HoneyThing: A New Honeypot Design for CPE Devices

Ömer Erdem[1*], Abdurrahman Pektaş[2] and Mehmet Kara[3]
[1] Istanbul Commerce University, Computer Eng., Kucukyali E5 Kavsagi Inonu Cad. No: 4, Kucukyali 34840, Istanbul, Turkey
[e-mail: omer.erdem@istanbulticaret.edu.tr]
[2] Galatasaray University, Computer Eng. Dept., Çırağan Cad. No:36, Ortaköy, Istanbul, Turkey
[e-mail: apektas@yandex.com]
[3] Sistem Bilgisayar Ltd., Rafer Karacan Cad. No: 24, Izmit, Kocaeli Turkey
[e-mail: mehmet29@hotmail.com]
*Corresponding author: Ömer Erdem

Received November 7, 2016; revised March 3, 2017; accepted April 10, 2018; published September 30, 2018

Abstract

The Internet of Things (IoT) has become an emerging industry that is broadly used in many fields from industrial and agricultural manufacturing to home automation and hospitality industry. Because of the sheer number of connected devices transmitting valuable data, the IoT infrastructures have become a main target for cyber-criminals. One of the key challenges in protecting IoT devices is the lack of security measures by design. Although there are many hardware and software based security solutions (firewalls, honeypots, IPDS, anti-virus etc.) for information systems, most of these solutions cannot be applied to IoT devices because of the fact that IoT devices have limited computing resources (CPU, RAM,). In this paper, we propose a honeypot system called HoneyThing for modem/router devices (*i.e.* a kind of IoT device). HoneyThing emulates TR-069 protocol which is prevalent protocol used to remotely manage customer-premises equipment (CPE) devices, e.g. modems, routers. Honeything also serves an embedded web server simulating a few actual, critical vulnerabilities associated with the implementation of TR-069 protocol. To show effectiveness of the HoneyThing in capturing real world attacks, we have deployed it in the Internet. The obtained results are highly promising and facilitate to reveal network attacks targeting to CPE devices.

Keywords: Computer security, internet of things, CPE, TR-069 protocol, honeypot

1. Introduction

The expanding scale and ubiquity of the Internet is increasing continually since the early days of the Internet. Especially, with the introduction of the IoT concept, almost all devices such as refrigerators, cars, houses, etc., can connect and communicate with each other in our daily life without human intervention. It is estimated that the number of devices connected to the Internet is expected to reach 25 billion device by 2020 [1]. The main reason of this growth is not human population; rather the fact that the advances of operational technologies employed in industries. For instance, devices and equipments being manufactured with ability to connect the Internet (e.g., health devices, fans, lights, sensors) are one of the key reason in exponential increase of the Internet.

The expansion of IoT in people's social life causes this area to become a remarkable market. Therefore, companies and organizations make much more investment. At this age of interconnected things, the humans are interacting with the machines and machines are talking with each other (M2M). However, communications of different types of IoT devices creates new security issues. Moreover, the lack of enough security measures in IoT and proliferation of the IoT attract cyber-criminals to target IoT infrastructure. Last but not least, even though the security patches for IoT devices are produced and published by vendors, generally the security risks related to these issues are not closed. Because these patches are not applied automatically by vendors and end-users don’t have know-how to apply them to IoT devices.

By taking advantage of these security problems and vulnerabilities in IoT, hackers can cause big damages for people and companies in terms of money, reputation and time. The aim of the attacks on IoT devices can vary from basic information disclosure to remote code execution (arbitrary code execution) and total system compromise. For instance, in 2016, cyber-criminals comprised a large number of Internet-enabled devices such as surveillance cameras, digital video recorders, home routers (e.g. CPE) and other embedded devices by exploiting factory default username and password. The attackers then utilized compromised IoT devices to create a botnet, known as Mirai, to launch large-scale DDoS attacks [2].

Network security technologies such as firewall, IDS/IPS, VPN, NAC, anti-virus are very prevalent for information systems. Basically, these solutions provide security while carrying out business processes. However, the same security solutions can't be applied on IoT devices due to the fact that they haven’t got enough computing resources such as CPU power, memory, bandwidth or storage to run security mechanism [3]. Furthermore, today's strong encryption and authentication algorithms are based on cryptographic suites such as AES, RSA and Diffie-Hellman. Although these algorithms are robust, they also need high CPU power and RAM resources. Consequently, authentication and authorization will require appropriate re-engineering to accommodate for IoT world. In order to solve these issues, there are many research projects to develop new authentication and encryption algorithms.

Honeypots are hardware or software based solutions used to detect attacks on information systems. Essentially, they are designed to detect hacking attempts, network scans and malware such as worms and viruses in the networks they are deployed in. Besides that, they can log any interaction between attacker and honeypot. Due to the fact that they are not deployed on targeted system as an antivirus or a host based intrusion detection system, they are not influenced by the lack of hardware resources as IoT devices that we mentioned above. In this regard, honeypot can be employed to capture network attacks for IoT. Though, there are various honeypot applications developed by security communities such as Honeyd, Dionaea,

Kippo, Conpot, Glastopf, Thug. To the best of our knowledge, there is not any comprehensive honeypot for devices managed by TR-069 (e.g. CPE devices).

In this work, we developed a honeypot named HoneyThing to monitor and detect attacks on CPE devices. New IoT protocols and applications could be addable to HoneyThing. HoneyThing emulates the following services:

- TR-069 protocol which is used for remote management of CPE devices,
- RomPager 4.07 embedded web server,
- Modem/router web management application and
- Three important vulnerabilities related to RomPager.

1.1 Our Contributions

Our contributions are as follows:

- Implementing a honeypot system for CPE devices to track cyber-attacks and share it with security community on Github [4].
- Developing a Bro IDS script for detecting and analyzing network attacks targeting ADSL modem router running TR069 protocol.
- Pointing out the latest attacks targeting ADSL router by deploying a honeypot in the Internet.

The remainder of the paper is structured as follows. Section 2 includes literature review for the Internet of Things, honeypots and TR-069 protocol headings. Section 3 explains HoneyThing application basic features as well as design and development phases in detail. Then, in Section 4, we evaluate the developed honeypot. Finally, we provide the conclusion and possible future work in Section 5.

2. Literature Review

2.1 Internet of Things

The concept of “IoT” was first introduced by the Auto-ID research center at Massachusetts Institute of Technology (MIT) in 1999. Then, “IoT” term was officially confirmed in the World Summit on the Information Society (WSIS) hold by the International Telecommunications Union (ITU) in Tunisia in 2005 [5]. Nevertheless, some ambiguity exists in the literature in defining IoT; however, it can be best described as a network that control, monitor smart devices and sensors such as smart-phones, networked entertainment devices, auto-mobiles, smart meters, wearable and portable health devices, industrial controllers, power management devices, RFID tags and sensors [6]. Main properties of this network are to collect, process, analyze and save the data generated by the “things”'.

The Internet of Things (IoT) is transforming itself gradually from small closed networks to a complex network of decentralized smart devices. This transformation leads to new services and easy management opportunities. IoT network is growing rapidly which gives us flexibility to connect different types of devices from cars to machines to home appliances. This process affects our daily life; transforming smart devices to smart factories.

With the IoT technology, it is possible to control different information systems from a power plant to a small sensor. Performance and security requirements can change from one application to another. On one hand, some of them has a very limited RAM, CPU and other resources. On the other hand, people demand secure, easy to use, fail safe, robust systems. Thus, there are great challenges for developers in securing IoT.

Cyber threats are real, global and imposing threats to industrial IoT. When IoT devices transfer a great volume of sensitive data, the security risks and attacks can be substantially increased. The interested readers can look at [7] for top challenging areas of IoT.

2.2 Honeypots

In computer security, honeypots are decoy computer system deployed in a network for the purpose of pro-actively gathering information about network scan, intrusion and malicious activity. Since honeypots use unannounced (non-routable) IP addresses, all network traffic they receive are considered as suspected. Because of this reason, the honeypots extensively record and monitor via capturing all network traffic and logging all activities [8]. Furthermore, as honeypots operate in an isolated fashion from their networks, their compromises by attackers do not generate any security risks to the organizations.

Honeypot is used to collect attack records and malware samples by emulating network services, applications, operating systems or network devices. All these components include specific vulnerabilities and misconfiguration issues. While hackers and malicious software try to employ these defects, honeypots collect the attackers' techniques to exploit specific vulnerabilities.

Honeypots are classified into three groups according to their level of interaction between attacker and honeypot: low, medium and high [9]. Low-interaction honeypots emulates a network service, an application, operating system service or whole sub-net. For example, it can emulate the network services such as an SMTP server, IIS and Apache to attract attackers' attention. As they offer imitated services, low interaction honeypots are relatively limited to capture details about the attack. The Honeyd application is a good example of a low-interaction honeypot.

Mid-interaction honeypots do not include a real operating system like low interaction honeypots. However, they provide richer interaction capabilities for attackers. Nepenthes is a well-known example of a mid-interaction honeypots. One of the main drawbacks of such systems is that if vulnerabilities (e.g buffer overflow, etc.) exist within the honeypot, the attackers can compromise the main operating system.

High-interaction honeypots offer network services and applications on real operating systems that also include real vulnerabilities. From this point of view, they give a great opportunity to researchers to analyze attacks in-depth. Even high-interaction honeypots facilitate detailed analysis, their set-up and management are usually complex. Besides, the clean-up procedure after infection is difficult to maintain and needs an automation mechanism. Sebek, Argos, Windows XP without SPs or another unpatched operating system are examples of high interaction honeypot. The best choice of honeypot for a given network depends on different factors. Relationship for these factors and honeypot types are listed in **Table 1**.

There are many honeypots actively used to notify network attacks. Most of them are open source and their source code are hosted on github.com such as Honeyd, Dionaea, Kippo, Conpot, Glastopf and Thug. Some of the well-known honeypots and their primary features are as follows:

- Honeyd, Dionaea and Kippo emulate SMB, HTTP, FTP, TFTP, MSSQL, SSH protocols.
- Conpot emulates industrial control system protocols. Jicha et al. conduct a set of real world experiments to determine treats pertaining to SCADA devices by using Conpot [10].

Table 1. Classification of Honeypots, [11]

Factors	Low Interaction	Medium Interaction	High Interaction
Infection Probability	Low	Medium	High
Real Operating System	No	No	Yes
Setup Easy	Difficult	Very	Difficult
Maintenance	Easy	Easy	Medium
Risk	Low	Medium	High
Hacking Expectation	No	No	Yes
Control Expectation	No	No	Yes
Knowhow for Setup	Low	Low	High
Knowhow for Development	Low	High	Medium-High
Data Collection	Limited	Medium	Detailed
Interaction	Service Emulation	Depend on Expectation	Full control
Real Operating System	No	No	Yes

- Glastopf is a low-interaction honeypot and emulates web application vulnerabilities. Glastopf collects web-based attacks such as SQL injection, directory traversal or HTML injection [12].
- Thug honeypot is used for client side attacks. Thug aims to mimicking the behavior of a web browser to detect and dissect malicious contents. It is capable of performing static and dynamic code analysis [13].
- SIPHON is a high-interaction honeypot for IoT devices. In [14], the authors designed a network architecture to monitor and collect IoT attacks by establishing secure VPN tunnels between the deployed forwarding machine and the physical devices.
- Nepenthes is a low interaction honeypot and emulates network vulnerabilities that worms exploit to spread. The main purpose of the Nepenthes is to collect worm malware [15].
- IoTPOT is a frontend low-interaction honeypot and emulates telnet services of various IoT devices [16]**.** IoTPOT solely examines telnet-based attacks against IoT devices.

The interested readers can refer to an extensive overview on honeypots as well as techniques to analyze honeypot attacks in A Survey on Honeypot Software and Data Analysis. In [17], Piggin et al. present a methodology to ensure active defense against attacks on Operational Technology (OT) or Industrial Control System (ICS) by using honeypot technology. Although extensive research has been carried out on honeypots systems, there is not a comprehensive honeypot for CPE devices. Thus, this paper primarily focuses on developing base honeypot system for CPE devices.

2.3 TR-069 Protocol

Technical Report 069 (TR-069) is a technical specification that describes CPE WAN Management Protocol, designed for remote management of end-user devices such as modem, router, VoIP phone and network based storage. It is a SOAP/HTTP-based (Simple Object Access Protocol/Hypertext Transfer Protocol) application layer protocol [18]. Thus, its main goal is to handle the communication between Customer-Premises Equipment (CPE) and Auto Configuration Server (ACS). Indeed, it is an integrated framework maintaining auto configuration and controlling the other CPE functions. The latest version of TR-069 was published in 2013.

Practically, ISPs use TR-069 protocol to manage the TR-069 CPEs. When CPE is turned on, it tries to connect ACS operated by ISPs. These servers run a software that implements and runs TR-069 protocol. Configuring customer devices, monitoring them for faults, running

diagnostics and even silently upgrading their firmware are main tasks of TR-069 [19]. The protocol enables bidirectional communication by using SOAP/HTTP protocols. Messages are first prepared by SOAP with Extensible Markup Language (XML) format and then transferred by Remote Procedure Call (RPC). An example of these communications is shown in **Fig. 1**.

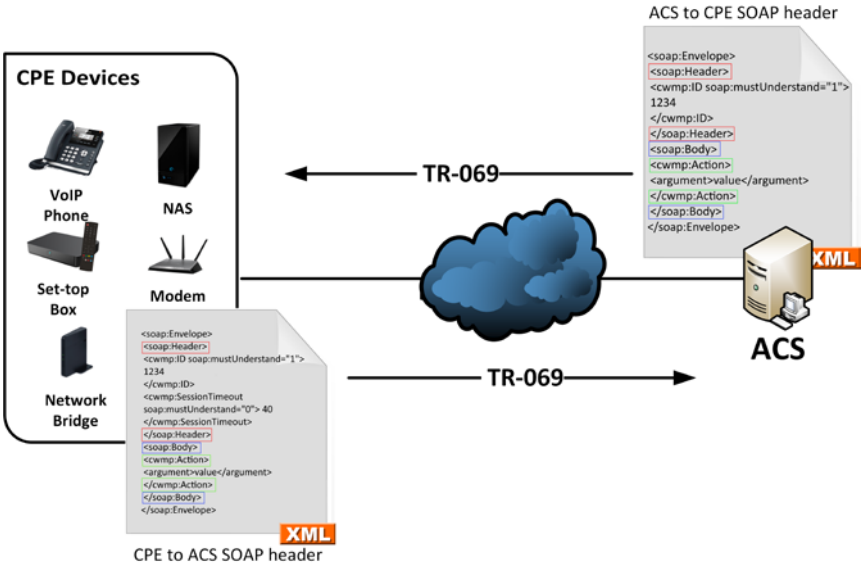


Fig. 1. Bidirectional TR-069 communication between ACS and CPE

Probably, many users are not aware that their ISPs have strict control over their routers. Some providers hide option to disable TR-069 protocol on the web management interface of the modems and routers. Most interestingly, although this option is disabled manually on some modems offering to disable, TR-069 still continues to run at background and the device still has an open port to wide area network. Thus, if TR-069 enabled device has vulnerabilities and the firmware is not updated, disabling TR-069 option from web interface will not be the proper solution for this scenario.

3. HoneyThing

The main goal of HoneyThing is to provide a complete, easy-to-use low-interaction honeypot application that supports TR-069 protocol and logs all interaction between attacker and itself in detail. HoneyThing includes a few recently discovered vulnerabilities which is related to the modem/routers and being actively used by hackers in the wild. The developed honeypot application consists of two main components: CPE implementation of TR-069 protocol and web management application of modems. The schematic diagram of the HoneyThing topology is shown in **Fig. 2**.

Firstly, we integrated an open source CPE implementation of TR-069 protocol into HoneyThing for the aim of understanding attackers' behaviours, determining the volume of attacks targeting TR-069 protocol and identifying zero day attacks. Then, we implemented a web application that emulates a few vulnerabilities and acts as a modem web management interface. By this way, we can obtain detailed information about attackers' activities and some statistical information like username/password attempts, the most visited URLs etc.

The HoneyThing is written in Python programming language. Thus, it is portable, i.e. it can be installed and run easily on any operating system possessing a Python runtime environment. Besides, it also provides flexibility by editable configuration parameters before run. In development process, many modem/routers such as ZyXEL, TP-LINK, AirTies, D-Link, Belkin are tested to perceive the nature of TR-069 protocol and the response of the chosen vulnerabilities when triggered.

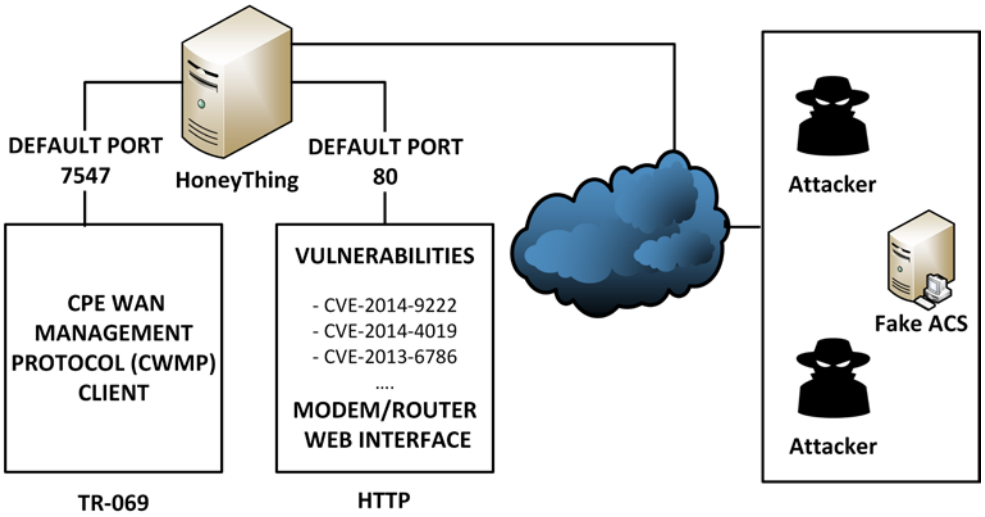


Fig. 2. HoneyThing topology

3.1 Basic Features

To develop the first component of the application, we analyzed lots of vulnerabilities from different brands/models and then decided to use 3 of them that have huge number of potential victims and therefore, a large impact potential. The selected vulnerabilities affect devices running RomPager embedded web server. According to the Checkpoint malware and vulnerability research team's report published in 2014, RomPager was constituted 52% of the servers that run TR-069 protocol. Among these servers, the 4.07 version of RomPager, released in 2002, has a distribution of 98% and it is being used by approximately 12 million devices from 50 different brands all over the world [20]. The May 2016 scan of the default TCP/7547 port by Censys team revealed that the vulnerable version of the RomPager possess still higher usage rate [21]. According to this scanning result, there are approximately 67 million IPs actively serving on TCP port 7547 and the server information about 26 million of these IPs had been captured. The version of 4.07 web server comprises 32.4% of servers. Details about scanning results can be seen in **Fig. 3**. Nevertheless, RomPager 4.07 still has the highest proportion among the embedded web servers of the modems/routers.

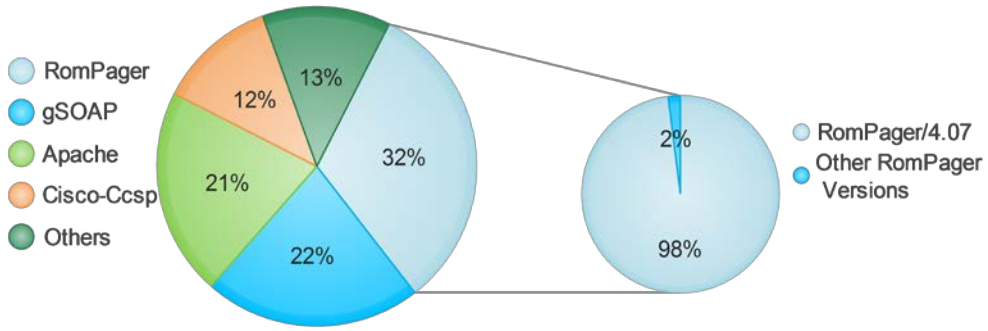


Fig. 3. Types of IoT web servers and distribution of RomPager embedded web server

The first emulated vulnerability is CVE-2014-9222 (also known as Misfortune Cookie) discovered by Check Point's researchers in December 2014 [22]. The attacker can command the fortune of a request (i.e. how the request will be processed) by manipulating HTTP cookies. The misfortune cookie can be triggered by sending a single specifically crafted packet to the affected versions of RomPager. Then, this packet would poison the device's memory and allows the attacker to get administrative control on the modem's web management interface. However, sometimes, the crafted HTTP request may cause the failure of the device. After gaining admin rights over management interface of the device, attackers can manipulate user internet traffic by changing the DNS settings, access the other devices connected to modem/router with port mapping, capture private and sensitive user data and so on.

ROM-0 (coded as CVE-2014-4019 in the Common Vulnerabilities and Exposures database) is the other emulated vulnerability for RomPager published in early January 2014. This vulnerability allows attackers to download the router's configuration (backup) file by issuing a single HTTP request to unauthenticated URI (`/rom-0'). The backup file contains sensitive information such as ISP, Wi-Fi and management interface passwords and other sensitive information.

CVE-2013-6786 is the third emulated vulnerability on HoneyThing. This security flaw allows attackers to make URL redirection or XSS attack by sending a request to a non-existing URL on device with special parameters. As it is well known, the URL redirection attack is a good way to redirect users to a malicious site instead of the original web site.

These three vulnerabilities are chosen and implemented by taking into consideration of the number of possible victims. To increase the level of interaction and plausibility judgment, the first component of HoneyThing offers a modem web management interface with an authentication mechanism. Additionally, a new vulnerability module can be added to the system without requiring additional effort.

To develop the second component of the application, lots of open source CPE implementation of TR-069 protocol such as Freecwmp, Mini CWMP, Catawampus and EasyCWMP were analyzed. Other than implementing TR-069 protocol from scratch, we incorporated an open source TR-069 implementation to HoneyThing. The open-source implementation of the protocol minimize the probability of the identification of the honeypot system when the attacker tries to explore all details of TR-069 or sends specific crafted requests. Catawampus is chosen due to the fact that it is written in Python and open source. Some changes were made on Catawampus in order to liken to honeypot as follows:

- **Changing the authentication method:** The required ACS credentials are not taken into account on the HoneyThing. In other words, attackers can connect to HoneyThing (a CPE device) without any username/password.

- **Storing file activities:** Normally, Catawampus does not allow saving the downloaded files by ACS. As, it is important to analyze the transferred files, we incorporated a “file save” feature into HoneyThing.
- **Preventing honeypot identification:** In this context, we modified some parameters that can result in detection of HoneyThing.

HoneyThing has ability to log details of the attacks. It supports three types of log file: "**http.log**" keeps a record of every HTTP connection related to web management interface, "**cwmp.log**" holds all TR-069 communication events (for instance session initialization, configuration update, information gathering, etc.), and **"honeything.log"** contains status and error messages used for debugging of HoneyThing. The self-descriptive fields for each log file generated by HoneyThing are shown in **Fig. 4**.

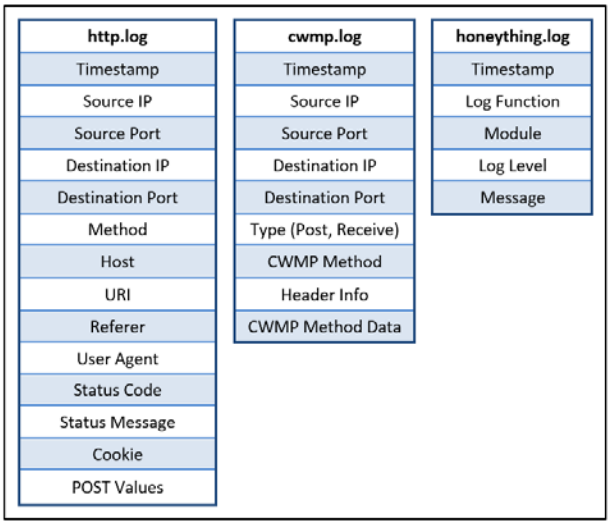


Fig. 4. HoneyThing log details

3.2 Usage Scenarios

HoneyThing can be used both in research or production environment. In both of two cases, especially in a production environment, as HoneyThing is a low interaction honeypot, it is recommended to use it in an isolated network. Because, once an attacker compromise Honeything honeypot, he/she can download an additional malware via the TR-069 "Download" command, scan other systems on the network or spread on the network . Besides that, all security updates on host operating system where HoneyThing is hosted should be done to lower the impact of such a risk. HoneyThing can be deployed either in virtual machines or on physical servers with the same configuration.

HoneyThing listens at TCP port 7547 as a default port for TR-069 protocol. However, the user can change the port for a different setup. This port adjustment will increase the usability and efficiency acquired from HoneyThing. Moreover, some statistical information can be obtained about wrong password attempts by activating the password complexity in HoneyThing's configuration file for modem web management interface.

The usage scenarios of HoneyThing can be extended and adapted to the user's specific needs. For example, if multiple instance of HoneyThing is running, all log files can be sent to a central log server. Then, due to the fact that they are parseable, logs can be easily visualized by using some open source tools to track attacks, connections and the state of honeypot.

Furthermore, the downloaded files can be sent automatically to online/offline sandboxes and a multi-antivirus scanning system like VirusTotal through a simple script.

3.3 Bro IDS Script for TR-069 Protocol

Apart from developing honeypot system for CPE devices, we also created a Bro-IDS script to inspect TR-069 communication. The Bro Network Security is an open-source powerful network analysis framework. Bro is different from the typical signature-based intrusion detection systems since it doesn't block the attacks but it enables extensive close monitoring of all network traffic.

Bro-IDS supports well-known network protocols, extracts related information from network packets and exposes network activities at high-level [23]. By default, Bro stores all network events into tab-separated parseable log files suitable for post-processing and visualization with external tools. The main advantage of Bro is that it offers highly powerful even-driven scripting language for specifying users' arbitrary analysis tasks.

The developed Bro script inspects the network traffic for TR-069 protocol and logs related protocol fields. More specifically, the script scans all extracted files from HTTP protocol which have XML file magic. Then, it checks the possible matching of given regular expression. If it finds a match in XML document, it extracts and logs the RPC methods, user agent, transport layer information (e.g., IPs and ports), http method, uri, fields etc. to a log file named tr-069.log. The pseudo-code algorithm of the Bro-IDS script for TR-069 protocol is given in **Fig. 5**. The developed script can be found in [24].

Algorithm 1: The pseudo-code algorithm of the Bro-IDS script for TR-069 protocol

```
create file log stream for logging TR-069 attacks
rpc_commands = [GetRPCMethods, Download, Reboot, ...]
foreach file_sniff event do
    if protocol == HTTP and mime_type == "application/soap+xml" then
        http_body == HTTP data exacted while downloading the file
        rpc_pattern = /<SOAP-ENV:Body>(\x0A|\x0D)*<cwmp:[A-Za-z0-9]+/
        if rpc_pattern in http_body and http_body contains rpc_commands then
            write rpc method, user agent, host, http method, uri, fields etc. to log a file
        end
    else
        continue
    end
end
```

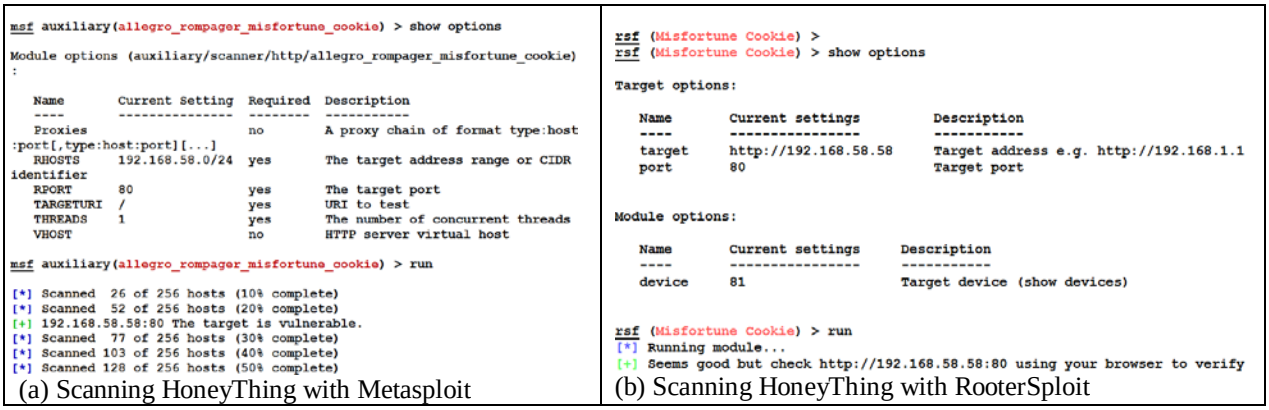
Fig. 5. The pseudo-code algorithm of the Bro-IDS script for TR-069 protocol

4. Evaluation

4.1 Evaluation in Test Environment

The functional test of HoneyThing is run in a local area network. Before testing HoneyThing in a WAN, we examined some open source ACS software such as LibreACS, GenieACS, PerlCWMP, TR069 D-Link wih a real ADSL modem to choose the most appropriate one. Based on our experiments, we selected LibreACS and wrote scripts for TR-069 CPE methods which aren't supported by default in LibreACS such as "Download", "GetParameterValues", "SetParameterValues", etc. Our test environment includes 4 PCs: HoneyThing installed on Linux OS, Kali Linux distribution (specialized penetration testing platform) to exploit vulnerabilities of the HoneyThing, LibreACS on Linux to serve TR-069 protocol functions and a testing machine with a web browser.

In these experiments, we firstly aim at finding out whether the vulnerabilities hosted by HoneyThing can be exploited by hacking tools or not. Secondly, we attempt to check if our TR-069 implementation protocol is running properly. ROM-0 vulnerability is tested by downloading modem configuration from unauthenticated URL and getting web management password with "RouterPassView" tool. The misfortune cookie vulnerability is evaluated with open source hacking tools; Metasploit's "allegro_rompager_misfortune_cookie" module on Kali Linux distribution and RouterSploit (Router Exploitation Framework). When the network/device is scanned/exploited with these tools, HoneyThing returns "Vulnerable"/"Exploited" response. An example scanning result is shown in **Fig. 6**.



(a) Scanning HoneyThing with Metasploit (b) Scanning HoneyThing with RooterSploit

Fig. 6. Misfortune cookie vulnerability test with open source hacking tools

The assessment of the implementation and functionality of TR-069 protocol is done with LibreACS machine. Before beginning the evaluation, the connection URL for ACS is configured on HoneyThing in order to connect to LibreACS service. By the way, this URL options is disabled in default configuration, which means that HoneyThing accepts all ACS connections from Internet. When both CPE (HoneyThing) and ACS (LibreACS) are run, we observed that emulated TR-069 protocol works as intended. Interested readers can check the following activities.

- CPE information about the HoneyThing located in configuration file under cpe section can be sent to LibreACS.
- TR-069 CPE methods, for example, "Download" method can download a file from given URL regardless of its file type.
- HoneyThing can reply the requests sent from LibreACS to its connection request URL regardless of username/password pair.
- While running HoneyThing, all interactions are recorded to log files.

4.2 Evaluation in Real-World Deployment

To evaluate HoneyThing under the real network attacks, we deployed it on the Internet. For this purpose, firstly we installed and configured HoneyThing on Ubuntu Server 14.04 LTS with Intel(R) Core(TM) i5-2410M@2.30 GHz processor and 2 GB of RAM. As HoneyThing emulates the modem/router services, we deployed it into 3G network infrastructure. Accordingly, the R206 3G wifi router is used and configured to forward network traffic coming to its public static IP address to the internal server in which HoneyThing is hosted.

After one week of experiment, HoneyThing captured current attack patterns. Based on the results, Honeything is probed from 21 unique IP addresses from 8 countries. Among these

network attacks, some attacker IPs just scan network for identifying running network services. However, the attacks highlighted red in **Table 2** attempted to exploit the Honeything by logging in HoneyThing's web management interface with default username and password (i.e. admin:admin) and then trying to change the DNS to 5.152.208.2-6. We ensured that these IPs, which are attempted to set as DNS, are actually hosting DNS service through making DNS requests using dig utility. The attackers probably aimed at redirecting the users into the fake website to steal their credentials or the malicious website to get access to their computer by taking advantages of vulnerabilities in their web browser. Another significant idea behind these attacks might be to join CPE devices into a botnet, like Mirai botnet.

In summary, the main advantage of the Honeything against other honeypots is that to the best of our knowledge it is the only honeypot system designed for CPE devices. The experimental results in test environment and in the real world scenario show that it can efficiently and effectively capture the attacks targeting CPE devices. The main drawbacks of the Honeything is that currently it can only simulates existing CPE vulnerabilities. However, as long as new vulnerabilities are found, these vulnerabilities can be added to the Honeything.

Table 2. Distribution of the attacks targeting HoneyThing

Attacker Country	Number of Attack
United Kingdom	390
United Kingdom	390
United Kingdom	330
United Kingdom	300
United Kingdom	270
United Kingdom	240
United Kingdom	210
United Kingdom	210
Netherlands	184
Turkey	22
Netherlands	16
Mexico	2
Argentina	2
Argentina	2
Argentina	2
Argentina	1
Argentina	1
Argentina	1
United Arab Emirates	1
India	1
United States	1

5. Conclusion

The IoT concept gains more and more importance and is already partly integrated in our daily lives. As a consequence, there have been significant increase in attacks targeting IoT devices. Besides that, it is difficult to realize or detect an intrusion attempt because end users do not usually interact with a modem/router as much as they interact with their laptops and mobile phones. Honeypots are one the most convenient and effective method for detecting intrusion and it can work on any operating system or network topology.

With this work, we have implemented a honeypot; called HoneyThing for the CPE devices which is widely in use in the Internet. HoneyThing includes Misfortune Cookie, ROM-0 and CVE-2013-6786 vulnerabilities currently seen in many modem/rooter devices and supports TR-069 protocol. These emulated vulnerabilities are selected by considering the number of potential victims. Besides the basic functionalities, the application is designed with usability in mind. After testing HoneyThing with pentest tools, we can conclude that HoneyThing supports all features of the emulated modems.

There are some possible improvements and future works for HoneyThing. New vulnerability modules and telnet service support can be added to HoneyThing. HoneyThing can provide the attacker to get a shell access after successful exploit. Some important shell commands can be emulated by HoneyThing in post-exploitation phase. Logs can be send to syslog or database for easy processing and also Hpfeeds from Honeynet organization can be supported. Lastly, HoneyThing can be developed to support lots of IoT device such as fridge, air conditioner and iron with a modular structure.

Acknowledgements

The authors gratefully acknowledge support of TUBITAK-BILGEM as the primary investigator of the advanced information security projects in Turkey. TUBITAK-BILGEM is founded by The Scientific and Technological Research Council of Turkey (TUBITAK) as the leading national agency for management, funding and conducting research in Turkey. This work is also supported by the Turkish Honeynet Chapter.

References

[1] "Gartner says 4.9 billion connected things will be in use in 2015,", 2014, last accessed on 4 July 2016 URL: Article (CrossRef Link).

[2] Angrishi Kishore, "Turning Internet of Things (IoT) into Internet of Vulnerabilities (IoV): IoT Botnets," *in arXiv preprint arXiv*:1702.03681, 2017.

[3] Raja Benabdessalem, Mohamed Hamdi and Tai-Hoon Kim, "A Survey on Security Models, Techniques, and Tools for the Internet of Things," in *Proc. of 7th IEEE Conference on Advanced Software Engineering and Its Applications (ASEA)*, pp. 44-48, 2014. Article (CrossRef Link).

[4] Ömer Erdem, Mehmet Kara and Abdurrahman Pektaş, "Honeything: A Honeypot for Internet of TR-069 Things,", last accessed on March 2017. URL: Article (CrossRef Link).

[5] Handong Zhang and Lin Zhu, "Internet of things: Key technology, architecture and challenging problems," in *Proc. of IEEE Conference on Computer Science and Automation Engineering (CSAE)*, vol. 4, pp. 507-512, June 2011. Article (CrossRef Link).

[6] Luigi Atzori, Antonio Iera and Giacomo Morabito, "The internet of things: A survey," *Computer networks*, vol.54, no. 15, pp. 2787-2805, 2010. Article (CrossRef Link).

[7] Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini and Imrich Chlamtac, "Internet of things: Vision, applications and research challenges," *Ad Hoc Network*, vol. 10, no. 7, pp. 1497-1516, 2012. Article (CrossRef Link).

[8] Iyatiti Mokube and Michele Adams, "Honeypots: concepts, approaches, and challenges," in *Proc. of Proceedings of the 45th Annual Southeast Regional Conference*, pp. 321-326, 2007. Article (CrossRef Link).

[9] Spitzner Lance, "Honeypots: tracking hackers," *AddisonWesley Reading*, 2003.

[10] Arthur Jicha, Mark Patton and Hsinchun Chen,"SCADA honeypots: An in-depth analysis of Conpot," in *Proc. of Intelligence and Security Informatics (ISI)*, pp. 196-198, IEEE, 2016. Article (CrossRef Link).

[11] RC Joshi and Anjali Sardana, "Honeypots: A New Paradigm to Information Security," *CRC Press*, 2011.
[12] Lukas Rist, "Glastopf: Web Application Honeypot," last accessed on March 2017 URL: Article (CrossRef Link).
[13] Angelo Dell'Aera, "Thug honeypot," last accessed on March 2017 URL: Article (CrossRef Link).
[14] Juan Guarnizo, Amit Tambe, Suman Sankar Bhunia, Martín Ochoa, Nils Tippenhauer, Asaf Shabtai and Yuval Elovici, "SIPHON: Towards Scalable High-Interaction Physical Honeypots," *in arXiv preprint arXiv:1701.02446*, 2017. Article (CrossRef Link).
[15] Paul Baecher, Markus Koetter, Thorsten Holz, Maximillian Dornseif and Felix Freiling, "The nepenthes platform: An efficient approach to collect malware," in *Proc. of International Workshop on Recent Advances in Intrusion Detection*, pp. 165-184, 2006, Springer. Article (CrossRef Link).
[16] Yin Minn Pa Pa, Shogo Suzuki, Katsunari Yoshioka, Tsutomu Matsumoto, Takahiro Kasama and Christian Rossow, "IoTPOT: Analysing the rise of IOT compromises," in *Proc. of 9th USENIX Workshop on Offensive Technologies (WOOT 15)*, 2015.
[17] Richard Piggin and Ian Buffey, "Active Defence Using an Operational Technology Honeypot," in *Proc. of* System Safety and Cyber-Security (SSCS 2016), pp. 6-15, 2016. Article (CrossRef Link)
[18] "TR-069 CPE WAN management protocol," 2013 URL: Article (CrossRef Link).
[19] Juan Pablo Martínez Rojas, "Split management of TR069 enabled CPE devices," *Master's thesis, The Regio Politecnico di Torino (Royal Turin Polytechnic)*, 2011.
[20] Shahar Tal and Lior Oppenheim, "The internet of TR-069 things: One exploit to rule them all," 2015, URL: Article (CrossRef Link).
[21] Zakir Durumeric, David Adrian, Ariana Mirian, Michael Bailey and J Alex Halderman, "A search engine backed by internet-wide scanning," in *Proc. of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 542-553, 2015. Article (CrossRef Link).
[22] "Misfortune Cookie: CVE-2014-9222,"2014, last accessed 4 July 2016. URL: Article (CrossRef Link)
[23] Bing Chen, Joohan Lee and Annie S Wu, "Active event correlation in bro IDS to detect multi-stage attacks," in *Proc. Fourth IEEE International Workshop on Information Assurance (IWIA)*, pp. 16, 2006. Article (CrossRef Link).
[24] Abdurrahman Pektaş and Ömer Erdem, "Bro Script for TR-069," 2016. Article (CrossRef Link)

Ömer Erdem is a researcher at TUBITAK BILGEM Cyber Security Institute. He received his B.Sc. degree in Computer Engineering from Istanbul Commerce University and M.Sc. degree in Cyber Security Engineering from Istanbul Sehir University. He is currently Ph.D candidate student in Computer Engineering at Istanbul Commerce University. His research interests are intrusion detection/prevention systems, honeypots, network security and forensics. He has also experienced in Unix/Linux systems, networking, Python/PHP programming and many open source tools. He is Turkish Chapter member of The Honeynet Project.

Abdurrahman Pektaş received his B.Sc. and M.Sc. at Galatasaray University and his PhD at the University of Joseph Fourier, all in computer engineering, in 2009, 2012 and 2015, respectively. He is a senior researcher at the Cyber Security Institute in The Scientific and Technological Research Council of Turkey. His research interests are analysis, detection and classification of malicious software, machine learning and security analysis tool development.

Mehmet Kara received his Ph.D. in Electronics and Communications Engineering from Kocaeli University. His research interests include most aspects of cyber security and in particular, the areas of secure network design, network security testing, intrusion detection/prevention, firewalls, and secure software development and malicious code analysis. Mehmet took part in the foundation of the Common Criteria laboratory in Turkey and he is a Common Criteria evaluator of this laboratory since 2003. Mehmet has served as program committee member and reviewer in various conferences. His research has been published in national and international journals and conferences.