, "Securing context-aware applications using environment roles," in *Proc. 6th ACM Symp. Access Control Models Technol.*, 2001, pp. 10–20. [Online]. Available: http://doi.acm.org/10.1145/373256.373258

[9] R. Bhatti, E. Bertino, and A. Ghafoor, "A trust-based context-aware access control model for web-services," in *Proc. IEEE Int. Conf. Web Serv.*, 2004, pp. 184–. [Online]. Available: https://doi.org/10.1109/ICWS.2004.15

[10] R. J. Hulsebosch, A. H. Salden, M. S. Bargh, P. W. G. Ebben, and J. Reitsma, "Context sensitive access control," in *Proc. 10th ACM Symp. Access Control Models Technol.*, 2005, pp. 111–119. [Online]. Available: http://doi.acm.org/10.1145/1063979.1064000

[11] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the Internet of Things: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 414–454, First Quarter 2014.

[12] J. Al-Muhtadi, A. Ranganathan, R. Campbell, and M. D. Mickunas, "Cerberus: A context-aware security scheme for smart spaces," in *Proc. 1st IEEE Int. Conf. Pervasive Comput. Commun.*, 2003, pp. 489–. [Online]. Available: http://dl.acm.org/citation.cfm?id=826025.826359

[13] J. Hightower and G. Borriello, "Location systems for ubiquitous computing," *Computer*, vol. 34, no. 8, pp. 57–66, Aug. 2001. [Online]. Available: http://dx.doi.org/10.1109/2.940014

[14] P. Baumann, W. Kleiminger, and S. Santini, "The influence of temporal and spatial features on the performance of next-place prediction algorithms," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2013, pp. 449–458. [Online]. Available: http://doi.acm.org/10.1145/2493432.2493467

[15] T. M. T. Do and D. Gatica-Perez, "Where and what: Using smartphones to predict next locations and applications in daily life," *Pervasive Mobile Comput.*, vol. 12, no. Supplement C, pp. 79–91, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1574119213000576

[16] I. Agadakos, P. Hallgren, D. Damopoulos, A. Sabelfeld, and G. Portokalidis, "Location-enhanced authentication using the IoT: Because you cannot be in two places at once," in *Proc. 32nd Annu. Conf. Comput. Secur. Appl.*, 2016, pp. 251–264. [Online]. Available: http://doi.acm.org/10.1145/2991079.2991090

[17] K. Nimmy, S. Sankaran, and K. Achuthan, "A novel multi-factor authentication protocol for smart home environments," in *Information Systems Security*, V. Ganapathy, T. Jaeger, and R. Shyamasundar, Eds. Cham, Switzerland: Springer, 2018, pp. 44–63.

[18] A. Kurniawan and M. Kyas, "A trust model-based Bayesian decision theory in large scale Internet of Things," in *Proc. IEEE 10th Int. Conf. Intell. Sensors, Sensor Netw. Inf. Process.*, Apr. 2015, pp. 1–5.

[19] M. Shahzad and M. P. Singh, "Continuous authentication and authorization for the Internet of Things," *IEEE Internet Comput.*, vol. 21, no. 2, pp. 86–90, Mar. 2017.

[20] T. Kumar, A. Braeken, M. Liyanage, and M. Ylianttila, "Identity privacy preserving biometric based authentication scheme for naked healthcare environment," in *Proc. IEEE Int. Conf. Commun.*, May 2017, pp. 1–7.

[21] Y. Wang, J. Liu, Y. Chen, M. Gruteser, J. Yang, and H. Liu, "E-eyes: Device-free location-oriented activity identification using fine-grained wifi signatures," in *Proc. 20th Annu. Int. Conf. Mobile Comput. Netw.*, 2014, pp. 617–628. [Online]. Available: http://doi.acm.org/10.1145/2639108.2639143

[22] F. Adib, Z. Kabelac, D. Katabi, and R. C. Miller, "3d tracking via body radio reflections," in *Proc. 11th USENIX Conf. Netw. Syst. Des. Implementation*, 2014, pp. 317–329. [Online]. Available: http://dl.acm.org/citation.cfm?id=2616448.2616478

[23] C. Shi, J. Liu, H. Liu, and Y. Chen, "Smart user authentication through actuation of daily activities leveraging wifi-enabled IoT," in *Proc. 18th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2017, pp. 5:1–5:10. [Online]. Available: http://doi.acm.org/10.1145/3084041.3084061
[24] Y. Zeng, P. H. Pathak, and P. Mohapatra, "Wiwho: Wifi-based person identification in smart spaces," in *Proc. IEEE 15th Int. Conf. Inf. Process. Sensor Netw.*, 2016, pp. 4:1–4:12. [Online]. Available: http://dl.acm.org/citation.cfm?id=2959355.2959359
[25] F. Hong, X. Wang, Y. Yang, Y. Zong, Y. Zhang, and Z. Guo, "WFID: Passive device-free human identification using wifi signal," in *Proc. 13th Int. Conf. Mobile Ubiquitous Syst.: Comput., Netw. Serv.*, 2016, pp. 47–56. [Online]. Available: http://doi.acm.org/10.1145/2994374.2994377
[26] M. Trnka, F. Rysavy, T. Cerny, and N. Stickney, "Using wi-fi enabled Internet of Things devices for context-aware authentication," in *Information Science and Applications 2018*, K. J. Kim and N. Baek, Eds. Singapore: Springer Singapore, 2019, pp. 635–642.
[27] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: Methods, systems and tools," *IEEE Commun. Surveys Tut.*, vol. 16, no. 1, pp. 303–336, First Quarter 2014.
[28] S. Raza, L. Wallgren, and T. Voigt, "Svelte: Real-time intrusion detection in the Internet of Things," *Ad Hoc Netw.*, vol. 11, no. 8, pp. 2661–2674, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870513001005
[29] J. Hui and P. Thubert, "Compression format for IPv6 datagrams over IEEE 802.15.4-based networks," Internet Requests for Comments, RFC Editor, RFC 6282, Sep. 2011, http://www.rfc-editor.org/rfc/rfc6282.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6282.txt
[30] E. Hodo *et al.*, "Threat analysis of IoT networks using artificial neural network intrusion detection system," in *Proc. Int. Symp. Netw., Comput. Commun.*, May 2016, pp. 1–6.
[31] R. L. Lawrence and A. Wright, "Rule-based classification systems using classification and regression tree (cart) analysis," *Photogrammetric Eng. Remote Sens.*, vol. 67, no. 10, pp. 1137–1142, 2001.
[32] K. Nozaki, H. Ishibuchi, and H. Tanaka, "Adaptive fuzzy rule-based classification systems," *IEEE Trans. Fuzzy Syst.*, vol. 4, no. 3, pp. 238–250, Aug. 1996.
[33] "Information technology – open systems interconnection – basic reference model: Naming and addressing," International Organization for Standardization and the International Electrotechnical Commission, Geneva, Switzerland, ISO/EIC 7498-1:1997, 1997.
[34] L. Oliveira, J. Rodrigues, S. Kozlov, R. Rabêlo, and V. Albuquerque, "Mac layer protocols for Internet of Things: A survey," *Future Internet*, vol. 11, no. 1, Jan. 2019, Art. no. 16. [Online]. Available: http://dx.doi.org/10.3390/fi11010016

Michal Trnka received the master's degree in computer science from the Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic, in 2015, where he is currently working toward the Ph.D. degree in computer science - software engineering..

He is also a Fulbright Scholar. His research interests include software engineering, especially in context aware applications security. Lately, he focused on using context information to enhance security rules as well as methods to obtain context from trusted sources, like Internet of Things devices.

Jan Svacina received the bachelor's degree in software engineering from the Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic, in 2018.

He is currently a Research Assistant with the Department of Computer Science, Baylor University, Waco, TX, USA. His research interests include security in Internet of Things using event-driven programming and program code analysis for quality assurance and testing.

Tomas Cerny received the master's degree in computer science and engineering from the Faculty of Electrical Engineering, Czech Technical University, Prague, Czech Republic, in 2009, the master's degree in computer science from Baylor University, Waco, TX, USA, in 2009, and the Ph.D. degree in information science and computer engineering from the Faculty of Electrical Engineering, Czech Technical University, in 2016.

In 2017, he was a Postdoc with Baylor University. He is currently a Professor of Computer Science with Baylor University. His research interests include software engineering, security, aspect-oriented programming, code-analysis, user-interface engineering, and enterprise application design.

Eunjee Song received the B.S. degree in computer engineering and the other B.S. degree in architecture from Seoul National University, Seoul, South Korea, in 1991 and 1988, respectively and the M.S. and Ph.D. degrees in computer science from Colorado State University, Fort Collins, CO, USA, 2001 and 2007, respectively.

She is an Associate Professor & Graduate Program Director with the Department of Computer Science, Baylor University, Waco, TX, USA. Her research interests include the field of software engineering, with a focus on pattern specification, software analysis and testing, and applying aspect-oriented modeling and model-driven engineering techniques to specifying and analyzing complex systems. Prior to her graduate study, she was with IBM Korea as a Software Engineer for more than five years.

Dr. Song is a Member of the ACM Special Interest Group on Applied Computing (SIGAPP).

Jiman Hong received the B.S. degree in computer science from Korea University, Seoul, South Korea, in 1991, and the M.E. and Ph.D. degrees in computer science and engineering from Seoul National University, Seoul, South Korea, in 1997 and 2003, respectively.

Since 2007, he has been with the School of Computer Science and Engineering, Soongsil University, Seoul, South Korea, where he is currently a Full Professor. He was a Chief of Technical Officer with the R&D Center of GmanTech Incorporated Company, Seoul, South Korea between March 2000 and December 2003. His research interests include operating systems, embedded systems, and Internet of Things systems.

Dr. Hong has served as technical and organizational committees, program chair, poster chair, workshop chair of more than 40 IEEE, and Association for Computing Machinery international conferences. He is currently serving as the Chair of ACM Special Interest Group on Applied Computing (SIGAPP).

Miroslav Bures received the Ph.D. degree in computer science with the Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic, in 2006.

He is an Associate Professor in Software Testing and Quality Assurance with Faculty of Electrical Engineering, Czech Technical University in Prague. His research interests include model-based testing (process and workflow testing, data consistency testing, and combinatorial interaction testing), effective test automation (test automation architectures, assessment of automated testability, and economic aspects), and quality assurance methods for Internet of Things solutions, reflecting specifics of this technology. In these areas, he also leads several R&D and experimental projects.

Dr. Bures is a Member of Czech chapter of the ACM, czech and slovak testing board (CaSTB), international software testing qualifications board (ISTQB) Academia work group and participates in broad activities in professional testing community.