In: Cyber-Security and Information Warfare
Editor: Nicholas J. Daras
ISBN: 978-1-53614-385-0

Chapter 5

R -T C I P F A A P T

Konstantinos Demertzis**[*] **and Lazaros Iliadis**[†]
Democritus University of Thrace,
School of Engineering, Department of Civil Engineering,
Faculty of Mathematics, Programming and General Subjects,
Kimmeria, Xanthi, Greece

Abstract

An Advanced Persistent Threat (APT) is a set of stealthy and continuous computer hacking processes in which an unauthorized person gains access to a network and stays there undetected for a long period of time. The "advanced" process signifies sophisticated techniques using zero-days malware to exploit vulnerabilities in systems. The "persistent" process suggests that an external command and control system is continuously monitoring and extracting data from a specific target. The "threat" process indicates human involvement in orchestrating the attack. APT attacks target organizations in sectors with high-value information, such as military networks, national defense, manufacturing and the financial industry. Traditional digital security mechanisms face such vulnerabilities in a conventional manner, they create often false alarms and they fail to forecast them. As APT activities are stealthy because they use Tor anonymity network, the command and control network traffic associated with APT can be undetected at the network layer level. Deep log analyses and log correlation from various sources cannot be useful in detecting APT activities and network agents are not able to collect logs TCP (*Transfer Control Protocol*) and UDP (*User Datagram Protocol*) directly from assets into a syslog server. This paper proposes an innovative fast and accurate Real-time Computational Intelligence Protection Framework against Advanced Persistent Threats (CIPFaAPT). It is about an automate forensic analysis system that uses Online Sequential Extreme Learning Machines. It can process millions of data points in real-time, establishing, or learning a "normal" baseline, comparing data points to past behavior and identifying anomalous differences in values over time, differences in rates over time and population outliers. Using computational intelligence and machine learning algorithms, all user transactions, server processes,

[*]E-mail address: kdemertz@fmenr.duth.gr.
[†]E-mail address: liliadis@civil.duth.gr.

internet traffic, alerts caused by the IPS (*Intrusion Prevention Systems*) and traffic flow can all be analyzed for unusual activities. The CIPFaAPT is a next generation security platform that uses sophisticated analytics to monitor, track and classify risk across critical network infrastructures in order to identify APT.

Keywords: Advanced Persistent Threats, Tor Anonymity Network, Tor Traffic Analysis, online sequential learning, extreme learning machine

1. Introduction

1.1. Advanced Persistent Threats

Cyber-attacks differ regarding their complexity and they are characterized by their persistency and their targets. More specifically, Advanced Persistent Threats (APT) are recently introduced and they can belong to either of the following types [1], [2]:

1. *Advanced*: The opponent is totally familiarized with the Cyber intrusion methods. He/she is capable to develop personalized tools and it is possible to exploit vulnerabilities which are not known to the public (zero days) in order to achieve adjustability to the needs of every attack.

2. *Persistent*: The opponent has long term targets, which he/she is trying to achieve without being traced and without considering time limitations.

3. *Threats*: The opponent is extremely capable, organized, funded and he/she has motivations.

During the APT attacks, the Tactics, Techniques and Procedures (TTPs) are adjustable in order to forecast and overcome the defense systems and security practices of the target [3].

1.2. APTs vs Traditional Threats

The APTs differ from the traditional cyber-attacks in terms of the following [1], [2], [3], [4]:

1. *Customized attacks*: Personalized tools and techniques are used for these attacks which are designed to perform optimally for the specific case. These tools include zero-day exploits, stealth virus, worms and rootkits. Additionally, the APTs apply tactics of multiple simultaneous attacks known as “kill chains”, aiming in violating the defensive mechanisms of the targets in order to ensure unlimited access. Their approach occasionally includes the sacrifice of a threat. In this way, the target might falsely believe that the attack has been successfully faced whereas he might be in big trouble.

2. *Low and slow:* The APT attacks take place and evolve in long periods of time, where the attackers are moving slowly and silently in order to avoid to be traced. The whole process includes application of simultaneous and continuous monitoring combined with targeted interaction till the achievement of their goals.

3. *Higher aspirations*: The APTs differ from the opportunistic fast performance attacks, in the sense that they have being designed to satisfy the demands and the targets of international espionage organizations, secret services, states policies, industrial espionage, military or terrorist operations. All of the above usually include covered support by state officials. The groups behind the APTs are well funded by secret capitals.

4. *Specific targets*: The APT targets are very specific and they usually include critical infrastructure like electrical energy networks, government services, critical installations of national infrastructure like defense industry.

1.3. How do APT Attacks Work?

The APTs are designed and executed very carefully and they include the following four specific phases: Incursion, Discovery, Capture, and Exfiltration [1], [3], [5]. Various techniques can be employed in each phase as indicated below [1], [2], [5]:

1. Phase 1 *Incursion*: In the cases of targeted attacks, the cyber criminals are trying to invade the target's network and to overcome the defensive mechanisms by using social engineering, zero-day vulnerabilities, SQL injection, targeted malware or other methods. In most of the cases the above approaches are used in parallel and combined, aiming in the creation of an initial robust access point, that will be used to initiate long term secret cyber-attacks.

2. Phase 2 Discovery: After the successful intrusion, the intruder is mapping the organization's systems and automatically scans for confidential data, operational instructions and functional information of the organism. This phase also includes the discovery of network, software or hardware vulnerabilities. This is done carefully, in order to avoid tracing and discovery of the attack.

3. Phase 3 Capture: This phase includes recording of the exposed data which are stored in unprotected systems and they are directly available, whereas *rootkits* are stored in targeted systems and access points, aiming in the full recording of the organism's data.

4. Phase 4 Exfiltration: As soon as the targeted systems are captured, the intruders can move to the final actions of their plan, which can be related to the stealing of rights, patents, or other confidential data, the deactivation of weapons or the destruction of control systems.

2. Relevant Malicious Activity

2.1. APTs Threat Intelligence

Exploring the way in which the ATP attacks are operating requires analysis of the complex attackers' techniques, understanding their motivations plus intentions and moreover their own characteristics. The following concepts must be explored and understood [1], [2], [3], [4], [5]:

1. *Victim's intelligence*: It is related to the recording of action methods and experiences gained by the staff that handled the attack case.

2. *Machine intelligence*: It refers to the search and analysis of the critical hardware and software points (e.g., *Firewall, SysLog Servers, Switch*) for the discovery of traces that reveal the types of attacker's actions.

3. *Adversarial intelligence*: This term includes the collection of information gathered during the detection of the most recent attacks in the cyber space. It includes the data obtained from the start of the attack, the tools used and the targets of the attacks. More complex attacks like botnets, ransomware, remote access Trojans and zero-day malware, are related to or they are part of the APT attacks.

Searching the most recent methods and techniques used by the cyber criminals is a first priority process towards understanding the way the ATPs are operating [3], [5].

2.2. Bots & Botnets

Bots [6] are one of the most sophisticated and popular types of cybercrime today. They allow hackers to take control of many computers at a time, and turn them into "zombie" computers, which operate as part of a powerful "botnet" [6]. Botnets employ evolving techniques to obfuscate the specific host involved in their phishing schemes, malware delivery or other criminal enterprises, like money mule recruitment sites, illicit online pharmacies, extreme or illegal adult content sites, malicious browser exploit sites and web traps for distributing virus [6].

One of the biggest challenges for botnet owners is the protection of Command-and-Control traffic (C&C). C&C traffic is required to give orders to the "zombies", the infected computers that are part of the botnets. Generally, up to now, two approaches existed for C&C traffic: Either a central control server is put somewhere on the Internet or Peer-to-Peer-networks (P2P) are built up to ensure the chain of commands [6], [7], [8].

2.3. IP-Flux

IP-Flux [9], [10] refers to the constant changing of IP address information (e.g., 192.168.1.1) related to a particular, fully qualified domain name (e.g., mypc.atl.damballa.com). Botnet operators abuse this ability to change IP address information associated with a host name by linking multiple IP addresses with a specific host name and rapidly changing the linked addresses. These IPs are interchanged too fast, with a very small Time-To-Live (TTL) for each partial DNS Resource Record [11], [12].

In this way, a domain name can change its corresponding IP address very often (e.g., every 3 minutes). This rapid changing aspect is referred to as "Fast-Flux" [9], [10], [11].

Single-flux is characterized by having multiple IP addresses associated with a domain name. These IP addresses are registered and de-registered rapidly – using a combination of round-robin allocation and very short TTL values against a particular DNS Resource Record. DNS A records that change quickly [9], [11].

On the other hand, Double-flux not only fluxes the IP addresses associated with the Fully-Qualified Domain Name (FQDN), but also fluxes the IP addresses of the DNS servers

(e.g., NS records) that are in turn used to lookup the IP addresses of the FQDN. DNS A and NS records change quickly [9], [10], [11], [12].

2.4. Blind Proxy Redirection (BPR)

Redirection disrupts attempts to track down and mitigate fast-flux service network nodes [10], [12]. What happens is the large pool of rotating IP addresses are not the final destination of the request for the content (or other network service). Instead, compromised front end systems are merely deployed as redirectors that funnel requests and data to and from other backend servers, which actually serve the content. Essentially the domain names and URLs for advertised content no longer resolve to the IP address of a specific server, but instead fluctuate amongst many front-end redirectors or proxies, which then in turn forward content to another group of backend servers [11].

2.5. Domain Flux

Domain Flux [9], [10] is effectively the inverse of IP flux and refers to the constant changing and allocation of multiple FQDN's to a single IP address or C&C infrastructure [11], [12].

Domain Wildcarding abuses native DNS functionality to wildcard (e.g., *) a higher domain such that all FQDN's point to the same IP address. For example, *.damb.com could encapsulate both mypc.atl.damb.com and server.damb.com. This technique is most commonly associated with spam or phishing botnets – whereby the wildcarded information that appears random (e.g., "asdk" of asdk.atl.damb) is used by the botmasters to uniquely identify a victim, track success using various delivery techniques, and bypass anti-spam technologies [9], [10], [11], [12].

2.6. Domain Generation Algorithm (DGA)

Bot agents create a dynamic list of multiple FQDN's that can be used as rendezvous points with their C&C servers [13], [14], [15]. The large number of potential rendezvous points makes it difficult for law enforcement to effectively shut down botnets since infected computers will attempt to contact some of these domain names every day to receive updates or commands. By using public-key cryptography, it is unfeasible for law enforcement and other actors to mimic commands from the malware controllers as some worms will automatically reject any updates not signed by the malware controllers. For example, an infected computer could create thousands of domain names such as: www.gi9bfb4er2ig4fws8h.ir and would attempt to contact a portion of these with the purpose of receiving an update or commands. Embedding the DGA instead of a list of previously-generated (by the C&C servers) domains in the unobfuscated binary of the malware protects against a strings dump that could be fed into a network blacklisting appliance preemptively to attempt to restrict outbound communication from infected hosts within an enterprise [13], [14], [15].

3. Tor-Based Botnets

3.1. Tor Network

Tor is generally known as web anonymization service for end users, but Tor [16] offers more than that: "Tor makes it possible for users to hide their locations while offering various kinds of services, such as web publishing or an instant messaging server." Tor directs Internet traffic through a free, worldwide, volunteer network consisting of more than seven thousand relays to conceal a user's location and usage from anyone conducting network surveillance or traffic analysis [17]. Using Tor makes it more difficult for Internet activity to be traced back to the user. In this particular case, the creators of the malware decided to build an IRC server as hidden service.

3.2. Tor Traffic Analysis

The objective of Tor [16] is to conceal the user IDs and their activity in the network, in order to prevent the monitoring and analysis of the traffic and to separate the detection from the routing by using virtual circuits, or overlays, which change periodically.

It is the implementation of onion routing [17], in which multiple layers of encryption are employed, in order to ensure perfect forward secrecy between the nodes and the hidden services of Tor, while launching randomly the communication via tor nodes (consensus) operated by volunteers worldwide. Although the Tor network is operating in the Transport layer of the OSI, the onion proxy software shows customers the *Secure Socket Interface* (SOCKS) which operates in the session layer.

Also, a continuous redirection of traffic requests between the relays, (entry guards, middle relays and exit relays), takes place in this network. Both the sender and recipient addresses and the information are in the form of encrypted text, so that no one at any point along the communication channel cannot decrypt the information or identify both ends directly [17]. The most famous types of malware are seeking communication recovery and its maintenance with the C&C remote servers on a regular basis, so that botmasters can collect or transfer information and upgrades to the compromised devices (bots). This communication is usually performed using hardcoded address or default lists address (pool addresses) controlled by the creator of the.

The mode of communication of the latest, sophisticated malware generations, lies in the creation of an encrypted communication channel, based on the chaotic architecture of Tor, in order to alter the traces and to distort the elements that define an attack and eventually to increase the complexity of the botnets.

Although modern programming techniques enable the malware creators to use thousands, alternating and different subnet IP address, in order to communicate with their C2 servers, the trace of those IPs is relatively straightforward for the network engineers, or for the responsible security analysts. Once identified, they are included in a blacklist and eventually they are blocked as spam. On the other hand, the limitation of the Tor-based botnets is extremely difficult because the movement of the Tor network resembles that of the HTTPS (*Hypertext Transfer Protocol Secure*) protocol [16], [17], [18].

3.3. Tor vs HTTPS

The Tor network not only performs encryption, but it is also designed to simulate normal HTTPS protocol traffic, which makes the identification of its channels an extremely complex and specialized process, even for experienced engineers or network analyzers. Specifically, the Tor network can use the TCP port 443, which is used by the HTTPS, so that the supervision and interpretation of a session exclusively with the determination of the door cannot constitute a reliable method.

A successful method for detecting Tor traffic is the statistical analysis and the identification of the Secure Sockets Layer protocol differences (SSL) [18]. The SSL protocol uses a combination of public and symmetric key encryption. Each SSL connection always starts with the exchange of messages by the server and the client until the secure connection is established (handshake). The handshake allows the server to prove its identity to the client by using public-key encryption techniques and then allows the client and the server to cooperate in the creation of a symmetric key to be used to quickly encrypt and decrypt data exchanged between them. Optionally, the handshake also allows the client to prove its identity to the server [18]. Given that each Tor client creates self-signed SSL, using a random domain name that changes around every 30 minutes, a statistical analysis of the network traffic based on the specific SSL characteristics can identify the Tor sessions, in a network full of HTTPS traffic.

4. The Proposed System

4.1. CIPFaAPT

Since information systems' security is an extremely complex process, the systems' administrators cannot be based only in the use of specific isolated protection products installed in each checkpoint aiming to avoid an incident. The detection of an intrusion in a terminal, in the network, or in the email gate is a manual and time consuming process, something that offers an important advantage to the attackers. In most of the ATPs this only aims to trick the system and to cover more serious threats. The CIPFaAPT is a forensic analysis system which uses *Online Sequential Extreme Learning Machines* (OSELM) [19], [20]. It can process multiple data in real time mode, in order to detect "*non-normal*" system's behavior by comparing to past data. In this way, it is possible to check and detect potential anomalies in various cases over time (e.g., user transactions, server processes, internet traffic, IPS alerts and traffic flow). The analysis is done by using computational intelligence and advanced machine learning algorithms.

The CIPFaAPT is a next generation platform using advanced systems for the tracing and classification of risk in critical infrastructures aiming in the detection of ARTs. It correlates the suspicious activities in all of the control points and it classifies the facts that appear to have the highest risk, whereas it activates the defense mechanisms as soon as it spots a critical threat.

More specifically, the CIPFaAPT offers the following potentials:

1. It reveals the total spectrum of the APT threats with a combined trace of the key control points of the terminals and the network.

2. It offers priority for the confrontation of the threats that are worth attention among all local control points.

3. It blocks potential new incidents

This chapter proposes the development of the CIPFaAPT, a cyber-threat bio-inspired intelligence system, which provides smart mechanisms for the supervision of networks. It provides intelligent approaches and it can defend over sophisticated attacks and of exploiting effectively the hardware capabilities with minimum computational and resources cost. More specifically, this research proposes an innovative and very effective Online Sequential Extreme Learning Machine model that it is proper for big data analysis, for solving a multidimensional and complex cyber security problem.

4.2. Innovation of the Proposed Method

APTs are the most sophisticated and highly intelligent techniques that make detection of "contamination" and analysis of malicious code, a very complex task. It is a fact that they spread through chaotic Tor-based botnets in which communication is done using the anonymity Tor network, which makes it impossible to identify and locate the Command and Control (C&C) servers. *Tor* is free software for enabling anonymous communication. The name Tor is derived from an acronym for the original software project name "*The Onion Router*". In addition, the network traffic for the Tor packet is designed to simulate the respective traffic of the HTTPS protocol which causes serious Tor traffic identification weaknesses by the motion analysis systems. Finally, given the passive mode of traditional security systems, which are unable in most cases to identify these types of major threats, the development and use of alternative more radical and more substantial methods appear as a necessity. This work proposes the development and testing of a novel computational intelligence system named CIPFaAPT. The system requires the minimum consumption of resources and it significantly enhances the security mechanisms of the network OS.

Specifically, the architecture of the proposed system is based on the Online Sequential Extreme Learning Machines. The CIPFaAPT employs the OSELM algorithm in order to perform malware localization, DGA and Tor traffic identification and botnets prohibition.

The CIPFaAPT system is a Biologically inspired artificial intelligence computer security technique [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35], [36]. Unlike other existing approaches which are based on individual passive safety techniques, the CIPFaAPT is an integrated active safety system. It provides intelligent surveillance mechanisms and classification of malware, it is able to defend itself and to protect from DGA malware, it detects and prevents encrypted Tor network activities and it can efficiently exploit the potential of the hardware, with minimal computational cost.

An innovation of the CIPFaAPT approach is related to the architecture of the proposed computational intelligence system, which use for the first time a very fast and highly effective biologically inspired machine learning algorithm towards the solution of a multidimensional and complex IT security problem. Furthermore, a major innovative feature of this proposal is related to the identification and separation of the Tor network traffic from the traffic of the HTTPS protocol, which is presented from the authors for first time in network traffic analysis systems [28].

4.3. Literature Review

Traffic analysis attacks have been extensively studied over the past decade [37], [38]. The authors have acknowledged the potential of machine learning-based approaches in providing efficient and effective detection, but they have not provided a deeper insight on specific methods, neither the comparison of the approaches by detection performances and evaluation practice. On the other hand, Cheng et al. [39] proposed the use of ELM methods to classify binary and multi-class network traffic for intrusion detection with high accuracy. Hsu et al. [40] proposed a real-time system for detecting botnets based on anomalous delays in HTTP/HTTPS requests from a given client with very promising results. Also, Haffner et. al. [41] employed AdaBoost, Hidden Markov, Naive Bayesian and Maximum Entropy models to classify network traffic into different applications, with very high SSH (Secure Shell, is a cryptographic network protocol operating at layer 7 of the OSI Model to allow remote login and other network services to operate securely over an unsecured network) detection rate and very low false positive rate, but they employed only few bytes of the payload. Furthermore, Alshammari et al. [42] employed Repeated Incremental Pruning, to Produce Error Reduction (RIPPER) and AdaBoost algorithms for classifying SSH traffic from offline log files without using any payload, IP addresses or port numbers.

Nhauo et al. [43] proposed a method for Classification of Malicious Domains using Support Vector Machine and Bi-gram algorithm using only domain names and their results showed that features extracted by bi-gram were performing a lot better than single alphanumeric character. Antonakakis et al. [44] uses a combination of clustering and classi?cation algorithms to detecting the DGA-Based Malware. Zhao et al. [45] select a set of attributes from the network ?ows and then applies a Bayes network and a decision tree algorithm to classify malicious traffic.

Chakravarty et al. [46] assess the feasibility and effectiveness of practical traffic analysis attacks against the Tor network using NetFlow data and proposed an active traffic analysis method based on deliberately perturbing the characteristics of user traffic at the server side, and observing a similar perturbation at the client side through statistical correlation. Almubayed et al. [47] proposed a research has considered many ML algorithms in order to fingerprint Tor usage in the network. Chaabane et al. [48] provides a deep analysis of both the HTTP and BitTorrent protocols giving a complete overview of their usage, depict how users behave on top of Tor and also show that Tor usage is now diverted from the onion routing concept and that Tor exit nodes are frequently used as 1-hop SOCKS proxies, through a so-called tunneling technique. Finally, Chakravarty et al. proposed methods for performing traffic analysis using remote network bandwidth estimation tools, to identify the Tor relays and routers involved in Tor circuits [49], [50].

4.4. Datasets

The selection of the data was the result of an extensive research on the functionality of the protocol SSL, combined with a deep analysis of the independent variables, in order to obtain the ones that give the maximum precision under the strict condition of minimal computing resources consumption. This effort resulted in the creation of training sets, capable to properly train the employed learning algorithms.

Table 1. MTA: Extracted features from network traffic (31 Independent and 2 depended)

ID	Feature Name	ID	Feature Name
1	seq_number	17	response_seq_number
2	ack_number	18	response_ack_number
3	src_port	19	response_src_port
4	dst_port	20	response_dst_port
5	fin	21	response_fin
6	syn	22	response_syn
7	rst	23	response_rst
8	psh	24	response_psh
9	ack	25	response_ack
10	window	26	response_window
11	check	27	response_check
12	ip_len	28	response_ip_len
13	ip_id	29	response_ip_id
14	ip_off	30	response_ip_off
15	ip_ttl	31	response_ip_ttl
16	ip_sum	32	**Classes (Benign or Malicious)**

Four datasets with high complexity were constructed and used for testing by the CIP-FaAPT. The first Malware Traffic Analysis (MTA) dataset comprised of 32 independent variables and 2 classes (benign or malware). This dataset containing 73,469 patterns (37,127 benign samples they were chosen from the Pcaps from National Cyber Watch Mid-Atlantic Collegiate Cyber Defense Competition and 36,342 malicious samples they were chosen from http://malware-traffic-analysis.net/) [51]. The idea of the network traffic analysis and the features extraction approach were based on the functional mode of the TCP protocol and moreover on the acknowledgement method of the reliable submission and receipt of the data. Also, it relies on the error-free data transfer mechanisms between the network layer and the application layer, of the TCP header structure and the three-way handshake process [52].
The full list of the 32 features with the class is detailed in Table 1.

The second Network Traffic Classification (NTC) dataset comprised of 22 independent variables and 12 network traffic classes (TELNET, FTP, HTTP, HTTPS, DNS, Lime, Local Forwarding, Remote Forwarding, SCP, SFTP, x11 and Shell). This dataset containing 137,050 patterns they were chosen from the Pcaps from Information Technology Operations Center (ITOC), US Military Academy [53].

The features management and export (for tables 2 and 3) was based on the analysis of the network traffic and specifically on the methodology used in [54]. The full list of the 22 features with the corresponding classes is presented in the following Table 2.

Table 2. NTC: Extracted features from network traffic (22 Independent and 12 depended)

ID	Feature Name	ID	Feature Name
1	min_fpktl	13	min_biat
2	mean_fpktl	14	mean_biat
3	max_fpktl	15	max_biat
4	std_fpktl	16	std_biat
5	min_bpktl	17	duration
6	mean_bpktl	18	proto
7	max_bpktl	19	total_fpackets
8	std_bpktl	20	total_fvolume
9	min_fiat	21	total_bpackets
10	mean_fiat	22	total_bvolume
11	max_fiat	23	**Classes (TELNET, FTP, HTTP, HTTPS, DNS, Lime, Local Forwarding, Remote Forwarding, SCP, SFTP, x11 and Shell)**
12	std_fiat		

The third Tor-Traffic Identification (TTI) dataset comprised of 45 independent variables and 2 classes (Tor or HTTPS). This dataset containing 217,483 patterns they were chosen from the Pcaps from [55]. The full list of 45 features with their corresponding classes is presented in the following Table3.

In the preprocessing process the duplicate records and records with missing values were removed. Also, the datasets were determined and normalized to the interval [-1,1] to phase the problem of prevalence of features with wider range over the ones with a narrower range, without being more important [56].

Considering the limited capacity of resources and computing power of mobile devices and the limitations posed by their dependence on the battery, we have made a transformation of the TTI independent variables vector space, which is a very complex and extensive dataset. Principal Components Analysis (PCA) has been performed to obtain new linear combinations, capable to contain the largest possible part of the variance of the original information, without limiting the predictive capability and accuracy of the learning algorithm. However, since the results have deteriorated enough (almost 12% less accuracy was obtained) the approach was abandoned.

Then, we have performed correlation analysis on various subsets of features which had the highest correlation with the obtained class, regardless of their interrelation. Also, other subsets were used, which were highly correlated with the class and they appeared to have high cross-correlation (Correlation Attribute Evaluation) [57].

Table 3. TTI: Extracted flow statistics from network traffic (45 Independent and 2 depended)

ID	Feature Name	ID	Feature Name
1	srcip	24	max_biat
2	srcport	25	std_biat
3	dstip	26	duration
4	dstport	27	min_active
5	proto	28	mean_active
6	total_fpackets	29	max_active
7	total_fvolume	30	std_active
8	total_bpackets	31	min_idle
9	total_bvolume	32	mean_idle
10	min_fpktl	33	max_idle
11	mean_fpktl	34	std_idle
12	max_fpktl	35	sflow_fpackets
13	std_fpktl	36	sflow_fbytes
14	min_bpktl	37	sflow_bpackets
15	mean_bpktl	38	sflow_bbytes
16	max_bpktl	39	fpsh_cnt
17	std_bpktl	40	bpsh_cnt
18	min_fiat	41	furg_cnt
19	mean_fiat	42	burg_cnt
20	max_fiat	43	total_fhlen
21	std_fiat	44	total_bhlen
22	min_biat	45	dscp
23	mean_biat	46	**Classes (Tor or HTTPS)**

Finally, we have tried to use subsets for which we have calculated the cost-sensitive classification, based on the cost-matrix (Cost Sensitive Subset Evaluation). The method takes a cost matrix and a base evaluator. Cost matrix is a way to change the threshold value for a decision boundary. If the base evaluator can handle instance weights, then the training data is weighted per the cost matrix, otherwise the training data is sampled per the cost matrix. The process of performing Cost Sensitive Subset Evaluation is a very effective method because the error-based methods consider the classification errors as equally likely, which is not the case in all the real-time applications. [57], [58]. The subsets for which the value of each feature was calculated using the information gain with respect to the class (Information Gain Attribute Evaluation) [57].

Eventually, the subset chosen was based on the method of Correlation-based Feature Subset Selection (subsets of features that correlate highly with the class value and appear low correlation with each other). From this dataset, we have obtained the minimum error of the classifier in the training and test data, in relation to the value of each feature (Attribute

Evaluation with particle swarm optimization) [57].

Finally, we have gained 33,5% reduction of the initial parameters, whereas the accuracy dropped only by 0.1% compared to the accuracy of the system that used all 45 features. The following table 4 presents the 30 features included in the final dataset.

Table 4. The TTI feature vector after the feature selection process

ID	Feature Name	Interpretation
1	srcip	The source IP address of the flow.
2	srcport	The source port number of the flow.
3	dstip	The destination IP address of the flow.
4	dstport	The destination port number of the flow.
5	total_fpackets	The total number of packets travelling in the forward direction.
6	total_bpackets	The total number of packets travelling in the backward direction.
7	min_fpktl	The minimum packet length (in bytes) from the forward direction.
8	max_fpktl	The maximum packet length (in bytes) from the forward direction.
9	min_bpktl	The minimum packet length (in bytes) from the backward direction.
10	max_bpktl	The maximum packet length (in bytes) from the backward direction.
11	min_fiat	The minimum interarrival time (in microseconds) between two packets.
12	max_fiat	The maximum interarrival time (in microseconds) between two packets.
13	min_biat	The minimum interarrival time (in microseconds) between two packets.
14	max_biat	The maximum interarrival time (in microseconds) between two packets
15	duration	The time elapsed (in microseconds) from the first packet to the last packet.
16	min_active	The minimum duration (in microseconds) of a sub-flow.
17	max_active	The maximum duration (in microseconds) of a sub-flow.
18	min_idle	The minimum time (in microseconds) the flow was idle.
19	max_idle	The maximum time (in microseconds) the flow was idle.
20	sflow_fpackets	The average number of forward travelling packets in the sub-flows.
21	sflow_fbytes	The average number of bytes, travelling in the forward direction.
22	sflow_bpackets	The average number of backward travelling packets in the sub-flows.
23	sflow_bbytes	The average number of bytes, travelling in the backward direction.
24	fpsh_cnt	The number of times the PSH flag was set for packets travelling in the forward direction.
25	bpsh_cnt	The number of times the PSH flag was set for packets travelling in the backward direction.
26	furg_cnt	The number of times the URG flag was set for packets travelling in the forward direction.
27	burg_cnt	The number of times the URG flag was set for packets travelling in the backward direction.
28	total_fhlen	The total header length (network and transport layer) of packets travelling in the forward direction.
29	total_bhlen	The total header length (network and transport layer) of packets travelling in the backward direction.
30	dscp	Differentiated services code point, a field in the IPv4 and IPv6 headers.

Table 5. DGA dataset: Extracted features from domain names (5 Independent and 2 depended)

ID	Feature Name	Interpretation
1	length	The length of the strings of the domains.
2	entropy	The entropy of each domain as degree of uncertainty, with the higher values met in the DGA domains.
3	alexa_grams	The degree of coherence between the domain and the list of domains originating from Alexa. This is done with the technique of the probability linguistic model for the forecasting of the next n-gram element.
4	word_grams	The degree of coherence between the domain and a list of 479,623 words or widely used characters. It is estimated with the same method as in the previous one.
5	differences	The difference between the values of alexa_grams and word_grams.
6	**Classes**	**Legit or Malicious.**

Finally, a dataset namely Domain Generation Algorithms (DGA) dataset constructed and used for testing. As legit domains 100,000 domain names were used. They were chosen randomly from the database with the 1 million most popular domain names of Alexa [59]. For the malicious domains, the updated list of the Black Hole DNS database was used [60]. This list includes 16,374 records from domains that have been traced and characterized as dangerous. More over 15,000 domain name records were added labeled as malicious. They were created based on a time stamp DGA algorithm, with length from 4 to 56 characters of the form 18cbth51n205gdgsar1io1t5.com. Also, 15,000 domain name records were added labeled as malicious, which were created with the use of words of phrases coming from an English dictionary. Their length varied from 4 to 56 characters of the form hotsex4rock69burningchoir.com. The full list of features with their corresponding classes is presented in the following Table 5 [61].

Duplicate records and records with incompatible characters were removed. Also, the outliers removed based on the Inter Quartile Range (IQR) technique [62]. After this preprocessing operation, the DGA dataset containing 136,519 patterns.

5. Research Methodology

5.1. Online Sequential Extreme Learning Machine

The CIPFaAPT is essentially a tool for analysis of web streaming traffic in fixed intervals, to extract timely conclusions in which some or all the incoming data is not available for access from any permanent or temporary storage medium, but it arrives in a form of consecutive flows. For these data, there is no control over the order in which they arrive, their size may vary and many of them offer no real information. Also, the examination of individual IP packets or TCP segments, can extract only few conclusions and therefore the interdependence of the individual packets to each other, their analysis cannot be done with simple static methods, but it requires further modeling of traffic and the use of advanced analytical methods for the extraction of knowledge from complex data sets. This modeling is achieved using the computational intelligence Online Sequential Extreme Learning Machine algorithm [19],[20].

The Extreme Learning Machine (ELM) [63] as an emerging biologically inspired learning technique provides efficient unified solutions to "generalized" Single-hidden Layer feed forward Networks (SLFNs) but the hidden layer (or called feature mapping) in ELM need not be tuned [63]. Such SLFNs include but are not limited to support vector machine, polynomial network, RBF networks, and the conventional feed forward neural networks. All the hidden node parameters are independent from the target functions or the training datasets and the output weights of ELMs may be determined in different ways (with or without iterations, with or without incremental implementations). ELM has several advantages, ease of use, faster learning speed, higher generalization performance, suitable for many nonlinear activation function and kernel functions [63].

According to the ELM theory [63], the ELM with Gaussian Radial Basis Function kernel (GRBFK) $K(u,v)=exp(-\gamma\|u-v\|^2)$ is used in this approach. The hidden neurons are k=20 that chosen with trial and error method. Subsequently, $w_i$are the assigned random input weights and b_i,i=1,. . . ,N are the biases. To calculate the hidden layer output matrix *H,* the equation (1) is used [63].

$$H = \begin{bmatrix} h(x_1) \\ \vdots \\ h(x_N) \end{bmatrix} = \begin{bmatrix} h_1(x_1) & \cdots & h_L(x_1) \\ \vdots & & \vdots \\ h_1(x_N) & \cdots & h_L(x_N) \end{bmatrix} \tag{1}$$

$h(x) = [h_1(x), \ldots, h_L(x)]$ is the output (row) vector of the hidden layer with respect to the input x. Also $h(x)$ actually maps the data from the d-dimensional input space to the L-dimensional hidden-layer feature space (ELM feature space) H and thus $h(x)$ is indeed a feature mapping. ELM is to minimize the training error as well as the norm of the output weights [63]:

$$Minimize : \|H\beta - T\|^2 and \|\beta\| \tag{2}$$

where H is the hidden-layer output matrix of the equation (1), $\|\beta\|$ is used to minimize the norm of the output weights and actually to maximize the distance of the separating margins of the two different classes in the ELM feature space$2/\|\beta\|$.

To calculate the output weights β the function (3) is used [63]:

$$\beta = (\frac{I}{C} + H^T H)^{-1} H^T T \tag{3}$$

where c is a positive constant is obtained and T resulting from the *Function Approximation of SLFNs with additive neurons* [63]

$$T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}$$

which is an arbitrary distinct sample with $t_i=[t_{i1}, t_{i2},\ldots,t_{im}]^T \in R^m$

The OSELM [19],[20] is an alternative technique for large-scale computing and machine learning approaches that used when data becomes available in a sequential order to determine a mapping from data set corresponding labels. The main difference between online learning and batch learning techniques is that in online learning the mapping is updated after the arrival of every new data point in a scale fashion, whereas batch techniques are used when one has access to the entire training data set at once. It is a versatile sequential learning algorithm because of the training observations are sequentially (one-by-one or chunk-by-chunk with varying or ?xed chunk length) presented to the learning algorithm [19],[20]. At any time, only the newly arrived single or chunk of observations (instead of the entire past data) are seen and learned. A single or a chunk of training observations is discarded as soon as the learning procedure for that particular (single or chunk of) observation(s) is completed. The learning algorithm has no prior knowledge as to how many training observations will be presented. Unlike other sequential learning algorithms which have many control parameters to be tuned, OSELM with RBF kernel only requires the number of hidden nodes to be speci?ed [19],[20].

The proposed method uses an OSELM that can learn data chunk-by-chunk with a fixed chunk size of 20x20, with RBF kernel classification approach in order to perform malware localization, Tor traffic identification and botnets prohibition in an energetic security mode that needs minimum computational resources and time. The OSELM consists of two main phases namely: Boosting Phase (BPh) and Sequential Learning Phase (SLPh). The BPh used to train the SLFNs using the primitive ELM method with some batch of training data in the initialization stage and these boosting training data will be discarded as soon as boosting phase is completed. The required batch of training data is very small, which can be equal to the number of hidden neurons [19],[20],[63].

The general classification process with OSELM classifier described below:

Phase 1 (BPh) [19],[20].
The process of BPh for a small initial training set $N=\{(x_i, t_i)|x_i \in R^n, t_i \in R^m, i=1, \ldots, \tilde{N}\}$ described as follows:

1. Assign arbitrary input weight $w\beta^{(0)} = M_0 H_0^T T_{0i}$ and bias b_i or center m_i and impact width σ_i, i=1, ... $\tilde{N}$, where $\tilde{N}$ number for hidden neuron or RBF kernel for a specific application.

2. Calculate the initial hidden layer output matrix $H_0 = [h_1, \ldots, h_{\tilde{N}}]^T$, where $h_i = [g(w_1 * x_i + b_1), \ldots, g(w_{\tilde{N}} * x_i + b_{\tilde{N}})]^T$, $i = 1, \ldots, \tilde{N}$, where g activation function or RBF kernel.

3. Estimate the initial output weight, where $M_0 = (H_0^T H_0)^{-1}$ and $T_0 = [t_1, \ldots, t_{\tilde{N}}]^T$.

4. Set $k = 0$.

Phase 2 (SLPh) [19],[20].
In the SLPh the OSELM will then learn the train data chunk-by-chunk with a fixed chunk size of 20x20 and all the training data will be discarded once the learning procedure on these data is completed. The essentials step of this phase for each further coming observation(x_i, t_1), where $x_i \in R^n$, $t_i \in R^m$ and $i = \widetilde{N} + 1, \widetilde{N} + 2, \widetilde{N} + 3$, described as follow:

1. Calculate the hidden layer output vector $h_{(k+1)} = [g(w_1 * x_i + b_1), \ldots, g(w_{\tilde{N}} * x_i + b_{\tilde{N}})]^T$

2. Calculate latest output weight $\beta^{(k+1)}$ by the algorithm $\widehat{\beta} = (H^T H)^{-1} H^T T$ which is called the Recursive Least-Squares (RLS) algorithm.

3. Set $k = k + 1$

The proposed CIPFaAPT includes the following ruleset which is the core of its reasoning and described below.

Step 1. Performs malware localization by OSELM with Malware Traffic Analysis (MTA) dataset. If the malware analysis gives a positive result (Malware) the network traffic blocked and the process terminated. If the malware analysis gives a negative result (Benign), no action is required and goes to step 2.
Step 2. Performs network traffic analysis by OSELM with Network Traffic Classification (NTC) dataset. If the network traffic classification result is not a HTTPS, no action is required. If the network traffic classification result is a HTTPS, go to the next step 3.
Step 3. Performs Tor-traffic identification by OSELM with Tor-Traffic Identification (TTI) dataset. If the botnet classification result gives a positive result (Botnet) the network traffic blocked and the process terminated. If the botnet classification result gives a negative result (HTTPS), go to step 4.
Step 4. Performs domain identification by OSELM with Domain Generation Algorithms (DGA) dataset. If the botnet classification result gives a positive result (Malicious) the network access blocked and the process terminated. If the classification result gives a negative result (Legit), no action is required.

The overall algorithmic approach of CIPFaAPT that is proposed herein is described clearly and in details in the following Figure 1.

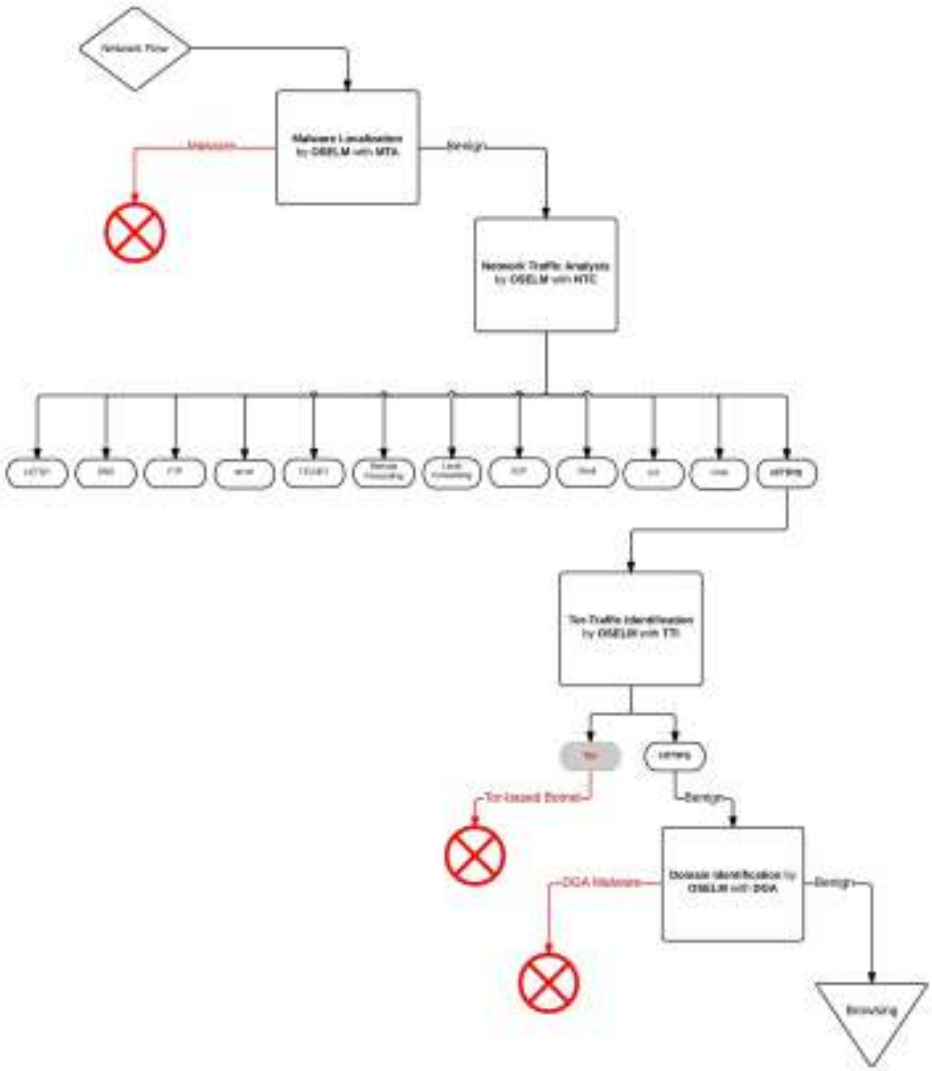

Figure 1. The proposed architecture of the CIPFaAPT

6. Results and Comparative Analysis

Given the complexity of the analysis and of the network traffic monitoring which is a realistic and very difficult task of computer security sector, the proposed system managed to perform with high accuracy. Also, it is important that the created datasets appear to have particularly high complexity, as they emerged taking into account even the most unfavorable scenarios and almost all potential cases of network traffic that may occur. This is a factor that played an important role towards the generalization capacity of the proposed system. It is characteristic that during the classification process, in each case of different scenarios represented using different datasets, the proposed system achieved accuracy rates of at least 93.97%.

Performance evaluation was performed based on a thorough comparative analysis (20 trials for each test and we compute the average result) of the obtained prediction accuracy and generalization capability between the CIPFaAPT and the following machine learning methods [19],[20],[63], namely: ELM with RBF activation function, ELM with Sigmoid activation function, OSELM with RBF activation function and 1 by 1 learning mode, OS-ELM with RBF activation function and 20 by 20 learning mode, OSELM with Sigmoid activation function and 1 by 1 learning mode and OSELM with Sigmoid activation function and 20 by 20 learning mode.

It is extremely comforting and hopeful, the fact that the proposed system manages to solve a particularly complex cyber security problem with high accuracy. Regarding the overall efficiency of the methods, the results show that the OSELM with RBF activation function and 20 by 20 learning mode, has much better generalization performance and more accurate classification output from the other compared algorithms.

The following tables 6, 7, 8 and 9 presents the analytical values of the predictive power of the method and the corresponding results when competitive algorithms were used.

Table 6. Comparison between algorithms in the MTA dataset

Classifier	**Properties**		**Classification Accuracy & Performance Metrics**					
	Activation Function	Learning Mode	ACC	RMSE	Precision	Recall	F-Score	ROC Area
ELM	RBF	Batch	96.71%	0.1421	0.967%	0.967	0.967%	0.980
ELM	Sigmoid	Batch	96.64%	0.1432	0,967%	0.966	0.966%	0.979
OSELM	RBF	1 by 1	98.28%	0.1342	0.982%	0.983	0.983%	0.985
OSELM	Sigmoid	20 by 20	96.99%	0.1426	0.970%	0.970	0.970%	0.970
OSELM	**RBF**	**20 by 20**	**98.34%**	**0.1331**	**0.983%**	**0.984**	**0.983%**	**0.990**
OSELM	Sigmoid	1 by 1	96.81%	0.1429	0,969%	0.969	0.970%	0.969

Table 7. Comparison between algorithms in the NTC dataset

Classifier	**Properties**		**Classification Accuracy & Performance Metrics**					
	Activation Function	Learning Mode	ACC	RMSE	Precision	Recall	F-Score	ROC Area
ELM	RBF	Batch	99.15%	0.1027	0.991%	0.991	0.992%	0.991
ELM	Sigmoid	Batch	99.11%	0.1030	0,991%	0.990	0.990%	0.990
OSELM	RBF	1 by 1	99.51%	0.1006	0.995%	0.995	0.995%	0.995
OSELM	Sigmoid	20 by 20	99.68%	0.0990	0.996%	0.997	0.996%	0.996
OSELM	**RBF**	**20 by 20**	**99.72%**	**0.0982**	**0.998%**	**0.997**	**0.997%**	**0.997**
OSELM	Sigmoid	1 by 1	99.44%	0.1016	0.994%	0.994	0.994%	0.994

The Precision measure [64] shows what percentage of positive predictions where correct, whereas Recall [64] measures what percentage of positive events were correctly predicted. The F-Score [64] can be interpreted as a weighted average of the precision and recall. Therefore, this score takes both false positives and false negatives into account.

Table 8. Comparison between algorithms in the TTI dataset

Classifier	**Properties**		**Classification Accuracy & Performance Metrics**					
	Activation Function	Learning Mode	ACC	RMSE	Precision	Recall	F-Score	ROC Area
ELM	RBF	Batch	94.19%	0.1561	0.942%	0.942	0.942%	0.942
ELM	Sigmoid	Batch	94.10%	0.1570	0.941%	0.941	0.941%	0.941
OSELM	RBF	1 by 1	94.31%	0.1537	0.943%	0.943	0.943%	0.970
OSELM	Sigmoid	20 by 20	94.24%	0.1543	0.942%	0.943	0.943%	0.965
OSELM	**RBF**	**20 by 20**	**94.39%**	**0.1521**	**0.944%**	**0.944**	**0.944%**	**0.970**
OSELM	Sigmoid	1 by 1	94.28%	0.1539	0.943%	0.943	0.943%	0.943

Table 9. Comparison between algorithms in the DGA dataset

Classifier	**Properties**		**Classification Accuracy & Performance Metrics**					
	Activation Function	Learning Mode	ACC	RMSE	Precision	Recall	F-Score	ROC Area
ELM	RBF	Batch	92.17%	0.1877	0.920%	0.921	0.921%	0.975
ELM	Sigmoid	Batch	91.35%	0.2031	0.914%	0.914	0.914%	0.960
OSELM	RBF	1 by 1	92.89%	0.1804	0.930%	0.929	0.929%	0.978
OSELM	Sigmoid	20 by 20	93.13%	0.1726	0.932%	0.932	0.932%	0.982
OSELM	**RBF**	**20 by 20**	**93.97%**	**0.1711**	**0.940%**	**0.940**	**0.940%**	**0.985**
OSELM	Sigmoid	1 by 1	91.92%	0.2012	0.919%	0.919	0.920%	0.963

Intuitively it is not as easy to understand as accuracy, but F-Score is usually more useful than accuracy and it works best if false positives and false negatives have similar cost, in this case. Finally, the ROC [64] curve is related in a direct and natural way to cost/benefit analysis of diagnostic decision making.

This comparison generates encouraging expectations for the identification of the OS-ELM with RBF activation function and 20 by 20 learning mode [19],[20],[63], as a robust online classification model suitable for difficult problems.

According to this comparative analysis, it appears that CIPFaAPT is highly suitable method for applications with huge amounts of data such that traditional learning approaches that use the entire data set in aggregate are computationally infeasible. This algorithm successfully reduces the problem of entrapment in local minima in training process, with very fast convergence rates. These improvements are accompanied by high classification rates and low test errors as well. The performance of proposed model was evaluated in a high complex dataset and the real-world sophisticated scenarios. The experimental results showed that the OSELM with RBF activation function and 20 by 20 learning mode, has better generalization performance at a very fast learning speed and more accurate and reliable classification results. The final conclusion is that the proposed method has proven to be reliable and efficient and has outperformed at least for this security problem the other approaches.

Conclusion

This research effort, presented a timely, innovative, small footprint and highly effective security system which relies on advanced methods of computational intelligence and it greatly enhances the IT security mechanisms. It is a Real-time Computational Intelligence Protection Framework Against Advanced Persistent Threats, a next generation security platform that uses sophisticated analytics to monitor, track and classify risk across critical network infrastructures to identify APT. It performs classification by using an Online Sequential ELM with Gaussian RBF kernel and 20 by 20 learning mode, a very fast approach with high accuracy and generalization with minimum computational power and resources. The classification performance and the accuracy of the proposed model were experimentally explored based on several scenarios and reported very promising results. Moreover, CIPFaAPT is an effective system of network supervision, with capabilities of automated control. This is done to enhance the energetic security and the mechanisms of reaction of the general system, without special requirements.

The performance of the proposed system was tested on four novel datasets of high complexity, which emerged after extensive research of how the SSL protocol operates and after performing comparisons inspections and tests of independent variables which give the maximum precision, while requiring minimal computational resources.

Future research could involve its model under a hybrid scheme, which will combine semi supervised methods for the trace and exploitation of hidden knowledge between the inhomogeneous data that might emerge. Also, it would be important for the proposed framework to be expanded with automatic extraction methods of network traffic characteristics, so that it would fully automate the process of identifying malicious applications. Finally, the CIPFaAPT could be improved towards with other machine learning methods (unsupervised - competitive learning) or hybrid soft computing approaches (fuzzy-neural networks) and optimization algorithms aimed at even higher rates of correct classification.

References

[1] Raj, Vaishali S., Dr. R. Manicka Chezhian, M. Mrithulashri, (2014), Advanced Persistent Threats & Recent High Profile Cyber Threat Encounters, *International Journal of Innovative Research in Computer and Communication Engineering* (An ISO 3297: 2007 Certified Organization) Vol. 2, Issue 1, January 2014.

[2] Hutchins E., Cloppert M., and Amin R., (2010), Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains, In The 6th International Conference on Information-Warfare & Security. 2011. *Academic Conferences Ltd.*, 2010, pp. 113–125.

[3] Sood, Aditya K., Enbody, Richard J. (2013), Targeted Cyberattacks: A Superset of Advanced Persistent Threats". *IEEE Security & Privacy*, vol. 11, no. 1, pp. 54-61, Jan.-Feb. 2013, doi:10.1109/MSP.2012.90.

[4] Bencsáth B. , Pék G., Buttyán L., Félegyházi M., (2012), Duqu: Analysis, Detection,

and Lessons Learned, *CrySys Lab. in Proceedings of EuroSec* 2012, Bern, Switzerland, April 10, 2012.

[5] Virvilis N., Gritzalis D., Apostolopoulos T., (2013), Trusted Computing vs. Advanced Persistent Threats: Can a defender win this game?, in *Proc. of 10th IEEE International Conference on Autonomic and Trusted Computing (ATC-2013)*, pp. 396-403, IEEE Press, Italy, December 2013.

[6] Schiller, Craig A.; Binkley, Jim; Harley, David; Evron, Gadi; Bradley, Tony; Willems, Carsten; Cross, Michael (2007-01-01). *Botnets.* Burlington: Syngress. pp. 29–75. ISBN 9781597491358.

[7] Brezo F., Gaviria de la Puerta J., Ugarte-Pedrero X., Santos I., Bringas P. G., Barroso D.: Supervised classification of packets coming from a HTTP botnet, (2012), *Informatica*, XXXVIII Conferencia Latinoamericana En, Pages: 1 - 8, DOI: 10.1109/CLEI.2012.6427168.

[8] Stevanovic M., Pedersen J. M.: Machine learning for identifying botnet network traffic, (2013), *Technical report*, Aalborg Universitet, http://vbn.aau.dk/files/75720938/paper.pdf.

[9] Nazario J., Holz T.: As the net churns: Fast-?ux botnet observations, (2008), MALWARE '08, 3rd *International Conference on Malicious and Unwanted Software.*

[10] Perdisci R., Corona I., Dagon D., Lee W.: Detecting malicious flux service networks through passive analysis of recursive dns traces, (2009), in: *ACSAC '09, IEEE Computer Society*, Washington, DC, USA, 2009, pp. 311–320. doi:10.1109/ACSAC.2009.36.

[11] Bailey M., Cooke E., Jahanian F., Xu Y., Karir M.: A survey of botnet technology and defenses, in: (2009), *Cybersecurity Applications Technology*, pp. 299–304.

[12] Feily M., Shahrestani: A survey of botnet and botnet detection, Emerging Security Information, (2009), *SECURWARE '09*. 268–273. doi:10.1109/SECURWARE.2009.48.

[13] www.damballa.com.

[14] www.crowdstrike.com.

[15] *DGAs and Cyber-Criminals: A Case Study, Research Note*, www.damballa.com.

[16] Hayes, Jamie. *Traffic Confirmation Attacks Despite Noise.* arXiv preprint arXiv:1601.04893 (2016).

[17] Backes, Michael, et al. "Provably secure and practical onion routing." *Computer Security Foundations Symposium (CSF), 2012 IEEE 25th. IEEE*, 2012.

[18] Bansal, Deepika, Priya Sethi, and Shipra Kataria. "Secure Socket Layer and its Security Analysis." *Networking and Communication Engineering* 7.6 (2015): 255-259.

[19] Liang N.-Y., Huang G.-B., Saratchandran P., Sundararajan N.: A Fast and Accurate On-line Sequential Learning Algorithm for Feedforward Networks, (2006), *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1411-1423.

[20] Huang G.-B. , Liang N.-Y., Rong H.-J., Saratchandran P., Sundararajan N.: *On-line sequential extreme learning machine*, (2005), IASTED.

[21] Demertzis K., Iliadis L., (2015), Intelligent Bio-Inspired Detection of Food Borne Pathogen by DNA Barcodes: The case of Invasive Fish Species Lagocephalus Sceleratus, *Engineering Applications of Neural Networks*, Vol 517 pp 89-99, DOI 10.1007/978-3-319-23983-5_9.

[22] Demertzis K., Iliadis L. (2014). A Hybrid Network Anomaly and Intrusion Detection Approach Based on Evolving Spiking Neural Network Classification. In: E-Democracy, Security, Privacy and Trust in a Digital World. *Communications in Computer and Information Science*, 441, 11-23. doi:10.1007/978-3-319-11710-2_2.

[23] Demertzis K., Iliadis L. (2014). Evolving Computational Intelligence System for Malware Detection, In: *Advanced Information Systems Engineering Workshops, Lecture Notes in Business Information Processing*, 178, 322-334. doi: 10.1007/978-3-319-07869-4_30.

[24] Demertzis K., Iliadis L. (2014, April). Bio-Inspired Hybrid Artificial Intelligence Framework for Cyber Security. *Springer Proceedings 2nd Conference on CryptAAF: Cryptography Network Security and Applications in the Armed Forces*, Springer, Athens, 161-193. doi: 10.1007/978-3-319-18275-9_7.

[25] Demertzis K., Iliadis L. (2014, November). Bio-Inspired Hybrid Intelligent Method for Detecting Android Malware, *Proceedings of the 9th KICSS 2014*, Knowledge Information and Creative Support Systems, Cyprus, 231-243. ISBN: 978-9963-700-84-4, 2014.

[26] Demertzis K., Iliadis L. (2015, April). Evolving Smart URL Filter in a Zone-based Policy Firewall for Detecting Algorithmically Generated Malicious Domains. *Proceedings SLDS* (Statistical Learning and Data Sciences) Conference LNAI (Lecture Notes in Artificial Intelligence) 9047 Springer, Royal Holloway University London, UK, 223-233. doi: 10.1007/978-3-319-17091-6_17.

[27] Demertzis K., Iliadis L. (2015, September). SAME: An Intelligent Anti-Malware Extension for Android ART Virtual Machine. *Proceedings of the 7th International Conference ICCCI 2015*, Lecture Notes in Artificial Intelligence LNAI 9330, Madrid, Spain, 235-245. doi: 10.1007/978-3-319-24306-1_23.

[28] Demertzis K., Iliadis L. (2016), *Computational Intelligence Anti-Malware Framework for Android OS, Special Issue on "Vietnam Journal of Computer Science (VJCS)"*, Springer, DOI 10.1007/s40595-017-0095-3.

[29] Demertzis K., Iliadis L. (2016), Detecting Invasive Species with a Bio-Inspired Semi Supervised Neurocomputing Approach: The Case of Lagocephalus Sceleratus, *Special issues Neural Computing and Applications Journal* by Springer, DOI :10.1007/s00521-016-2591-2.

[30] Demertzis K., Iliadis L. (2016), SICASEG: A Cyber Threat Bio-Inspired Intelligence Management System, *Journal of Applied Mathematics & Bioinformatics*, vol.6, no.3, 2016, 45-64, ISSN: 1792-6602 (print), 1792-6939 (online), Scienpress Ltd, 2016.

[31] Bougoudis I., Demertzis K., Iliadis L., (2016), Fast and Low Cost Prediction of Extreme Air Pollution Values with Hybrid Unsupervised Learning, *Integrated Computer-Aided Engineering*, vol. 23, no. 2, pp. 115-127, 2016, DOI: 10.3233/ICA-150505, IOS Press.

[32] Bougoudis I., Demertzis K., Iliadis L., (2016), HISYCOL a Hybrid Computational Intelligence System for Combined Machine Learning: The case of Air Pollution Modeling in Athens, *EANN Neural Computing and Applications* pp 1-16, DOI 10.1007/s00521-015-1927-7.

[33] Anezakis V. D., Demertzis K, Iliadis L, Spartalis S. (2016a) A hybrid soft computing approach producing robust forest fire risk indices. *IFIP Advances in Information and Communication Technology*, AIAI September 2016, Thessaloniki Greece, 475:191-203.

[34] Anezakis V. D., Dermetzis K, Iliadis L, Spartalis S. (2016b) Fuzzy cognitive maps for long-term prognosis of the evolution of atmospheric pollution, based on climate change scenarios: The case of Athens. *Lecture Notes in Computer Science including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*, 9875:175-186. doi: 10.1007/978-3-319-45243-2_16.

[35] Bougoudis I, Demertzis K, Iliadis L, Anezakis V. D., Papaleonidas A. (2016b) Semi-supervised hybrid modeling of atmospheric pollution in urban centers. *Communications in Computer and Information Science*, 629:51-63.

[36] Demertzis K., Iliadis L. (2016), Adaptive Elitist Differential Evolution Extreme Learning Machines on Big Data: Intelligent Recognition of Invasive Species, International Neural Network Society Conference on Big Data (INNS Big Data 2016), Thessaloniki, Greece 23-25 October 2016. *Proceedings, Advances in Big Data Volume 529 of the series Advances in Intelligent Systems and Computing* pp 333-345, DOI:10.1007/978-3-319-47898-2_34, Springer.

[37] Wright M. K., Adler M., Levine B. N. , and Shields C., "An analysis of the degradation of anonymous protocols," in *Proceed. of the Network and Distributed Security Symposium*, 2002.

[38] Shmatikov V. and Wang M. H., "Timing analysis in low-latency mixnetworks: Attacks and defenses," in *Proceedings of ESORICS* 2006.

[39] Cheng C., Peng T. W., Guang-Bin H.: Extreme learning machines for intrusion detection, (2012), *IJCNN, International Joint Conference*, DOI: 10.1109/IJCNN.2012.6252449.

[40] Hsu C.-H., Huang C.-Y., Chen K.-T.: Fast-?ux bot detection in real time, (2010), in *13th international conference on Recent advances in intrusion detection*, ser. RAID'10.

[41] Haffner P., Sen S., Spatscheck O., Wang D.: ACAS: Auto-mated Construction of Application Signatures, (2005), *Proceedings of the ACM SIGCOMM*, pp.197-202.

[42] Alshammari R., Zincir-Heywood N. A.: A ?ow based approach for SSH traffic detection, (2007), Cy-bernetics, *ISIC. IEEE International Conference on*, pp.296-301.

[43] Nhauo D. Sung-Ryul K.: Classification of Malicious Domain Names using Support Vector Machine and Bi-gram Method, (2013), *J. of Security and Its Applications* Vol. 7, No. 1.

[44] Antonakakis M., Perdisci R., Nadji Y., Vasiloglou N., Abu S., Lee W. , Dagon D.: *From Throw-Away traffic to Bots: Detecting the Rise of DGA-Based Malware*, (2012).

[45] Zhao D., Traore I., Sayed B., Lu W., Saad S., Ghorbani A.: Botnet detection based on traffic behavior analysis and low intervals (2013), *J. Computer Security* 39 2e16.

[46] Chakravarty S., Barbera M. V., Portokalidis G., Polychronakis M., Keromytis A. D., (2014), On the Effectiveness of traffic Analysis Against Anonymity Networks Using Flow Records, *Proceedings on 15th International Conference*, PAM 2014, pp 247-257, Springer.

[47] Almubayed A., Hadi A., Atoum J. (2015), A Model for Detecting Tor Encrypted Traffic using Supervised Machine Learning, *I. J. Computer Network & Information Security*, 7, 10-23.

[48] Chaabane A., Manils P., Kaafar M. A., (2010), Digging into Anonymous Traffic: A Deep Analysis of the Tor Anonymizing Network, *4th International Conference on Network and System Security (NSS)*, p. 167 - 174.

[49] Chakravarty S., Stavrou A., and Keromytis A. D., (2010), Traffic analysis against low-latency anonymity networks using available band width estimation, *Proceedings of the 15th European conference on Research in computer security*, ESORICS'10. Springer, pp. 249–267.

[50] Chakravarty S., Stavrou A., and Keromytis A. D., "Identifying Proxy Nodes in a Tor Anonymization Circuit," in *Proceedings of the 2ndWorkshop on Security and Privacy in Telecommunications and Information Systems* (SePTIS), December 2008, pp. 633–639.

[51] http://malware-traffic-analysis.net/).

[52] Haining W., Danlu Z., Kang G. S., (2002), Detecting SYN flooding attacks, *Proceedings on INFOCOM 2002*, Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, vol 3, pp 1530-1539.

[53] http://www.netresec.com/?page=PcapFiles.

[54] Arndt, D. J., Zincir-Heywood, A. N. (2011), A Comparison of Three Machine Learning Techniques for Encrypted Network Traffic Analysis, Computational Intelligence for Security and Defense Applications *(CISDA)*, 2011 IEEE Symposium on pp. 107 – 114.

[55] http://contagiodump.blogspot.gr/.

[56] Iliadis L. *Intelligent Information Systems and applications in risk estimation*, (2008), ISBN: 978-960-6741-33-3 A. Stamoulis publication, Thessaloniki, Greece.

[57] Bailey M., Oberheide J., Andersen J., Mao Z. M., Jahanian F., Nazario J.: Automated classification and analysis of internet malware., (2007), in: C. KrÃijgel, R. Lippmann, A. Clark (Eds.), *RAID*, Vol. 4637 of Lecture Notes in Computer Science, Springer, pp. 178–197.

[58] Desai A., Jadav P. M., (2012), An Empirical Evaluation of Adaboost Extensions for Cost-Sensitive Classification, *International Journal of Computer Applications*, Vol 44, No 13.

[59] http://www.alexa.com/.

[60] http://www.malwaredomains.com/.

[61] https://www.clicksecurity.com/.

[62] Upton, G., Cook, I. *Understanding Statistics*, (1996) Oxford University Press. p. 55.

[63] Cambria E., Guang-Bin H.: Extreme Learning Machines, (2013), *IEEE InTeLLIGenT SYSTemS,* 541-1672/13.

[64] Powers, David M. W. (2011), Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation, *Journal of Machine Learning Technologies*, 2, 37-63.