An Article Title That Spans Multiple Lines to Show Line Wrapping

John Smith[1,2], Robert Smith[3] and Jane Smith[1*]

[1]School of Chemistry, The University of Michigan
[2]Physics Department, The University of Wisconsin
[3]Biological Sciences Department, The University of Minnesota

Abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent porttitor arcu luctus, imperdiet urna iaculis, mattis eros. Pellentesque iaculis odio vel nisl ullamcorper, nec faucibus ipsum molestie. Sed dictum nisl non aliquet porttitor. Etiam vulputate arcu dignissim, finibus sem et, viverra nisl. Aenean luctus congue massa, ut laoreet metus ornare in. Nunc fermentum nisi imperdiet lectus tincidunt vestibulum at ac elit. Nulla mattis nisl eu malesuada suscipit. Aliquam arcu turpis, ultrices sed luctus ac, vehicula id metus. Morbi eu feugiat velit, et tempus augue. Proin ac mattis tortor. Donec tincidunt, ante rhoncus luctus semper, arcu lorem lobortis justo, nec convallis ante quam quis lectus. Aenean tincidunt sodales massa, et hendrerit tellus mattis ac. Sed non pretium nibh. Donec cursus maximus luctus. Vivamus lobortis eros et massa porta porttitor.

1 Introduction

The Internet of Things (IoT) refers to the network of physical objects, or "things", embedded with sensors, software, and other technologies, enabling them to collect and exchange data with other devices and systems over the internet. The applications of IoT are diverse and span various domains, such as healthcare, manufacturing, transportation, and home automation. IoT technology has gained significant popularity in recent years due to its ability to enhance automation, efficiency, and data-driven decision-making.

However, the widespread use of Internet of Things (IoT) devices presents new challenges in the realm of cybersecurity. As a class of devices focused more on usability than security, they are generally more vulnerable to attacks. Combined with their large population, IoT devices create a broad attack surface, opening up new attack paths once they are breached. Due to the sheer quantity of IoT devices, attacks initiated from them can easily overload the bandwidth of a network and even the physical capacity of a system.[**10.1007/978-3-319-68711-7_18**] From an economic perspective, IoT device manufacturers and users do not bear the brunt of these vulnerabilities since IoT devices are generally not the target of attacks, but rather serve as stepping stones for them. This property reduces the incentive to implement stronger cybersecurity defenses.

Certain types of malware have already infiltrated IoT devices, causing serious impacts on Internet infrastructure. For example, Mirai, discovered in May 2016, is an IoT malware notorious for executing powerful distributed denial-of-service (DDoS) attacks. It infects IoT devices like IP cameras and home routers by exploiting factory default or hard-coded usernames and passwords. Post-infection, these devices transform into a botnet for launching DDoS attacks.[**203628**] Other notable IoT malware, such as Reaper, Hajime, and Bashlite, have also emerged, each exhibiting unique infection strategies and posing distinct challenges to the security of IoT devices.[**7971869**]

One of the significant challenges of malware detection for IoT devices is their limited memory and processing power. Due to these limitations, performing all calculations locally can be costly; thus, cloud-based services become a logical choice. However, relying on cloud-based services can lead to latency and privacy issues, as sensitive data may be transmitted outside the network. Additionally, the heterogeneity of IoT devices, referring to the wide variety of devices with different architectures, operating systems, and communication protocols, makes it difficult to develop a one-size-fits-all solution for malware detection.

Two primary solutions currently exist[**ogsimbiota**][**cloudeyes**], each representing one extreme of the spectrum: performing all calculations locally or entirely on the cloud. However, both solutions have their own unique set of problems. Local computation, while preserving privacy and reducing latency, can be quite resource-intensive given the limited memory and processing power of IoT devices. On the other hand, cloud-based computation offers more substantial computational

*Corresponding author: jane@smith.com
Received: October 20, 2023, **Published:** December 14, 2023

power and scalability but introduces latency and potential privacy issues, as it requires transmitting sensitive data outside the network. Hence, neither of these extremes offers a perfect solution. There likely exists an optimal trade-off that lies somewhere between these two extremes, a hybrid approach that combines the benefits of local and cloud-based computations while mitigating their respective drawbacks. This balance is precisely what our proposed solution seeks to achieve.

In this paper, we propose a solution aiming to address the significant challenges of malware detection for IoT devices. A similarity-based hybrid solution which uses both cloud and local computation, striving to strike an optimal trade-off between the computational cost of performing all calculations locally and the potential latency and privacy issues associated with relying on cloud-based services. By leveraging the strengths of both cloud and local computation, we believe that this solution can effectively detect malware while minimizing the resources required to do so.

2 SIMBIoTA

In this section, we will provide a concise overview of the operation principle of SIMBIoTA and challenges it faces. From which we will explain that there is a correlation between memory requirement and computational time, as well as how to develop upon this model to better utilize the Internet connectivity. We will then make insightful observations and present measurement results demonstrating the memory and computational time demands of the original SIMBIoTA.

cohesive?

SIMBIoTA is a malware detection algorithm that heavily relies on a specialized hash algorithm called TLSH. This similarity-based algorithm provides a fast and simple way of calculating similarity between two different samples. Service providers maintain a constantly updated knowledge base of previously known malware samples, without any binary executables, only the TLSH hash value. This hash value is used to determine the similarity between samples and targets.

However, saving the entire database on the local device is not realistic due to its large size and accumulation over time. The authors have provided a solution to this problem by strategically selecting a subset of the database. This subset is sufficiently large to identify malware, eliminating the need for a complete database on the local device.

2.1 TLSH

TLSH is a locality sensitive hash[**6754635**] which can generate a hash value from binary file to be used to compare the similarity between two files. Authors of this paper also provided an algorithm to calculate distance score given two TLSH hash value. With distance 0 representing identical files and higher score representing lower similarity.

2.2 Operation

The detailed operation will be discussed for service provider and IoT devices separately.

Service providers In the SIMBIoTA algorithm, the service provider starts by constructing a similarity graph from the TLSH values of all known malware samples. A threshold T is set by the service provider. The vertices of the graph represent the malware samples, and an edge is connected between two vertices if their respective TLSH values have a distance score lower than T. Once the graph is constructed, the service provider calculates the dominating set of the graph, which is a set of vertices that covers all vertices in the graph such that each vertex in the set has at least one neighbor that is not in the set. This subset of vertices is sufficient to identify malware, so it is pushed to the IoT devices to use in identifying malware. This approach reduces the computational and memory resources required by the IoT devices while still providing good detection rate.

Algorithm 1 ConstructSimilarityGraph

```
function ConstructSimilarityGraph(MalwareSamples)
    graph ← EmptyGraph()
    for node1 in (MalwareSamples) do
        for node2 in (MalwareSamples) do
            similarityScore ← diff(node1, node2)
            if similarityScore < T then
                graph.AddEdge(node1, node2)
            end if
        end for
    end for
    return graph
end function
```

IoT devices When a device encounters a suspicious file, its TLSH hash value is calculated. This value is then compared with all malware samples in the dominating set. If the similarity between the file and one of the known malware is high, it will be identified as malware.

2.3 Observations

It can be understood from graph theory. For a finite set of vertices V where

$$|V| = v \tag{1}$$

The number of edges e in the similarity graph $G = (V, E)$ change as the threshold T changes. We can observe the two extreme

$$\begin{cases} G = V, & \text{if } T = 0 \\ G = K_v, & \text{if } T = \infty \end{cases} \tag{2}$$

A higher threshold results in more edges in the similarity graph, which then result in a smaller dominating set, since vertices generally have more adjacent vertices. In other words, threshold and dominating set size is inverse proportional.

A threshold $T = 40$ is used in original SIMBIoTA as the authors proved this value provided high clustering coefficient.**[9914145]** not enough

2.4 Challenges with SIMBIoTA

Although SIMBIoTA offers a high detection rate while maintaining moderate memory consumption, it still faces certain challenges. One such challenge is the considerable memory requirement of the malware database, which grows in size as new malware discoveries are made. Furthermore, the database relies on regular updates from the vendor, which may result in outdated information depending on the update frequency.

To tackle these issues, we present our new proposal: SIMBIoTA-cloud. This innovative solution aims to address the aforementioned problems.

3 SIMBIoTA-cloud

We proposed somethingDon't know what to write here.

3.1 Operation

We first establish two variables:

$$\text{Upper threshold } T_U \tag{3}$$
$$\text{Lower threshold } T_L \tag{4}$$

The operation of SIMBIOTA-Cloud is explained with regards to both the IoT devices and the cloud-based detection system.

IoT devices In SIMBIOTA-Cloud, the process begins similarly to the original SIMBIOTA algorithm, but with the addition of an upper and lower threshold, where the upper threshold is greater than or equal to the lower. The service provider constructs a similarity graph using the upper threshold from the TLSH values of all known malware samples. Following this, the dominating set of the graph is calculated, as with the original SIMBIOTA algorithm. This set is then used by IoT devices for local malware detection.

When an IoT device encounters a suspicious file, it calculates the file's TLSH hash value and then compares this with all malware samples in the dominating set. If the distance score between the file and a known malware is less than the lower threshold, the file is identified as malware. Conversely, if this value is higher than the upper threshold, the file is identified as benign. However, if the distance score falls between the two thresholds, the TLSH hash is sent to the cloud for further analysis.

Cloud-based detection system The cloud-based part of SIMBIOTA-Cloud provides an additional layer of detection for files that fall within the upper and lower thresholds, as determined by the IoT devices. When the cloud system receives a TLSH hash, it has the capability to use various detection methods due to its significantly higher computational power.

For instance, it can perform a comparison to the entire malware sample database, or it could utilize machine learning algorithms. In our measurements, we will use the full database for comparison. If the minimum distance score between the received TLSH hash and any malware sample in the database is lower than the lower threshold, the cloud system reports back to the IoT device, identifying the file as malware. If no such match is found, the file is reported as benign. This cloud-based approach enables more detailed analysis of potential malware, thus increasing the overall detection rate without significantly burdenishing the IoT devices.