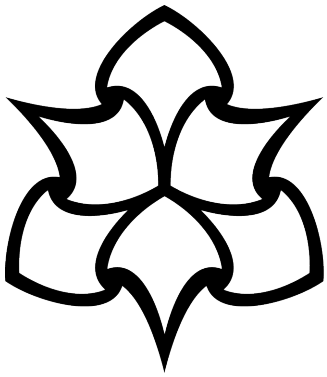

Malicious SSL Certificate Detection: A Step Towards Advanced Persistent Threat Defence

Ibrahim Ghafir[1,2], Vaclav Prenosil[1], Mohammad Hammoudeh[2], Liangxiu Han[2], Umar Raza[2]
[1] Faculty of Informatics, Masaryk University, Brno, Czech Republic
[2] Faculty of Science & Engineering, Manchester Metropolitan University, Manchester, UK
ghafir@mail.muni.cz,prenosil@fi.muni.cz,M.Hammoudeh@mmu.ac.uk,l.han@mmu.ac.uk,u.raza@mmu.ac.uk

ABSTRACT

Advanced Persistent Threat (APT) is one of the most serious types of cyber attacks, which is a new and more complex version of multi-step attack. Within the APT life cycle, continuous communication between infected hosts and Command and Control (C&C) servers is maintained to instruct and guide the compromised machines. These communications are usually protected by Secure Sockets Layer (SSL) encryption, making it difficult to identify if the traffic directed to sites is malicious. This paper presents a Malicious SSL certificate Detection (MSSLD) module, which aims at detecting the APT C&C communications based on a blacklist of malicious SSL certificates. This blacklist consists of two forms of SSL certificates, the *SHA1 fingerprints* and the *serial & subject*, that are associated with malware and malicious activities. In this detection module, the network traffic is processed and all secure connections are filtered. The SSL certificate of each secure connection is then matched with the SSL certificate blacklist. This module was experimentally evaluated and the results show successful detection of malicious SSL certificates.

CCS CONCEPTS

•**Security and privacy** → **Intrusion detection systems;** *Network security;* •**Networks** → Network monitoring;

KEYWORDS

Cyber attacks, malware, advanced persistent threat, malicious SSL certificate, intrusion detection system.

ACM Reference format:
Ibrahim Ghafir[1,2], Vaclav Prenosil[1], Mohammad Hammoudeh[2], Liangxiu Han[2], Umar Raza[2]. 2017. Malicious SSL Certificate Detection: A Step Towards Advanced Persistent Threat Defence. In *Proceedings of ICFNDS '17, Cambridge, United Kingdom, July 19-20, 2017,* 6 pages.
DOI: 10.1145/3102304.3102331

1 INTRODUCTION

Cyber attacks refer to intentional activities against software, hardware or data in computer networks or systems. These activities may degrade, disrupt, destroy, or deny access to legitimate users. Many intelligence agencies and governments' militaries are actively getting ready to launch or block cyber attacks, maybe in conjunction with traditional attacks or counter-attacks. Cyber exploitation is another expression, which refers to intelligence-gathering rather than devastating actions [17]. The goal is to get information and data without getting caught or being detected. As the number of ubiquitous devices grow [1, 2, 33–35, 42, 47, 50], the number of the security threats with implication for the general public increases. Unfortunately, new technologies, such as the Internet of Things, comes with new set of security threat that needs to be addressed.

Over the last decade, malware and botnets [28, 31] have increased to become a key reason of the majority of the (Distributed) Denial-of-Service (DOS) activities [38], direct attacks [4], spear phishing [22] and scanning [51], which takes place through the Internet. Botnets are networks formed by "enslaving" host computers, called bots (derived from the word robot), that are controlled by one or more attackers, called botmasters, with the intention of performing malicious activities [5]. In other words, bots are malicious codes running on host computers that allow botmasters to control the host computers remotely and make them perform various actions [15]. It has been noticed that there is a change in motivation, from curiosity and fame seeking/excitement-seeking to illegal financial gain, this has been marked by a rising sophistication in the evolution of malicious software [16]. Moreover, the availability of easy-to-use tool kits to build malware will probably keep malware a threat to consumers, businesses and governments in the foreseeable future. Furthermore, network monitoring approaches [52] have become more and more important in modern complicated networks.

One of the most serious types of cyber attacks is the Advanced Persistent Threat (APT) [20], which is targeting a specific organisation and it is performed through several steps. The main aim of APT is espionage and then data exfiltration. Therefore, APT is considered as a new and more complex version of multi-step attack [36]. These APTs form a problem for current detection methods [19] as they use advanced techniques like social engineering [27] and make use of unknown vulnerabilities. Moreover, the economic damages due to a successful APT attack can be very expensive. The expected cost of attacks is the major motivation for the investments in intrusion detection and prevention systems [46]. APTs are currently one of the most serious threats to the companies and governments [54].

A novel approach for APT detection is proposed in [26]. The suggested system undergoes two main phases, the first one detects eight techniques commonly used in APT life cycle. For that purpose, eight detection modules are presented, which are disguised exe file detection [18], malicious file hash detection [25], malicious domain name detection [24], malicious IP address detection [23], malicious

ICFNDS '17, Cambridge, United Kingdom

DOI: 10.1145/3102304.3102331

SSL certificate detection, domain flux detection [21], scan detection, and Tor connection detection [30]. The second phase includes a correlation framework to link the outputs of the detection modules.

Within the APT life cycle, continuous communication between the infected hosts and the C&C servers should be preserved to instruct and guide the compromised machines. These communications are usually protected by Secure Sockets Layer (SSL) encryption, making it difficult to identify if the traffic directed to sites is malicious. This paper presents the malicious SSL certificate detection (MSSLD) module, which aims at detecting the APT C&C communications based on a blacklist of malicious SSL certificates. This blacklist consists of two forms of SSL certificates, the *SHA1 fingerprints* and the *serial & subject*, that are associated with malware and malicious activities. In this detection module, the network traffic is processed and all secure connections are filtered. The SSL certificate of each secure connection is then matched with the SSL certificate blacklist.

The remainder of this paper is organized as follows. Section 2 presents the related work to APT detection. The malicious SSL certificate detection module and its algorithms are explained in Section 3. Section 4 shows the evaluation results and Section 5 concludes the paper.

2 RELATED WORK

A classification model for APT detection is presented in [14]. This model is built based on machine learning algorithms. First, the legitimate traffic for normal users is analysed aiming to extract the CPU usage, memory usage, open ports and number of files in the system32 folder. A piece of malware, which is previously used for the APT attack, is injected into the network and the four features are extracted. The dataset of benign and malicious features are used to train the detection model using different machine learning algorithms.

TerminAPTor, an APT detector, is described in [12]. This detector uses information flow tracking to find the links between the elementary attacks, which are triggered within the APT life cycle. TerminAPTor depends on an agent, which can be a standard intrusion detection system, to detect those elementary attacks. A statistical APT detector, similar to TerminAPTor detector, is developed in [48]. This system considers that APT undergoes five states which are delivery, exploit, installation, C&C and actions; and several activities are taken in each state. The generated events in each state are correlated in a statistical manner.

An APT detection system based on C&C domains detection is introduced in [53]. This work analyses the C&C communication based on the observation that the access to C&C domains is independent, while the access to legal domains is correlated.

An approach for APT detection based on spear phishing detection is explored in [13]. This approach depends on mathematical and computational analysis to filter spam emails. Tokens, which are considered as a group of words and characters such as (click here, free, Viagra, replica), should be defined for the detection algorithm to separate legitimate and spam emails.

An active-learning-based framework for malicious PDFs detection is suggested in [40]. These malicious PDFs can be used in the early steps of APT to get the point of entry.

An approach based on Data Leakage Prevention (DLP) is proposed in [49]. This approach focuses on detecting the last step of APT which is the data exfiltration. A DLP algorithm is used to process the data traffic to detect data leaks and generate "fingerprints" according to the features of the leak.

A context-based framework for APT detection is explained in [32]. This framework is based on modelling APT as an attack pyramid in which the top of the pyramid represents the attack goal, and the lateral planes indicates the environments involved in the APT life cycle.

An in-depth analysis of Duqu is presented in [7]. A European organisation was targeted by attackers using the Duqu malware to steal data. The authors propose the Duqu detector toolkit, which consists of six investigation tools developed to detect the Duqu malware involved in the APT attacks.

With regards to the processing of multiple streams of events, IBM suggests a conceptual model for event processing in [39]. It describes the basic requirements to design an efficient correlation system. The work presented in [10] is based on finite state machines and uses a query language for event processing. Both systems can process the events in real time, a key limitation shared by both approaches is that they can not detect so called low & slow attacks, which take place over an extended time period.

Finally, APT detection systems face serious shortcomings in achieving real time detection [6], detecting all APT attack steps [6], balance between false positive and false negative rates [7], and correlating of events spanning over a long period of time [10, 39]. To address those weaknesses, a new approach for APT detection has been presented in [26], and this paper is a step towards developing the proposed system.

3 MALICIOUS SSL CERTIFICATE DETECTION (MSSLD)

APT C&C communications are usually protected by Secure Sockets Layer (SSL) encryption, which makes it difficult to identify malicious traffic. MSSLD aims at detecting C&C communications based on a blacklist of malicious SSL certificates [3, 37]. This blacklist consists of two forms of SSL certificates, the *SHA1 fingerprints* and the *serial & subject*, which are associated with malware and malicious activities. As shown in Figure 1, the network traffic is processed and all secure connections are filtered. The SSL certificate of each secure connection is then matched with the SSL certificate blacklist.

The MSSLD module is implemented on top of the *Bro* [29, 43] passive, open-source network traffic analyser. It is primarily a security monitor, which inspects all traffic on a link in depth for signs of suspicious activity. The most immediate benefit gained from deploying Bro is an extensive set of log files, which record a network's activity in high-level terms. These logs include not only a comprehensive record of every connection seen on the wire, but also application-layer transcripts such as, e.g., all HTTP sessions with their requested URIs, key headers, MIME types, and server responses; DNS requests with replies; and much more. Bro event engine reduces the incoming packet stream into a series of higher-level *events*, more than 300 events. These events reflect network activity in policy-neutral terms, i.e., they describe *what* has been

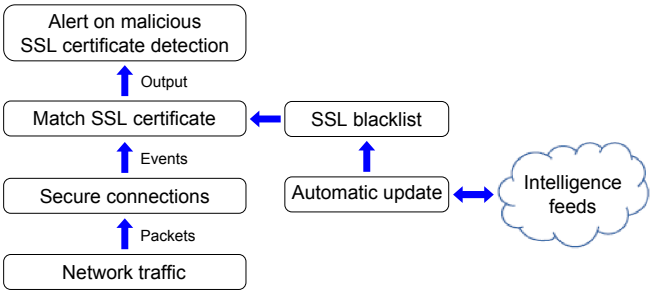

Figure 1: Methodology of the malicious SSL certificate detection.

seen, but not *why*, or whether it is significant. The MSSLD module consumes and handles some of Bro events, explained later in Section 3.1; all connection information (such as *timestamp, src_ip, src_port, dest_ip, dest_port*) can be extracted from those events.

This detection module runs through two algorithms, mentioned further in Section 3.1, the first one is intelligence-based MSSLD and the second algorithm is event-based MSSLD. The intelligence-based MSSLD makes use of the *Bro Intelligence Framework* [11]. This framework enables attack detection modules to consume data from different data sources and make it available for matching.

MSSLD is a blacklist-based detection module in which the malicious SSL certificates blacklists are automatically updated based on different intelligence feeds at once. The automatic update runs parallel with the module process and there is no need to stop or restart MSSLD. This parallel-running feature allows a continuous live monitoring of the network traffic and supports real time detection. Based on the MSSLD algorithm, two automatic update mechanisms have been applied. Figure 2 shows the automatic update of the blacklists used by intelligence-based MSSLD. The user *crontab file* is configured to run *blacklist_update.sh* each day at 3:00 am, this shell script connects through the Internet to the data source servers and downloads updated blacklists of malicious SSL certificate hashes into a new *blacklist.intel* text file. This text file is connected to the *Intelligence Framework*, which consumes it as described in Section 3.1.

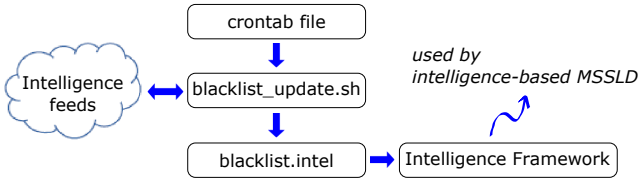

Figure 2: Automatic update of the blacklist used by intelligence-based MSSLD.

Figure 3 shows the automatic update of the blacklists used by the event-based MSSLD. The updated blacklist is downloaded into *ssl_blacklist.txt* text file. The *Input Framework* [44], built in Bro, enables MSSLD to use that text file as an input. The Input Framework reads *ssl_blacklist.txt* file into *bad_ssl* group, which is used by event-based MSSLD.

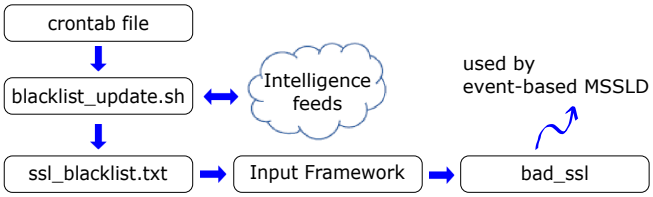

Figure 3: Automatic update of the blacklist used by event-based MSSLD.

As an output of the MSSLD module, in case of a malicious SSL certificate is detected, a corresponding event (ssl_alert) is generated. This event is to be used for alert correlation [8]. Additionally, an alert email is sent to the Request Tracker (RT) [9], where the network security team can perform additional forensics and respond to the triggered alert. It is assumed that the network security team responds to the generated alert within 24 hours, therefore, MSSLD suppresses all the same alerts, the same alert is the one which has the same infected host and the same malicious SSL certificate, into one alert per day, so no repeated alert emails sent the network security team. Moreover, this alert suppression reduces the computational cost of the alert correlation framework. To this end, after an alert is generated, this module adds the triggered alert into a specific table, the *t_suppress_ssl_alert*, where it stays for one day to ensure that the module does not generate the same alert within the next 24 hours. When a malicious SSL certificate is detected, and before an alert is generated, MSSLD checks the *t_suppress_ssl_alert* table in order to conclude if the same alert has been generated during the previous day, if so, the alert is ignored. Along with generating a new alert, information regarding the alert and the malicious connection (*alert_type, timestamp, src_ip, src_port, dest_ip, dest_port, infected_host, malicious_ssl*) is written into a specific log, i.e., *blacklist_detection_ssl.log*, to keep a historical record of the monitored network.

3.1 MSSLD Algorithms

As the blacklist consists of two forms of malicious SSL certificates (*SHA1 fingerprints* and *serial & subject*), two methods are followed for malicious SSL certificate detection. The first one is intelligence-based MSSLD, shown in Algorithm 1, and the second method is event-based MSSLD, shown in Algorithm 2.

In the intelligence-based MSSLD, the *Bro Intelligence Framework* is used and configured to monitor all secure connections SSL certificates' hashes. This framework is connected to the *blacklist.intel* file, which contains the SSL certificate blacklist. After extracting all secure connections traffic, SSL certificates hashes are passed to the intelligence framework to be checked against the intelligence data set *blacklist.intel*. When a match with any *indicator_type* of the intelligence data is found, the intelligence framework generates an *Intel::match* event. Through this event, if the *indicator_type* is *CERT_HASH*, it means this connection has

Algorithm 1 Implementation pseudo-code of intelligence-based MSSLD

```
Get malicious SSL certificates hashes blacklist (blacklist.intel)
Filter secure connections traffic
Extract SSL certificate hash
Send SSL certificate hash to Bro Intelligence Framework
if SSL certificate hash is in blacklist.intel then
    if the connection source IP belongs to the monitored network then
        if the same ssl_alert had not been generated over the last day then
            Generate an event (ssl_alert)
            Write ssl_alert into blacklist_detection_ssl.log
            Send an alert email to RT
            Suppress the same ssl_alert over the next day
        end if
    else if the connection destination IP belongs to the monitored network then
        if the same ssl_alert had not been generated over the last day then
            Generate an event (ssl_alert)
            Write ssl_alert into blacklist_detection_ssl.log
            Send an alert email to RT
            Suppress the same ssl_alert over the next day
        end if
    else
        goto End
    end if
else
    goto End
end if
End
```

Algorithm 2 Implementation pseudo-code of event-based MSSLD

```
Get malicious SSL certificates [serials and subjects] (bad_ssl group)
Filter secure connections traffic
Get x509_certificate event
Extract SSL certificate [serial and subject]
if SSL certificate [serial and subject] is in bad_ssl then
    if the connection source IP belongs to the monitored network then
        if the same ssl_alert had not been generated over the last day then
            Generate an event (ssl_alert)
            Write ssl_alert into blacklist_detection_ssl.log
            Send an alert email to RT
            Suppress the same ssl_alert over the next day
        end if
    else if the connection destination IP belongs to the monitored network then
        if the same ssl_alert had not been generated over the last day then
            Generate an event (ssl_alert)
            Write ssl_alert into blacklist_detection_ssl.log
            Send an alert email to RT
            Suppress the same ssl_alert over the next day
        end if
    else
        goto End
    end if
else
    goto End
end if
End
```

a malicious SSL certificate. Next, both connection sides, source and destination IP addresses, are checked through the *is_local_addr* function to check if the connection is established to or from the monitored network. To avoid raising the same alert within the same day, the *t1_suppress_ssl_alert* table is checked to ensure that it does not contain the same detected [host IP address, SSL certificate hash] set.

MSSLD then generates *ssl_alert* event, writes the malicious connection information into a specific log *blacklist_detection_ssl.log*, sends an alert email regarding the malicious SSL certificate detection to RT and adds the current detected set [host IP address, SSL certificate hash] into *t1_suppress_ssl_alert* table. The written information into *blacklist_detection_ssl.log* is:

```
timestamp = s$conn$start_time
alert_type = "ssl_alert"
connection  = s$conn$id
infected_host = s$conn$id$orig_h
malicious_ssl = s\$indicator
```

In the event-based MSSLD, the network traffic is processed and filtered into secure connections traffic, and then *x509_certificate* event can be generated for encountered X509 certificates [45]. Through this event, the serial and subject of the X509 certificate are checked for the certificate presence in the *bad_ssl* group. This group contains many of serials and subjects of malicious X509 certificates. If a match is found, the module should determine if the connection is established to or from one of the monitored network hosts; accordingly, both the source and destination IP addresses are checked through *is_local_addr* function. Before an *ssl_alert* is raised, the *t2_suppress_ssl_alert* table is to be checked to ensure that the same alert was not raised previously during the same day.

As in the previous intelligence-based method, MSSLD generates an *ssl_alert* event, writes the malicious connection information into a specific log, i.e., the *blacklist_detection_ssl.log*, sends an alert email regarding the malicious SSL certificate detection to RT and adds the current detected set [host IP address, SSL certificate hash] into *t2_suppress_ssl_alert* table.

4 EVALUATION RESULTS

To evaluate the MSSLD module, a virtual Internet-connected network was built, malware samples were injected into the virtual network, the network traffic was recorded into pcap files, and then the MSSLD module was applied on those pcap files.

As illustrated in Figure 4, two Windows virtual machines were connected to a physical consumer-grade router, which provided

connection to the Internet. The virtual machines behaved as physical computers in a home network and were able to communicate with each other. The Virtual machines traffic was recorded into two separate pcap files using the nictrace VirtualBox functionality [41]. Because no software besides the operating system and the malware was installed on the virtual machines and the operating system updates were disabled, the majority of the captured traffic was initiated by the installed malware. Moreover, it becomes easy to establish the ground truth.

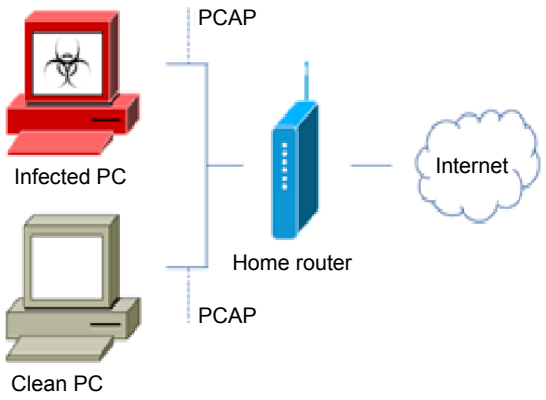

Figure 4: Topology of the implemented virtual network.

Two malware samples were injected independently into the virtual network for 5 minutes each. The first one is the *Trojan.Win32.Inject.sbqz*, also known as TorrentLocker, which has MD5 hash value of *aabe2844ee61e1f2969d7a96e1355a99*. The second injected malware sample is the *Trojan.Win32.Staser.bazr* malware, which has MD5 hash value of *e161a4d2716eb83552d3bd22ce5d603c*. The C&C servers for these two malware uses SSL certificates for communication over *https*. When the MSSLD module was applied on the captured pcap files, it successfully detected the malicious traffic as shown in Figure 5.

```
#fields  timestamp  alert_type  infected_host  malicious_ssl
#types  time  string  addr  string
138.857709  ssl_alert  192.168.1.101  45c0c2f1fa15b0ac5ce5fc018992a6ecf7e1e6bc
143.725647  ssl_alert  192.168.1.101  48a79b6bc3b9616f1e62fa4014997087673b358f
207.147152  ssl_alert  192.168.1.102  d8af2f6a1a2ba2b1b6e1a260e791fcab88cc2c8d
#close  2015-03-26-18-50-32
```

Figure 5: Part of a log produced by the MSSLD module.

5 CONCLUSION AND FUTURE WORK

This paper presents the malicious SSL certificate detection (MSSLD) module, which aims at detecting the APT C&C communications based on a blacklist of malicious SSL certificates. This blacklist consists of two forms of SSL certificates, the *SHA1 fingerprints* and the *serial & subject*, that are associated with malware and malicious activities. This module, processes the network traffic and filters all secure connections. The SSL certificate of each secure connection is then matched with the SSL certificate blacklist.

For future work, the output of this module will be correlated with the outputs of other detection modules, developed to detect commonly used techniques over the APT life cycle, to raise an alert on APT detection.

REFERENCES

[1] Abdelrahman Abuarqoub, Mohammad Hammoudeh, Bamidele Adebisi, Sohail Jabbar, Ahcène Bounceur, and Hashem Al-Bashar. 2017. Dynamic clustering and management of mobile wireless sensor networks. *Computer Networks* 117 (2017), 62–75.

[2] Abdelrahman Abuarqoub, Mohammad Hammoudeh, and Tariq Alsboui. 2012. An overview of information extraction from mobile wireless sensor networks. In *Internet of Things, Smart Spaces, and Next Generation Networking*. Springer Berlin Heidelberg, 95–106.

[3] Abuse.ch. 2017. SSL Blacklist a new weapon to fight malware and botnet. http://securityaffairs.co/wordpress/26672/cyber-crime/ssl-blacklist-new-weapon-fight-malware-botnet.html. (2017).

[4] Kostas G Anagnostakis, Stelios Sidiroglou, Periklis Akritidis, Konstantinos Xinidis, Evangelos P Markatos, and Angelos D Keromytis. 2005. Detecting Targeted Attacks Using Shadow Honeypots.. In *Usenix Security*.

[5] Paul Bacher, Thorsten Holz, Markus Kotter, and Georg Wicherski. 2005. Know your enemy: Tracking botnets. (2005).

[6] Marco Balduzzi, Vincenzo Ciangaglini, and Robert McArdle. 2013. Targeted Attacks Detection With SPuNge. (2013).

[7] Boldizsár Bencsáth, Gábor Pék, Levente Buttyán, and Márk Félegyházi. 2012. Duqu: Analysis, detection, and lessons learned. In *ACM European Workshop on System Security (EuroSec)*, Vol. 2012.

[8] Leau Yu Beng, Sureswaran Ramadass, and others. 2013. A comparative study of alert correlations for intrusion detection. In *Advanced Computer Science Applications and Technologies, 2013 International Conference on*. IEEE, 85–88.

[9] Best-Practical-Solutions. 2017. RT: Request Tracker. https://www.bestpractical.com/rt/. (2017).

[10] Lars Brenna, Alan Demers, Johannes Gehrke, Mingsheng Hong, Joel Ossher, Biswanath Panda, Mirek Riedewald, Mohit Thatte, and Walker White. 2007. Cayuga: a high-performance event processing engine. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. ACM, 1100–1102.

[11] Bro-Project. 2017. Intellegence Framework. https://www.bro.org/sphinx/frameworks/intel.html. (2017). Accessed: 15-02-2017.

[12] Guillaume Brogi and Valérie Viet Triem Tong. 2016. TerminAPTor: Highlighting Advanced Persistent Threats through Information Flow Tracking. In *New Technologies, Mobility and Security, IFIP International Conference on*. IEEE, 1–5.

[13] J Vijaya Chandra, Narasimham Challa, and Sai Kiran Pasupuleti. 2016. A practical approach to E-mail spam filters to protect data from advanced persistent threat. In *Circuit, Power and Computing Technologies (ICCPCT), 2016 International Conference on*. IEEE, 1–5.

[14] Saranya Chandran, P Hrudya, and Prabaharan Poornachandran. 2015. An efficient classification model for detecting advanced persistent threat. In *Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on*. IEEE, 2001–2009.

[15] Hyunsang Choi, Heejo Lee, and Hyogon Kim. 2009. BotGAD: detecting botnets by capturing group activities in network traffic. In *Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE*. ACM, 2.

[16] Kim-Kwang Raymond Choo. 2011. The cyber threat landscape: Challenges and future research directions. *Computers & Security* 30, 8 (2011), 719–731.

[17] Peter J Denning and Dorothy E Denning. 2010. Discussing cyber attack. *Commun. ACM* 53, 9 (2010), 29–31.

[18] Ibrahim Ghafir, Mohammad Hammoudeh, and Vaclav Prenosil. 2017. Disguised executable files in spear-phishing emails: Detecting the point of entry in advanced persistent threat. (2017). https://doi.org/10.7287/peerj.preprints.2998v1.

[19] Ibrahim Ghafir, Martin Husák, and Václav Přenosil. 2014. A Survey on Intrusion Detection and Prevention Systems. In *Proceedings of student conference Zvule, IEEE/UREL*. Brno University of Technology, 10–14.

[20] Ibrahim Ghafir and Vaclav Prenosil. 2014. Advanced Persistent Threat Attack Detection: An Overview. *International Journal of Advances in Computer Networks and Its Security (IJCNS)* vol. 4, Issue 4 (2014), 50–54.

[21] Ibrahim Ghafir and Vaclav Prenosil. 2014. DNS query failure and algorithmically generated domain-flux detection. In *International Conference on Frontiers of Communications, Networks and Applications (ICFCNA)*. IEEE Xplore Digital Library, 1–5.

[22] Ibrahim Ghafir and Vaclav Prenosil. 2015. Advanced Persistent Threat and Spear Phishing Emails. In *Proceedings of International Conference on Distance Learning, Simulation and Communication*. University of Defence, 34–41.

[23] Ibrahim Ghafir and Vaclav Prenosil. 2015. Blacklist-based malicious IP traffic detection. In *Global Conference on Communication Technologies (GCCT)*. IEEE Xplore Digital Library, 229–233.

[24] Ibrahim Ghafir and Vaclav Prenosil. 2015. DNS traffic analysis for malicious domains detection. In *2nd International Conference on Signal Processing and Integrated Networks (SPIN)*. IEEE Xplore Digital Library, 613–918.

[25] Ibrahim Ghafir and Vaclav Prenosil. 2016. Malicious file hash detection and drive-by download attacks. In *Proceedings of the Second International Conference on Computer and Communication Technologies*. Springer, 661–669.

[26] Ibrahim Ghafir and Vaclav Prenosil. 2016. Proposed approach for targeted attacks detection. In *Advanced Computer and Communication Engineering Technology*. Springer, 73–80.

[27] Ibrahim Ghafir, Vaclav Prenosil, Ahmad Alhejailan, and Mohammad Hammoudeh. 2016. Social Engineering Attack Strategies and Defence Approaches. In *IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)*. IEEE Xplore Digital Library, 145–149.

[28] Ibrahim Ghafir, Vaclav Prenosil, and Mohammad Hammoudeh. 2015. Botnet Command and Control Traffic Detection Challenges: A Correlation-based Solution. *International Journal of Advances in Computer Networks and Its Security (IJCNS)* vol. 7, Issue 2 (2015), 27–31.

[29] Ibrahim Ghafir, Vaclav Prenosil, Jakub Svoboda, and Mohammad Hammoudeh. 2016. A survey on network security monitoring systems. In *IEEE International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*. IEEE Xplore Digital Library, 77–82.

[30] Ibrahim Ghafir, Jakub Svoboda, and Vaclav Prenosil. 2014. Tor-based malware and Tor connection detection. In *International Conference on Frontiers of Communications, Networks and Applications (ICFCNA)*. IEEE Xplore Digital Library, 1–6.

[31] Ibrahim Ghafir, Jakub Svoboda, and Vaclav Prenosil. 2015. A survey on botnet command and control traffic detection. *International Journal of Advances in Computer Networks and its security (ICJNS)* vol. 5, Issue 2 (2015), 75–80.

[32] Paul Giura and Wei Wang. 2012. A context-based detection framework for advanced persistent threats. In *Cyber Security (CyberSecurity), 2012 International Conference on*. IEEE, 69–74.

[33] Mohammad Hammoudeh, Fayez Al-Fayez, Huw Lloyd, Robert Newman, Bamidele Adebisi, Ahcène Bounceur, and Abdelrahman Abuarqoub. 2017. A Wireless Sensor Network Border Monitoring System: Deployment Issues and Routing Protocols. *IEEE Sensors Journal* (2017).

[34] Mohammad Hammoudeh, Omar Aldabbas, Sarah Mount, Saeed Abuzour, Mai Alfawair, and Serein Alratrout. 2010. Algorithmic construction of optimal and load balanced clusters in Wireless Sensor Networks. In *Systems Signals and Devices (SSD), 2010 7th International Multi-Conference on*. IEEE, 1–5.

[35] Mohammad Hammoudeh, Robert Newman, Christopher Dennett, Sarah Mount, and Omar Aldabbas. 2015. Map as a Service: A Framework for Visualising and Maximising Information Return from Multi-ModalWireless Sensor Networks. *Sensors* 15, 9 (2015), 22970–23003.

[36] Zhijie Liu, Chongjun Wang, and Shifu Chen. 2008. Correlating multi-step attack and constructing attack scenarios based on attack pattern modeling. In *Information Security and Assurance, 2008. ISA 2008. International Conference on*. IEEE, 214–219.

[37] Mandiant. 2017. Mandiant APT1 Report Appendix F Update: SSL Certificate Hashes. https://www.mandiant.com/blog/md5-sha1/. (2017). Accessed: 10-02-2017.

[38] David Moore, Colleen Shannon, Douglas J Brown, Geoffrey M Voelker, and Stefan Savage. 2006. Inferring internet denial-of-service activity. *ACM Transactions on Computer Systems (TOCS)* 24, 2 (2006), 115–139.

[39] Catherine Moxey, Mike Edwards, Opher Etzion, Mamdouh Ibrahim, Sreekanth Iyer, Hubert Lalanne, Mweene Monze, Marc Peters, Yuri Rabinovich, Guy Sharon, and others. 2010. A conceptual model for event processing systems. *IBM Redguide publication* (2010).

[40] Nir Nissim, Aviad Cohen, Chanan Glezer, and Yuval Elovici. 2015. Detection of malicious PDF files and directions for enhancements: a state-of-the art survey. *Computers & Security* 48 (2015), 246–266.

[41] Oracle. 2017. Network tracing. https://www.virtualbox.org/wiki/Network_tips. (2017). Accessed: 07-02-2017.

[42] Jose Luis Gomez Ortega, Liangxiu Han, and Nicholas Bowring. 2016. A Novel Dynamic Hidden Semi-Markov Model (D-HSMM) for Occupancy Pattern Detection from Sensor Data Stream. In *New Technologies, Mobility and Security (NTMS), 2016 8th IFIP International Conference on*. IEEE, 1–5.

[43] Vern Paxson. 1999. Bro: a system for detecting network intruders in real-time. *Computer networks* 31, 23 (1999), 2435–2463.

[44] Bro Project. 2017. Input Framework. https://www.bro.org/sphinx/frameworks/input.html. (2017). Accessed: 01-06-2017.

[45] Bro Project. 2017. x509_certificate event. https://www.bro.org/sphinx/scripts/base/bif/plugins/Bro_X509.events.bif.bro.html#id-x509_certificate. (2017). Accessed: 01-06-2017.

[46] Terry R Rakes, Jason K Deane, and Loren Paul Rees. 2012. IT security planning under uncertainty for high-impact events. *Omega* 40, 1 (2012), 79–88.

[47] Muhammad Jasim Saeed, Liangxiu Han, and Maybin K Muyeba. 2014. An energy efficient and resource preserving target tracking approach for wireless sensor networks. In *Communication Systems, Networks & Digital Signal Processing (CSNDSP), 2014 9th International Symposium on*. IEEE, 232–237.

[48] Joseph Sexton, Curtis Storlie, and Joshua Neil. 2015. Attack chain detection. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 8, 5-6 (2015), 353–363.

[49] Johan Sigholm and Martin Bang. 2013. Towards Offensive Cyber Counterintelligence: Adopting a Target-Centric View on Advanced Persistent Threats. In *Intelligence and Security Informatics Conference (EISIC), 2013 European*. IEEE, 166–171.

[50] Tamir Sobeih, Nick Whittaker, and Liangxiu Han. 2015. DIVINE: Building a Wearable Device for Intelligent Control of Environment Using Google Glass. In *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on*. IEEE, 1280–1285.

[51] Stuart Staniford, James A Hoagland, and Joseph M McAlerney. 2002. Practical automated detection of stealthy portscans. *Journal of Computer Security* 10, 1 (2002), 105–136.

[52] Jakub Svoboda, Ibrahim Ghafir, and Vaclav Prenosil. 2015. Network monitoring approaches: An overview. *International Journal of Advances in Computer Networks and Its Security (IJCNS)* vol. 5, Issue I (2015).

[53] Xu Wang, Kangfeng Zheng, Xinxin Niu, Bin Wu, and Chunhua Wu. Detection of command and control in advanced persistent threat based on independent access. In *IEEE International Conference on Communications*.

[54] Paul Wood, Mathew Nisbet, Gerry Egan, and others. 2012. Symantec internet security threat report trends for 2011. *Volume XVII* (2012).