**2016 8th International Conference on Cyber Conflict**
Cyber Power
N.Pissanidis, H.Rõigas, M.Veenendaal (Eds.)
2016 © NATO CCD COE Publications, Tallinn



# Countering Advanced Persistent Threats through Security Intelligence and Big Data Analytics

**Mirco Marchetti**
Department of Engineering 'Enzo Ferrari'
University of Modena and Reggio Emilia
Modena, Italy
mirco.marchetti@unimore.it

**Fabio Pierazzi**
Department of Engineering 'Enzo Ferrari'
University of Modena and Reggio Emilia
Modena, Italy
fabio.pierazzi@unimore.it

**Alessandro Guido**
Department of Engineering 'Enzo Ferrari'
University of Modena and Reggio Emilia
Modena, Italy
alessandro.guido@unimore.it

**Michele Colajanni**
Department of Engineering 'Enzo Ferrari'
University of Modena and Reggio Emilia
Modena, Italy
michele.colajanni@unimore.it

**Abstract:** Advanced Persistent Threats (APTs) represent the most challenging threats to the security and safety of the cyber landscape. APTs are human-driven attacks backed by complex strategies that combine multidisciplinary skills in information technology, intelligence, and psychology. Defending large organisations with tens of thousands of hosts requires similar multi-factor approaches. We propose a novel framework that combines different techniques based on big data analytics and security intelligence to support human analysts in prioritising the hosts that are most likely to be compromised. We show that the collection and integration of internal and external indicators represents a step forward with respect to the state of the art in the field of early detection and mitigation of APT activities.

**Keywords:** *Advanced Persistent Threats, big data analytics, security intelligence, social engineering*

# 1. INTRODUCTION

The majority of cyber-attacks rely on automated scanning and exploitation of known vulnerabilities over large sets of targets. APTs represent a more dangerous category because they are sophisticated human-driven attacks against specific targets. Objectives of APT attacks include continuous exfiltration of information, cyber warfare, damage to critical infrastructure, and degradation of military assets (Data Breaches, 2016). They are typically perpetrated over long periods of time by groups of experts that leverage open source intelligence and social engineering techniques (Molok, Chang, & Ahmad, 2010), vulnerabilities not always known to the public (Jeun, Lee, & Won, 2012; Virvilis, Serrano, & Vanautgaerden, 2014), standard protocols, encrypted communications, and zero-day vulnerabilities to evade detection (Brewer, 2014). Consequently, traditional defensive solutions such as antiviruses and signature-based detection systems (Sabahi & Movaghar, 2008; Zhou, Leckie, & Karunasekera, 2010) that can identify standard malware are ineffective against APTs.

We claim that effective defences require analogous multi-factor approaches where human analysts must be supported by big data analytic techniques that are able to detect and prioritise weak signals related to APT activities. The proposed AUSPEX[1] framework follows these principles. It gathers and combines *internal information* from network probes located in an organisation, and *external information* from public sources such as the web, social networks, and blacklists. From these data, AUSPEX calculates two sets of indicators:

1. *compromise indicators*, that prioritise internal clients based on their suspicious network activities; and
2. *exposure indicators*, that estimate the likelihood of a social engineering or intelligence attack.

The final output of AUSPEX is a list of internal hosts ranked by *compromise* and *exposure* scores. In this version, AUSPEX focuses on client hosts that are likely the initial targets of APTs.

Papers related to APTs usually focus on the most popular attacks (Brewer, 2014; Jeun, Lee, & Won, 2012; Virvilis & Gritzalis, 2013) and identify the main phases of an APT without proposing detection approaches. Other works (Bhatt, Toshiro Yano, & Gustavsson, 2014; Giura & Wang, 2012; De Vries, Hoogstraaten, van den Berg, & Daskapan, 2012; Hutchins, Cloppert, & Amin, 2011) formalise the APT defence problem, but they only propose development guidelines and leave the definition of detection rules and analysis to future work. To the best of our knowledge, AUSPEX is the first framework that supports human analysts to detect and mitigate APTs in large organisations by prioritising weak signals through a combination of internal and external data.

1 An *auspex* was an interpreter of omens in ancient Rome.

The remainder of the paper is organised as follows. Section 2 introduces the scenario and main challenges related to APT identification. Section 3 presents an overview of the AUSPEX framework. Sections 4 and 5 discuss *compromise* and *exposure* indicators, respectively. Section 6 shows how hosts are ranked, section 7 compares AUSPEX with related literature, and section 8 draws conclusions and outlines future work.

## 2. ADVANCED PERSISTENT THREATS

A typical APT attack comprises five main phases (Brewer, 2014): reconnaissance; compromise; maintaining access; lateral movement; and data exfiltration.

In the *reconnaissance* phase, an attacker carries out intelligence analysis on the target organisation to extract information and identify weak spots (Lindamood, Heatherly, Kantarcioglu, & Thuraisingham, 2009; Irani, Webb, Pu, & Li, 2011). This phase involves both social and technological aspects.

In the *compromise* phase, an APT attacker infiltrates the system, possibly through social engineering strategies that exploit information gathered during the reconnaissance phase. Often, this phase involves infected files sent as email attachments or links (Data Breaches, 2016). The final goal is to install a RAT (Remote Access Trojan, or Remote Administration Tool) on at least one host of the organisation.

In the *maintaining access* phase, an attacker uses the RAT to communicate with an external *Command and Control* (CnC) server (Bailey, Cooke, Jahanian, Xu, & Karir, 2009). An internal host of the organisation initiates this communication, because outgoing traffic passes more easily through firewalls.

The *lateral movement* phase has two main purposes: to shift towards other internal hosts of the organisation that have more access privileges or intrinsic value; and to move data to a drop zone, such as a web server, that allows information exfiltration while minimising risks of detection.

Finally, in the *data exfiltration* phase, the attacker uploads data to an external server, either in a single burst or slowly over several days. The attacker may also use legitimate external servers as drop zones, such as cloud hosts.

The challenge of APT detection is inherent to the combined use of different attack vectors and evasion strategies. Therefore, defences cannot be purely technological and must be based on a combination of human analysis and automatic detection of weak signals likely characterising an APT. An overview of the proposed defensive approach is described in the following section.

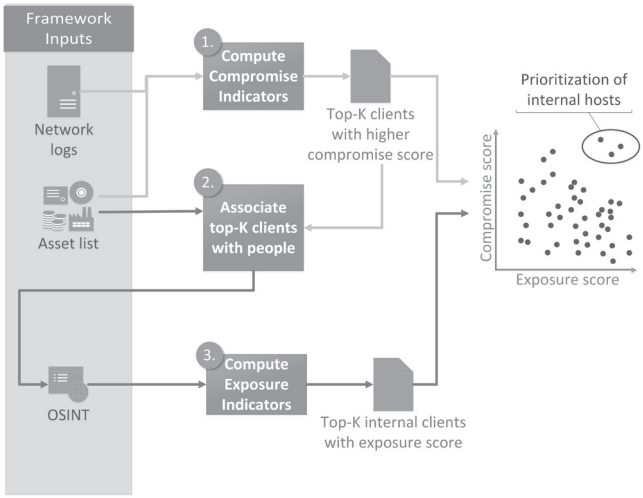
# 3. OVERVIEW OF THE DEFENSIVE METHOD

The main purpose of AUSPEX is to prioritise the internal clients of the organisation that are most likely compromised by an APT. To this end, it combines weak signals derived by security sensors with external information related to employees who may be victims of social engineering attacks. Figure 1 shows an overview of the main AUSPEX components.

The input data gathered and analysed by AUSPEX are shown on the left column of Figure 1:

- **Network logs** (e.g., network flows, requests to web servers, requests to name servers, security alerts) collected through SIEM and intrusion detection systems (Denning, 1987; Paxson, 1999). This data allows the identification of weak signals possibly corresponding to APT activities.
- A simplified **assets list** that maps the members of the organisation and their client devices. This information is useful to *link* technological and open source data.
- **OSINT information** collected from public open sources. This is used to identify and quantify the information that is available to APT attackers.

**FIGURE 1:** AUSPEX OVERVIEW




AUSPEX adopts a network-centric approach because network traffic can be collected and analysed more easily than host-based logs (Friedberg, Skopik, Settanni, & Fiedler, 2015) (e.g., OS system calls), especially in large and dynamic organisations comprising heterogeneous hardware and software components. Hence, as a first step (box 1 in Figure 1) AUSPEX analyses network logs collected within the organisation to evaluate a set of compromise indicators for each internal client. For example, it is possible to estimate the probability of data exfiltration by analysing outgoing traffic of each internal host. Then, an overall *compromise score* is calculated from the set of indicators. This part is described in Section 4. The clients with higher

compromise scores (top-K clients) are selected for further analysis on external sources. To this end, AUSPEX uses the list of the top-K clients and the asset list to identify users of the most likely compromised clients (box 2 in Figure 1). For each of these users AUSPEX calculates a set of *exposure indicators* (box 3 in Figure 1) by crawling open sources to understand whether these users might be likely victims of social engineering attacks (Molok, Chang, & Ahmad, 2010). This part is described in section 5.

Computation of the exposure indexes is only performed for the top-K clients with higher compromise scores to reduce the volume of data. Modern organisations usually have thousands of internal devices; hence, there is a need to narrow the scope of an analysis that can be repeated daily. Since the amount of open source data is theoretically unrestrained, focusing attention only on the top-K clients also makes crawling open sources feasible and more effective. Moreover, generalised crawling and inspection of employee information is likely to raise issues related to labour and privacy laws in many countries. Focused analyses on employees using likely compromised hosts will reduce legal risks.

The final output of AUSPEX is a list of internal clients and overall compromise and exposure scores. By prioritising hosts in which both scores are high, the security analysts can focus attention on a limited subset of internal clients.

# 4. COMPROMISE INDICATORS

Despite the huge challenge of detecting human-driven targeted APTs in large networks, it is possible to define automatic analyses to support security analysts in prioritising events possibly related to APTs (Brewer, 2014; Virvilis & Gritzalis, 2013). To this end, AUSPEX adopts algorithms targeted to prioritise internal clients possibly involved in *maintaining access, lateral movement*, and *data exfiltration* phases of an APT.

## *4.1 Maintaining access*

After initial compromise, an APT attacker deploys a RAT (Remote Administration Tool) on one or more hosts of the organisation. This RAT tries to contact one or more external CnC servers controlled by the APT attacker, hence the first goal is to detect communications towards these CnC servers. To achieve this, AUSPEX analyses communications between internal and external hosts to identify suspicious external hosts and evaluate how many flows exist between them and internal clients.

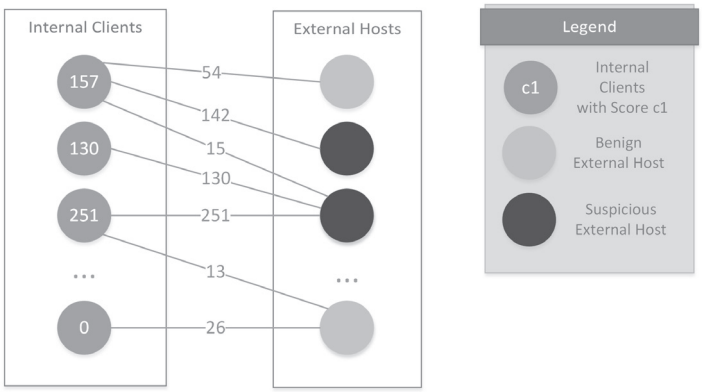
AUSPEX analyses network logs and builds an undirected bipartite graph where internal and external hosts are represented as two sets of vertices. Edges between vertices have weights that correspond to the number of flows between internal and external hosts. It then labels each external host as benign or suspicious, relying on a combination of algorithms including blacklist filtering, DGA analysis, regular access patterns, and non-matching flows (Bailey, Cooke, Jahanian, Xu, & Karir, 2009; Bilge, Balzarotti, Robertson, Kirda, & Kruegel, 2012; Schiavoni, Maggi, Cavallaro, & Zanero, 2014).

Finally, AUSPEX calculates a *CnC* compromise indicator $c_1^h$ for each internal client *h* that corresponds to the *sum of weights* of the edges that connect *h* with external hosts. The subscript 1 indicates that this is the first of three compromise indicators. Other two indicators related to the *lateral movement* and *data exfiltration* phases are presented in sections 4.2 and 4.3. The score $c_1^h$ estimates the likelihood that an internal host *h* is involved in CnC communications.

A simplified example of the proposed algorithm is shown in Figure 2, where internal clients (on the left) communicate with one or more external hosts (on the right). We observe that the internal client at the bottom has score $c_1^h$=0 because it is communicating with a benign external host.

AUSPEX marks an external host as suspicious if it satisfies at least one of the four criteria (Bilge, Balzarotti, Robertson, Kirda, & Kruegel, 2012; Bailey, Cooke, Jahanian, Xu, & Karir, 2009), *blacklist filtering, DGA analysis, regular access patterns, and non-matching flows.*

**FIGURE 2:** EXAMPLE FOR THE ALGORITHM FOR CNC COMMUNICATION RANKING




**Blacklist filtering.** Several public sites offer reputation scores for IP addresses and domain names that represent their likelihood of being malicious. Examples are *Malware Domain List*,[2] *WhatIsMyIP Blacklist Check*,[3] *Google Safe Browsing*,[4] and *Web of Trust*.[5] AUSPEX calculates a reputation score through the equation proposed in (Bilge, Balzarotti, Robertson, Kirda, & Kruegel, 2012).

**DGA analysis.** Domain-Generation Algorithms are often adopted by attackers to simplify communications between RATs and CnCs (Bailey, Cooke, Jahanian, Xu, & Karir, 2009). To determine whether a domain name is DGA-generated, AUSPEX adopts a simplified version of the algorithm proposed in (Schiavoni, Maggi, Cavallaro, & Zanero, 2014). The main intuition is

2 Malware Domain List homepage, [Online]. Available: http://www.malwaredomainlist.com, [Accessed 13.04.2016].
3 WhatIsMyIP.com's Blacklist Check, [Online]. Available: https://www.whatismyip.com/blacklist-check, [Accessed 13.04.2016].
4 Safe Browsing in Google's Transparency Report, [Online]. Available: https://www.google.com/transparencyreport/safebrowsing, [Accessed 13.04.2016].
5 Web of Trust homepage, [Online]. Available: https://www.mywot.com, [Accessed 13.04.2016].

that a domain name is likely benign (not a DGA) if it is composed of meaningful words (those in the English lexicon) and is pronounceable. Hence, AUSPEX calculates two main metrics for each domain *d:* the *meaningful characters ratio R(d)* and the *n-grams normality score* $S_n(d)$.

Then, a feature vector $f(d)=[R,S_1,S_2,S_3]$ is associated with each domain *d*. The training of benign domains is done on the Alexa top 100,000 sites list, where a centroid $\bar{x}$ of the feature space is determined. Then, a domain is marked as DGA if the *Mahalanobis distance* (Bishop, 2006) between a feature *f(d)* and the centroid $\bar{x}$ of the benign feature space is higher than a threshold, Λ, defined as the *p*- percentile of the distance vectors values from the centroid. Based on our experience and the literature (Schiavoni, Maggi, Cavallaro, & Zanero, 2014), we suggest setting Λ=0,9.

**Regular access patterns.** If access patterns between an external host and one or more internal hosts are too regular, they likely correspond to automatic communications. As proposed in (Bilge, Balzarotti, Robertson, Kirda, & Kruegel, 2012), AUSPEX calculates a *set of inter-arrival sequences* $I(h_{ext})$ as the union of the differences between the timestamps of contiguous flows between $h_{ext}$ and each internal client:

$$I(h_{ext}) = \bigcup_{int} \left( \bigcup_{k=1}^{K_{int,ext}} (ts_{int,ext,k} - ts_{int,ext,k-1}) \right)$$

where $K_{int,ext}$ is the number of flows between the external host $h_{ext}$ and internal host $h_{int}$, and $ts_{int,ext,k}$ is the value of the timestamp corresponding to flow number *k* identified between $h_{ext}$ and $h_{int}$ . An external host is marked as suspicious if there is an inter-arrival difference value $\bar{n}$ with probability higher than a threshold $\bar{P}$ (meaning that too many communications occur at regular intervals of duration $\bar{n}$). Based on our test deployments in real and large network environments, we recommend initially setting $\bar{P}$=0,95, that is, a probability of 0.95.

**Non-matching flows.** If there is an imbalanced number of flows between an external host and one or more internal hosts $h^1_{int}$ , $h^2_{int}$ , ... , $h^N_{int}$ , it is possible that an external CnC is no longer reachable, and one or more internal clients are still trying to contact it. To quantify this, AUSPEX adapts a metric initially proposed in (Bilge, Balzarotti, Robertson, Kirda, & Kruegel, 2012). An external host $h_{ext}$ is marked as suspicious if it has responded to less than 50% of the flows.

## *4.2 Lateral movement*

In the *lateral movement* phase, an APT attacker that has infiltrated an organisation's network tries to gain access to other internal hosts (clients or servers) to improve his chances of persistence and to acquire greater privileges for accessing resources of interest.

To identify revealing signals of ongoing lateral movements, AUSPEX analyses internal communications, defined as any network activity or interaction occurring between two internal

hosts. Internal hosts of an organisation belong to two groups: clients assigned to employees, and servers (e.g., a Network Attached Storage or an HTTP server). Since AUSPEX focuses on prioritising clients, we only analyse client-to-server and client-to-client communications.

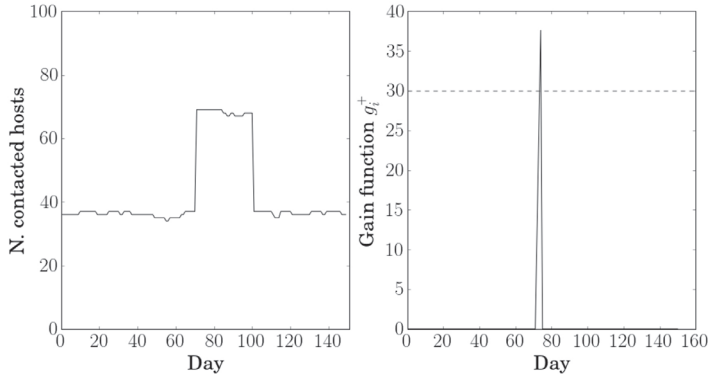
We analysed the network traffic generated within the internal network of a real-world organisation. Our analyses confirmed that the number of packets exchanged between internal hosts has a high variance, and greatly depends on human activities. We also observed that for each internal client $h$, the number of internal hosts contacted by is stable, because a client tends to communicate with a stable set of internal hosts. Hence, to identify possible lateral movements, for each internal client $h$ AUSPEX monitors the number of internal hosts (both clients and servers) that have communicated with $h$ in the recent past. Significant changes in this value imply that $h$ is contacting many new internal hosts, an activity that is likely related to lateral movements. To this purpose, AUSPEX calculates the time series of the number of internal hosts contacted by $h$ in a sliding time window $\Delta$. This time series is denoted by $D_t^h$. By monitoring state-changes (Montgomery, 1991) in $D_t^h$, AUSPEX is able to detect internal clients with sudden increases in the number of contacted hosts.

For example, let us consider an internal client $h$ at day $t$, that in the previous window $\Delta$=15 days communicated with the set of 5 hosts $\{h_1, h_2, h_3, h_4, h_5\}$. Let us suppose that $t$ at day $t$+1 contacted the set of hosts $\{h_4, h_5, h_6, h_7, h_8\}$. Although the internal hosts are still five in total, there are three new internal hosts ($h_6$, $h_7$, $h_8$) that $h$ did not contact in the previous window. Hence $D_{t+1}^h=5+3=8$, and a state-change can be detected since the number of hosts contacted by $h$ has almost doubled in one day. AUSPEX adopts a CUSUM-based state-change detection algorithm that monitors the *mean* of the series of contacted hosts (Montgomery, 1991). This family of state-change detectors is applicable to series with low variability and is computationally feasible even for online detection contexts. For each internal host, AUSPEX calculates a *lateral movement* indicator $c_2^h$ as:

$$c_2^h \text{ as: } = \Sigma_t \, g_t^+$$

where $g_t^+$ is a gain function higher than 0 if a positive state-change is detected at time $t$, and estimates the entity of the state-change (Casolari, Tosi, & Lo Presti, 2012). An example of state-change observed in a segment of an internal network environment is proposed in Figure 3, where Figure 3a shows an example of the time series $D_t^h$, and Figure 3b shows the magnitude of the state-change when it is detected around day 70, with $\Delta$=30 days. High values of $c_2^h$ imply that internal client $h$ has frequently or significantly increased the number of internal hosts that it has contacted.

**FIGURE 3:** EXAMPLE OF STATE-CHANGE DETECTION THROUGH THE CUSUM-BASED ALGORITHM




## *4.3 Data exfiltration*

As a final step of an APT campaign oriented to data exfiltration, the attacker must send confidential data from the target organisation to one or more remote servers. Some recent and popular examples of data exfiltration are: the *Adobe* leak in 2013, comprising 9GB of encrypted password; the *Ashley Madison* leak in 2015, in which their entire database of about 30GB was stolen (Data Breaches, 2016); and the *Hacking Team* data leak, including 400GB of corporate data.

To identify internal hosts possibly involved in data exfiltration, AUSPEX focuses on the analysis on outgoing traffic (Brewer, 2014). For each internal client *h*, AUSPEX calculates a feature vector $x_t=(x_t^1, x_t^2, x_t^3)$, with the following three components: *outbytes* ( $x_t^1$ ), which captures deviations in the number of bytes uploaded by to external hosts; and *numdst* ( $x_t^2$ ), and *numconns* ( $x_t^3$ ), which identify variations in the number of destinations and connections to external hosts. The choice of the right time granularity *t* for the feature vectors $x_t$ is context-dependent (Brockwell & Davis, 2013; Pierazzi, Casolari, Colajanni, & Marchetti, 2016). In general, we recommend a time granularity of 1 day because daily aggregation reduces noise related to normal use of clients and servers. It also allows security analysts to investigate suspicious activities on a daily basis.

AUSPEX then quantifies the suspiciousness of outgoing traffic for each internal host with respect to other internal hosts and their past behaviour. A commonly adopted method to estimate the variation of behaviour is to consider the movement vector as a Euclidean difference in the feature space (Bishop, 2006). To consider the past behaviour of an internal host, AUSPEX also calculates the centroid of its past positions in a time window *W* as follows:

$$\beta_t(W) = \left(\frac{\sum_j x_j^1}{W}, \frac{\sum_j x_j^2}{W}, \frac{\sum_j x_j^3}{W}\right), j \in \{t-W-1, \ldots, t-1\}$$

where the three components of $\beta_t\ (W)$ correspond to the mean of the last $W$ values of the three components of the feature vector $x_t$. This metric represents an average of the history of uploaded bytes, number of connections, and number of destinations contacted by an internal client. Finally, for each internal client AUSPEX calculates a compromise indicator $c_3^h$ as the magnitude of the movement vector $m_t$ (Bishop, 2006):

$$c_3^h = ||m_t|| = \sqrt{\sum_{i=1}^{N} \left( \frac{x_t^i - \beta_t^i(W)}{\beta_t^i(W)} \right)^2}$$

where $m_t$ quantifies changes in the outgoing traffic statistics. High values of $c_3^h$ imply suspicious uploads that differ significantly with respect to past behaviour.

# 5. EXPOSURE INDICATORS

Clients with higher compromise scores (top-K clients) are selected for further analysis based on external sources. The goal is to verify whether employees that are likely victims of social engineering attacks use these clients. To this purpose, AUSPEX calculates a set of *exposure indicators* for the employees that use likely-compromised clients by analysing information collected from public and open sources that are also available to an APT attacker. In the present version, AUSPEX considers three popular social networks (Facebook, Twitter, and LinkedIn) that allow fee-based APIs to social profiles and pages. Twitter exposes REST APIs[6] to search for users by name, and to gather information about users, their tweets and their followers. LinkedIn offers APIs to search for people by name[7] or company,[8] and access all metadata, profile information, and public posts of a person. Facebook uses a proprietary query language through the search box of its website. To leverage this interface, AUSPEX adopts Selenium,[9] a software library that handles browser interactions programmatically. In particular, it gets Facebook user IDs through their email address by means of the search bar, and then crawls public data related to user profiles, such as posts and friends lists.

For the employees of the organisation that are using likely compromised clients, AUSPEX calculates a set of exposure indicators that determine the likelihood of a social engineering attack: social activity; social connections; personal information leakage; and organisation information leakage.

The **social activity** indicator $e_1$ quantifies how much an employee is active on social networks (Romero, Galuba, Asur, & Huberman, 2011; Montangero & Furini, 2015; Canali, Casolari, & Lancellotti, 2010). Employees with higher social activity are more likely to become victims of social engineering attacks, because they may be approached by an APT attacker or may reveal

6 Twitter REST APIs, [Online]. Available: https://dev.twitter.com/rest/public. [Accessed 13.04.2016].
7 LinkedIn Profile API for developers, [Online]. Available: https://developer-programs.linkedin.com/documents/profile-api [Access to this resource requires a valid LinkedIn account.].
8 LinkedIn People Search API for developers, [Online]. Available: https://developer-programs.linkedin.com/documents/people-search-api. [Access to this resource requires a valid LinkedIn account.].
9 SeleniumHQ homepage, [Online]. Available: http://www.seleniumhq.org/. [Accessed 13.04.2016].

sensitive information. AUSPEX evaluates this indicator by considering the average number of posts (i.e., any form of user-generated content posted on the social network) per day on Facebook, Twitter, and LinkedIn.

The **social connections** indicator $e_2$ quantifies how much an employee is connected to other employees of the organisation on social networks. This information is useful to an attacker because employees connected to many other members of the organisation are likely to know information related to confidential projects. It might also be easier for an APT attacker to propagate through lateral movement to other hosts, such as through infected email attachments. AUSPEX performs this by determining for each employee the number of connections on Facebook, LinkedIn, and Twitter who work for the same organisation.

The **personal information leakage** indicator $e_3$ corresponds to the number of personal fields that the employee filled in his online social network profiles. The main motivation is that an employee might reveal personal information that attackers may leverage to carry out a more effective social engineering activity. The problem of personal information leakage has already been studied in the literature (Molok, Chang, & Ahmad, 2010; Irani, Webb, Pu, & Li, 2011; Lindamood, Heatherly, Kantarcioglu, & Thuraisingham, 2009; Lam, Chen, & Chen, 2008). AUSPEX considers the following fields of Twitter, Facebook and LinkedIn profiles: name, location, sex, relationship status, hometown, homepage, birthdate, 'about me', groups, interests, liked pages.

The **organisation information leakage** indicator $e_4$ quantifies the amount of information about the organisation that an employee has published on open sources, and that is publicly available. This is relevant because an APT attacker may select his target based on such information (e.g., target an employee that is working on a specific project). AUSPEX adopts a variation of the score defined in (Irani, Webb, Pu, & Li, 2011) that considers a set of *keywords* that refer to activities of the organisation, such as projects, customers, or suppliers. Each keyword has a risk level, representing its severity. Risk levels can be assigned according to risk assessment best practices (Ostrom & Wilhelmsen, 2012). Examples of risk levels related to a project name are proposed in Table 1.

**TABLE 1:** EXAMPLE OF RISK LEVELS RELATED TO KEYWORDS

| Risk level | Explanation |
|---|---|
| 1 [low] | Public mentions of the project do not affect the organisation. However, the knowledge that an individual is working on this project may increase the chances for an APT attacker to target him. |
| 2 [medium] | The project is known by the members of the organisation, but is not publicly disclosed on the outside. |
| 3 [high] | This project is known only to some members of the organisation. Its diffusion may have moderate legal and economic consequences. |
| 4 [critical] | This project is extremely critical and confidential. Its diffusion may have severe legal and economic consequences. |

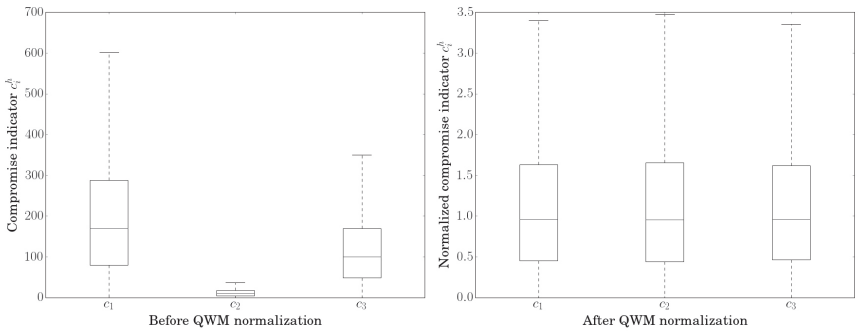
Finally, the organisation information leakage indicator $e_4$ associated with an employee is determined as the sum of the risk levels associated with keywords that they use. To associate all the exposure indicators to an internal client, AUSPEX uses the *internal assets* list described in Section 3.

# 6. PRIORITISATION OF INTERNAL CLIENTS

Previous sections described how AUSPEX calculates *compromise* and *exposure* indicators associated with internal clients of the organisation. In this section, we discuss how these indicators are combined to produce the final ranking that is presented to the security analyst.

In this section, we will examine the results of AUSPEX applied to the network traffic generated within a large organisation with 6,432 internal clients. As a first step, AUSPEX calculates the compromise indicators on all internal clients by analysing the network security logs. The output is a set of three compromise APT indicators related to maintaining access, lateral movement, and data exfiltration, denoted by $c_1^h$ , $c_2^h$ and $c_3^h$. Each index is characterised by different values and ranges, hence the evaluation of an overall *compromise score* $C_h$ for each internal client $h$ requires a normalisation step. For this, AUSPEX uses the *two-sided Quartile Weighted Median* (QWM) metric (Duffield & Lo Presti, 2009). Figure 4 shows an example of how the QWM normalises the distribution of the compromise indicators. The left chart shows a boxplot for each of the compromise indicators. This representation highlights the differences in scale and distribution of the three data sets. The chart on the right shows the same three indicators after normalisation through QWM, and demonstrates how scales and distributions are now comparable. The overall compromise score $C_h$ for each host is calculated as the weighted sum of the three normalised compromise indicators.

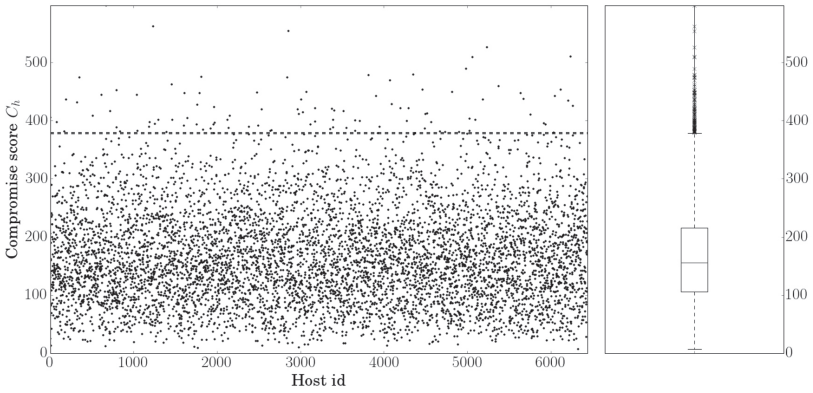
**FIGURE 4:** EFFECTS OF THE NORMALISATION OF THE COMPROMISE INDICATORS THROUGH THE QWM METRIC




The top-K clients are dynamically determined by applying the *boxplot rule* (Soong, 2004) to identify the hosts characterised by outlier compromise scores. If the compromise score values distribution does not have statistical outliers, then AUSPEX considers the clients whose score

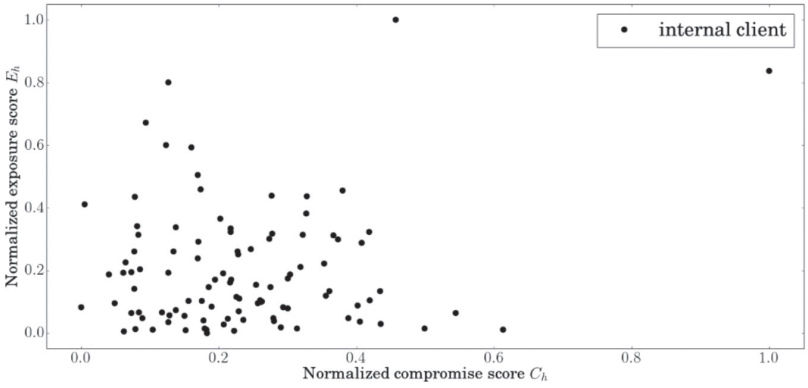
is higher than the 95-th percentile. An example of the choice of the top-K likely compromised internal clients is shown in Figure 5. On the left side, we have a scatterplot where the X-axis represents a *client id* (out of 6,432 clients in the organisation), and the Y-axis is the compromise score $C_h$. On the right side, we have a boxplot representation of the same distribution. The dashed horizontal line represents the threshold of the boxplot rule (Soong, 2004). In the considered example, only $K$=102 internal clients (2% of the original 6,432) present a compromise score higher than the threshold and are selected for further analysis.

**FIGURE 5:** CHOICE OF THE TOP-K LIKELY COMPROMISED INTERNAL CLIENTS




The top-K clients are then prioritised through the four *exposure indicators* $e_h^i$ described in section 5. For each client $h$ in the top-K list, AUSPEX calculates an overall exposure score $E_h$ by normalising the exposure indicators through QWM, as for the $C_h$ indicators. Figure 6 is a representation of the top-K internal clients where the Y-axis represents the *compromise score* $C_h$, and the X-axis the *exposure score* $E_h$.

**FIGURE 6:** NORMALISED COMPROMISED AND EXPOSURE SCORES OF THE TOP-K INTERNAL CLIENTS



To support the security analyst in prioritising analysis of likely compromised internal clients, AUSPEX produces two visual outputs: stacked histograms, and internal communications graphs.

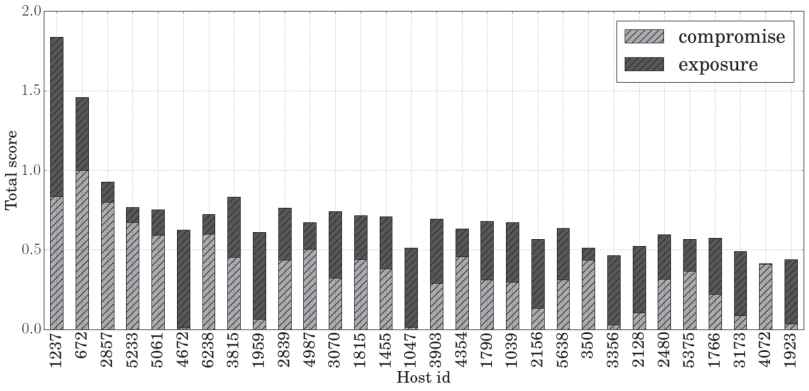
Internal hosts are ranked by considering, for example, the plot of Figure 6 and by computing the Euclidean distance from the origin that is,

$$d_h = \sqrt{E_h^2 + C_h^2}.$$

Results are sorted in decreasing order in Figure 7, where the X-axis shows the client IDs and the Y-axis denotes the sum of the compromise and exposure scores. For the sake of clarity of representation, in Figure 7 we report the internal clients having the thirty highest values of $d_h$.

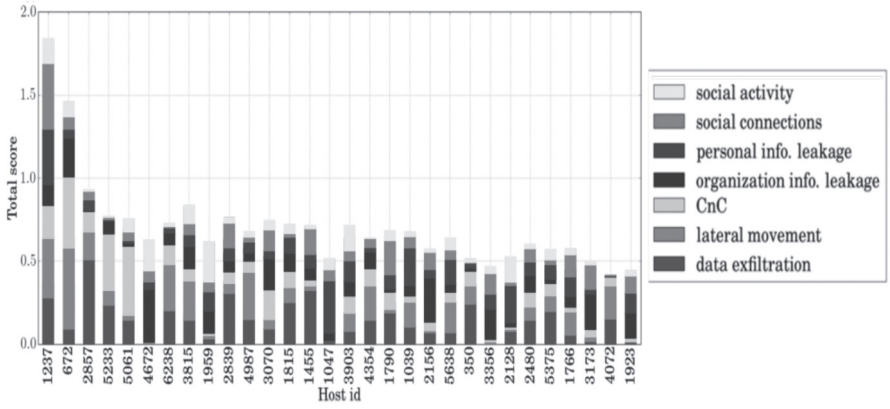
To aid the security analysts, AUSPEX also produces a view including the details of the compromise and exposure indicators. An example is given in Figure 8, where the X-axis shows the client IDs (that are the same of Figure 7) and the Y-axis represents the contribution of each indicator.

**FIGURE 7:** STACK HISTOGRAM REPRESENTING COMPROMISE AND EXPOSURE SCORES FOR THE TOP-30 INTERNAL CLIENTS



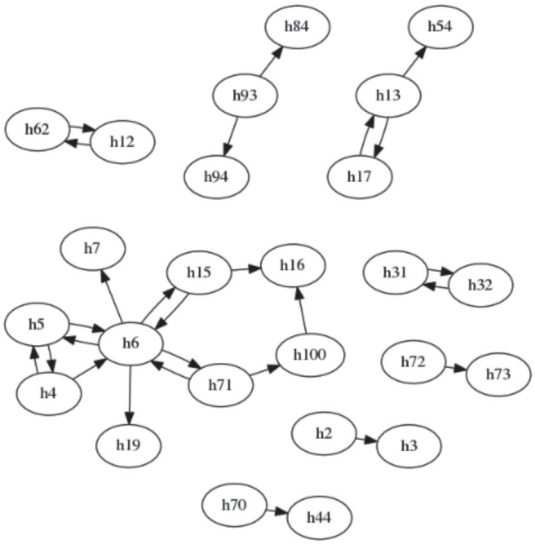

Through AUSPEX it is also possible to show the interactions between the top-K likely compromised clients that may reveal two types of information that is often related to an APT presence: the specific group of hosts that have been hit, and the lateral movements and timing of movements between hosts. These data are fundamental for forensics analyses and incident management processes.

**FIGURE 8:** STACK HISTOGRAM REPRESENTING THE SCORE DETAILS FOR THE TOP-30 INTERNAL CLIENTS




For these reasons, AUSPEX logs internal communications within the organisation to produce a graph where each node is a top-K likely compromised client, and an edge represents any type of communication between the nodes (e.g., email, chat, or other media used by the organisation). Figure 9 is an example of such a graph, where the node numbers the client order in the ranking considered in Figure 7.

**FIGURE 9:** EXAMPLE OF GRAPH OF INTERNAL SOCIAL COMMUNICATIONS RELATED TO THE TOP-K LIKELY COMPROMISED CLIENTS



# 7. RELATED WORK

To the best of our knowledge, AUSPEX is the first framework that ranks the most likely APT compromised hosts of an organisation by combining big data analytics and security intelligence on internal and external information. Big data analytics has already been applied to heterogeneous data to identify security violations (Chari, Habeck, Molloy, Park, & Teiken, 2013), but without a specific focus on APT detection. Other papers (Brewer, 2014; Jeun, Lee, & Won, 2012; Virvilis & Gritzalis, 2013) have analysed the main phases of popular APTs such as Stuxnet, Duqu, and Flame, and limit their proposals to security best practices that could be adopted to prevent them. They do not propose any novel solution or architecture for APT detection.

Other works focus on formalising the APT detection problem and defining possible detection rules, but the chosen approaches, implementation, and testing are left to future work. In this class, we can include several academic papers describing a 7-phase APT detection model (Bhatt, Toshiro Yano, & Gustavsson, 2014; Hutchins, Cloppert, & Amin, 2011); proposing an attack pyramid aiming to capture attacker movements through physical, network and application domains (Giura & Wang, 2012); or suggesting the main building blocks for an APT detection architecture (De Vries, Hoogstraaten, van den Berg, & Daskapan, 2012).

In (Friedberg, Skopik, Settanni, & Fiedler, 2015), the authors propose an anomaly detection system for identifying APTs from security logs. However, their approach requires a huge amount of data to be collected from each host, which is often impractical in large organisations. The analysis and interpretation of its output is also quite difficult and cumbersome because only generic anomalies are identified. Our focus on network logs and open source data makes it more practical, and our output is easier to interpret thanks to the use of several compromise and exposure indicators targeted on specific activities, information and clients.

Other interesting works deal with botnet detection (Bailey, Cooke, Jahanian, Xu, & Karir, 2009). A botnet is a huge set of compromised hosts that are controlled by one or more CnC servers. Several approaches have been proposed for detecting zombies and CnC servers (Gu, Perdisci, Zhang, & Lee, 2008; Gu, Porras, Yegneswaran, Fong, & Lee, 2007), but there are crucial differences that prevent the adoption of botnet detection methods in the APT domain. First, the scale of the problem is completely different, since APTs are human-driven attacks directed at specific organisations and target hosts. Hence, botnet approaches that detect similar behaviours in big groups of hosts (e.g., through clustering of traffic features) are ineffective against APTs that compromise only a few internal hosts. Infection strategies are also different. APTs often use spear phishing and zero-day exploits, while botnets tend to replicate aggressively and automatically (Bailey, Cooke, Jahanian, Xu, & Karir, 2009). AUSPEX is specifically tailored to the APT domain, and takes into account the limitations and challenges that are peculiar to the ranking of internal hosts that perform suspicious network activities.

# 8. CONCLUSIONS

We design and evaluate a novel framework that is tailored to support security analysts in detecting APTs, which represent the most critical menace to private and public organisations. They are human-driven attacks sustained by complex strategies that combine multidisciplinary skills. Hence, defence approaches based only on automatic methods or on limited information derived by internal sensors do not work because they are affected by too many false positive or negative alarms, respectively. The proposed framework uses multi-factor approaches where big data analytics methods are applied to internal and external information to support (not to replace) human specialists, so that those specialists can focus their security and intelligence analyses on the subset of hosts that are most likely to have been compromised. The proposed approach represents a step forward with respect to the state of the art and paves the way to novel methods for early detection and mitigation of APTs.

# REFERENCES


Bailey, M., Cooke, E., Jahanian, F., Xu, Y., & Karir, M. (2009). A survey of botnet and botnet detection. *Conference For Homeland Security, CATCH'09, Cybersecurity Applications \& Technology* (pp. 268-273). IEEE.

Bhatt, P., Toshiro Yano, E., & Gustavsson, P. M. (2014). Towards a Framework to Detect Multi-stage Advanced Persistent Threats Attacks. *IEEE International Symposium on Service Oriented System Engineering (SOSE)* (pp. 390-395). IEEE.

Bilge, L., Balzarotti, D., Robertson, W., Kirda, E., & Kruegel, C. (2012). Disclosure: detecting botnet command and control servers through large-scale netflow analysis. *Proceedings of the 28th ACM Annual Computer Security Applications Conference* (pp. 129-138). ACM.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Brewer, R. (2014). Advanced persistent threats: minimising the damage. *Network Security*, (pp. 5-9).

Brockwell, P. J., & Davis, R. A. (2013). *Time series: theory and methods*. Springer Science & Business Media.

Canali, C., Casolari, S., & Lancellotti, R. (2010). A quantitative methodology to identify relevant users in social networks. *IEEE International Workshop on Business Applications of Social Network Analysis (BASNA)*, (pp. 1-8).

Casolari, S., Tosi, S., & Lo Presti, F. (2012). An adaptive model for online detection of relevant state changes in Internet-based systems. *Performance Evaluation*, (pp. 206-226).

Chari, S., Habeck, T., Molloy, I., Park, Y., & Teiken, W. (2013). A bigData platform for analytics on access control policies and logs. *Proceedings of the 18th ACM symposium on Access control models and technologies (SACMAT '13)*.

Data Breaches. (2016, January). Retrieved from World most popular data breaches, Information is beautiful: http://www.informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks.

De Vries, J., Hoogstraaten, H., van den Berg, J., & Daskapan, S. (2012). Systems for Detecting Advanced Persistent Threats: A Development Roadmap Using Intelligent Data Analysis. *IEEE International Conference on Cyber Security (CyberSecurity)*, (pp. 54-61).

Denning, D. E. (1987). An intrusion-detection model. *Software Engineering, IEEE Transactions on*, (pp. 222-232).

Duffield, N. G., & Lo Presti, F. (2009). Multicast inference of packet delay variance at interior network links. *IEEE Computer and Communications Societies*., (pp. 280-285).

Friedberg, I., Skopik, F., Settanni, G., & Fiedler, R. (2015). Combating advanced persistent threats: from network event correlation to incident detection. *Computers & Security*, (pp. 35-57).

Giura, P., & Wang, W. (2012). A context-based detection framework for advanced persistent threats. *IEEE International Conference on Cyber Security (CyberSecurity)*, (pp. 69-74).

Gu, G., Perdisci, R., Zhang, J., & Lee, W. (2008). BotMiner: Clustering Analysis of Network Traffic for Protocol-and Structure-Independent Botnet Detection. *USENIX Security Symposium*, (pp. 139-154).

Gu, G., Porras, P. A., Yegneswaran, V., Fong, M. W., & Lee, W. (2007). BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation. *Usenix Security* (pp. 1-16). Usenix.

Hutchins, E. M., Cloppert, M. J., & Amin, R. M. (2011). Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. *Proceedings of the 6th International Ionference on i-Warfare and Security*.

Irani, D., Webb, S., Pu, C., & Li, K. (2011). Modeling unintended personal-information leakage from multiple online social networks. *IEEE Internet Computing*, (pp. 13-19).

Jeun, I., Lee, Y., & Won, D. (2012). A practical study on advanced persistent threats. *Computer Applications for Security, Control and System Engineering*, (pp. 144-152).

Lam, I.-F., Chen, K.-T., & Chen, L.-J. (2008). Involuntary information leakage in social network services. In *Advances in Information and Computer Security* (pp. 167-183). Springer.

Lindamood, J., Heatherly, R., Kantarcioglu, M., & Thuraisingham, B. (2009). Inferring private information using social network data. *Proceedings of the 18th AMC international conference on World wide web* (pp. 1145-1146). ACM.

Molok, N. N., Chang, S., & Ahmad, A. (2010). Information leakage through online social networking: Opening the doorway for advanced persistent threats. *School of Computer and Information Science, Edith Cowan University, Perth, Western Australia*.

Montangero, M., & Furini, M. (2015). TRank: Ranking Twitter users according to specific topics. *IEEE Consumer Communications and Networking Conference (CCNC)*, (pp. 767-772).

Montgomery, D. C. (1991). *Introduction to statistical quality control*. Wiley New York.

Ostrom, L. T., & Wilhelmsen, C. A. (2012). *Risk assessment: tools, techniques, and their applications*. John Wiley & Sons.

Paxson, V. (1999). Bro: a system for detecting network intruders in real-tim. *Elsevier Computer Networks*, (pp. 2435-2463).

Pierazzi, F., Casolari, S., Colajanni, M., & Marchetti, M. (2016). Exploratory security analytics for anomaly detection. *Computers & Security*, (pp. 28-49).

Romero, D. M., Galuba, W., Asur, S., & Huberman, B. A. (2011). Influence and passivity in social media. *Springer, Machine learning and knowledge discovery in databases*, (pp. 18-33).

Sabahi, F., & Movaghar, A. (2008). Intrusion detection: A survey. *IEEE Systems and Networks Communications (ICSNC)*, (pp. 23-26).

Schiavoni, S., Maggi, F., Cavallaro, L., & Zanero, S. (2014). Phoenix: DGA-based botnet tracking and intelligence. *Detection of Intrusions and Malware, and Vulnerability Assessment (DIVMA)* (pp. 192-211). Springer.

Soong, T. T. (2004). *Fundamentals of probability and statistics for engineers*. John Wiley & Sons.

Virvilis, N., & Gritzalis, D. (2013). The big four-what we did wrong in advanced persistent threat detection? *IEEE International Conference on Availability, Reliability and Security (ARES)*, (pp. 248-254).

Virvilis, N., Serrano, O., & Vanautgaerden, B. (2014). Changing the game: The art of deceiving sophisticated attackers. *IEEE International Conference On Cyber Conflict (CyCon)* (pp. 87-97). IEEE.

Zhou, C. V., Leckie, C., & Karunasekera, S. (2010). A survey of coordinated attacks and collaborative intrusion detection. *Computers & Security*, (pp. 124-140).