Formalising policies for insider-threat detection: A tripwire grammar

Ioannis Agrafiotis*, Arnau Erola, Michael Goldsmith, and Sadie Creese
University of Oxford, Parks Rd, Oxford OX1 3QD, United Kingdom
{ioannis.agrafiotis, arnau.erola, michael.goldsmith, sadie.creese}@cs.ox.ac.uk

Abstract

The threat that organisations face from within is growing significantly, as it has been widely demonstrated by the harm that insiders have caused recently. For many years the security community has invested in barriers and perimeters, of increasing sophistication, designed to keep those with malign intent *outside* of the organisations' information infrastructures. But assuming that one can keep the threat out of an organisation is simply not a practical stance to adopt. In our research we are concerning ourselves with how technology might be deployed to help with the detection of insider threats both automatically and in support of human-led mechanisms. This paper describes our recent research into how we might support threat detection when actions taken can be immediately determined as of concern. In particular we capture actions that fall into one of two categories: those that violate a policy which is specifically crafted to describe behaviours that should be avoided; or those that exhibit behaviours which follow a pattern of a known insider-threat attack. We view these concerning actions as something that we can design and implement *tripwires* within a system to detect. We then orchestrate these tripwires in conjunction with an anomaly detection system. We present a review of the security policies organisation apply and a grammar to describe tripwires. We further validate our grammar by formalising the most common tripwires for both categories. Our aim is to provide a single framework for unambiguously capturing tripwires, alongside a library of existing ones in use. Therefore, tripwires may be used to map experiences regardless of the heterogeneity of the security tools and practices deployed.

Keywords: Insider-threat detection, Security policies, Validation, Grammar.

1 Introduction

There is a growing concern that insider threats pose one of the most significant risks to organisations today. An insider has access to an organisations' inner systems that can persist for long periods of time, providing a potential for them to cause great harm [1]. For many years the security community has invested in barriers and perimeters, of increasing sophistication, designed to keep those with malign intent *outside* of the organisations' information infrastructures. With recent reports concurring that there is an increase in cases of insider activity [2, 3], the assumption that the threat can result only outside the organisation is not a practical stance to adopt. For that reason many organisations are investing in exploring how to detect the presence of an insider threat.

Addressing the risk from insider threat is not a new concept, particularly for large organisations. Government departments and those involved in the defence and intelligence supply chains have for some time concerned themselves with personnel security. However, the practices that have evolved commonly are based around human-centric controls; trained investigators who are able to determine when an employee may have become a threat. Such investigations are usually triggered when suspicious behaviour is

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 8:1 (Mar. 2017), pp. 26-43
*Corresponding author: Wolfson Building, University of Oxford, Parks Rd, Oxford OX1 3QD, United Kingdom, Tel: +44-1865-610805, Email: ioannis.agrafiotis@cs.ox.ac.uk

reported either by by customers or fellow employees [4, 5]. In our research we are concerning ourselves with how technology might be deployed to help with the detection of insider threats both automatically and in support of human-led mechanisms.

A strong security calture is founded in well-formed and transparent policies [4]. Therefore, it is not surprising that initially organisations attempted to mitigate insider threats by designing ad-hoc policies based on incidents they experienced in the past [6]. The dynamic regulatory and legal environment, the volatile threat landscape of insider activity, as well as the cultural differences in ethical policies posses a great challenge in capturing the appropriare security policies. These policies, which we will refer in the paper as *tripwires*, are required to be constantly under scrutiny and refinement. As a consequence, the design of an unambigious method for capturing and implementing tripwires is of paramount importance. Follow on research aimed to provide detection tools, which are able to analyse network data and investigate anomalous behaviours indicative of insider attacks [7–9]. These two approaches towards the deterrence and detection of insider threat, however, are typically only combined as selected parts of the policy space is interpreted as rules. These rules are consumed by a detection system designed to find those who are intently working outside the allowed parameters. This, in a sense, remains almost a creative space, whereby the gut-instinct of the security analyst or investigator drives the formulation of such rules based on experience and threat intelligence.

This paper describes our recent research into how we intend to enhance anomaly detection systems by capturing actions of concern that can be immediately determined because they fall into one of two categories: they violate a policy which is specifically crafted to describe behaviours that are highly likely to be of concern when observed, or they exhibit behaviours which follow a pattern of a known insider-threat attack. We view these concerning actions as something that we can design and implement *tripwires* within a system to detect. We then orchestrate these tripwires in conjunction with an anomaly detection system which is designed to find behaviours or actions that are outside normal behaviours (for an actor across time as compared to their own history, or across time when compared to others in a similar role, or across time for the entire set of actors associated with the organisation being secured). Our working hypothesis being that by deploying tripwires alongside anomaly detection we can find more of the threatening behaviours organisations are being subjected to.

During the course of our research we have had the opportunity to consider many insider attacks that organisations have faced [5, 10, 11], alongside the openly published work within the research community [6]. This has led us to believe that there is a need for a collected understanding of the kinds of *tripwires* commonly in use, and that this understanding should enable those involved in deploying insider-threat detection to benefit from the experiences of others. Sharing of experience and insight is not an unusual practice within the expert community, especially when it concerns external threats. We believe that this practice can be substantially scaled up by providing a framework within which this experience can be unambiguously formulated and shared to directly implement tripwires or equivalent detection rules in users' technology of choice.

This paper presents the foundations for such a framework – an approach to formalising tripwires of both categories. Our aim is to provide a single framework for unambiguously capturing tripwires, alongside a library of existing ones in use. Therefore, tripwires may be used to map experiences regardless of the heterogeneity of the security tools and practices deployed. We intend this formal framework for describing and documenting tripwires to be used not only in supporting our programme of research. Our aim is to bridge the gap between practitioners and the configuration of our detection tool prototype, but also others can choose to select tripwires from the library and implement them in their own tooling. In our detection system, namely Corporate Insider Threat Detection (CITD), we are building a programme generator that will take these tripwires and create the relevant detection code. Others might choose a more manual translation. We hope that this may provide a foundation for further sharing and learn-

ing, and perhaps a medium for the community to transcend operational sensitivities towards a sharing paradigm that could offer significant harm mitigation benefits.

This paper is an extended version of the work originally presented at the 8th ACM CCS International Workshop on Managing Insider Security Threats (MIST'16), Vienna, Austria, October 2016 [12]. It provides a novel section reflecting on security policies that organisations implement to deter insider threats, an extended literature review on formal approaches for capturing such policies, as well as a validation chapter focusing on formalising the most common tripwires. In what follows, Section 2 reviews best practices and security policies for insider threat and groups these into families of concerns. In Section 3 we further reflect on methods focusing on formalising the insider-threat problem. Section 4 summarises the architecture and technical details of our CITD detection system, while Section 5 provides the common policies identified for both types of tripwires which are used to validate our formal approach presented in Section 6. The formalisations for tripwires for both categories is the way in which we document how tripwires are suitable for automatic programme generation. Finally, Section 7 concludes the paper and provides insights for future work.

2 Reviewing insider-threat organisational policies

The significantly increasing threat, which insiders pose to organisations, has motivated the evolution of insider-threat mitigation programs from ad-hoc policies, aiming at addressing general security concerns, to comprehensive frameworks vocalising the issue of insider threat and promoting best practises for mitigating this risk [13–16]. The Vormetric Insider Threat report provides further emphasis for the need of clear directives for organisations regarding insider threat, since only 11% of the respondents were reassured that their organisation was not vulnerable to insider threat, with the vast majority (93%) planning to invest in implementing best practices and compliance to mitigate this threat [2].

According to Balakrishnan [15] a structured program for insider threat requires senior management support "addressed by policies, procedures and technical controls". The INSA roadmap describes the key broad categories for such a program. The roadmap comprises of three steps, namely initiation, planning and operations and fundamental to its success is the identification of stakeholders, and a detailed governance structure and policy linked to detection indicators [15]. In a similar vein, the NIST cybersecurity framework [13] suggests that organisations need to identify a comprehensive set of policies and requirements for employees. These policies should be implemented through access control, awareness training and data protection while continuous monitoring of assets should highlight anomalous behaviours. Finally, a respond and recover strategy is deemed essential for a complete program. The same exact principles (identify, protect, detect, respond, recover) are described by SIFMA [14] with similar requirements for the fulfilment of each one. There is an additional focus on risk management in the identification step as well as information on legal, ethical and privacy concerns. The CERT institution provides a guide with 19 practices, covering most of the aforementioned points. The authors also raise international considerations regarding cultural backgrounds in organisations from different parts of the world providing insights on how these may be resolved.

Reflecting on best practice frameworks [3, 13, 14, 17] and on security policy reports publicly available [18–22] the five step model for addressing the insider-threat problem is evident. The policies which derive from this model can be categorised into those whose effective implementation or potential violation is evident from network data and digital sources, and those for which digital data is not available. The later category comprises policies about raising awareness via educating employees, establishing a strong security culture, strategies for incident response and recovery planning. In addition, policies which require human intervention to detect an insider threat, such as notification of an employee for a colleague's suspicious behaviour [4], belong to this category.

Focusing on policies where digital evidence exists, these may be categorised into those whose purpose is to protect assets and deter attacks, and those whose purpose is to detect an attack once it has occured. A plethora of reports concur that the identification of sensitive data, the implementation of access control systems and the allocation of clear roles to organisation's workforce are key policies for deterring (as well as detecting) insider threats [3,13,16,17]. NIST emphasise the importance of performing credential management, the protection of remote access to organisational networks including mobile restrictions and cloud computing policies and the monitoring of systems for use of unauthenticated devices [13]. Elaborating on remote access policies, CERT identifies the need for immediately disabling this access during employee termination [16]. A report from PwC emphasises on the implementation of access control systems, denoting the values of policies regarding the control of the use of USB ports on computers, the control of access and transfer of sensitive data as well monitoring internet access of privileged users [3].

Departing from deterrence policies, companies have also focused on the design of policies able to detect suspicious insider-threat behaviour. These policies are derived based on performing data analytics on network data and on examining cases of well known insider attacks. Splunk suggests that systems who provide threat intelligence on malware and questionable sites, such as FireEye, should be used to monitor online activity of employees [23]. CERT suggests the monitoring of accounts for password violations as well as the use of social media during working hours. The reason being that social media may be used to publicly share organisational information which could facilitate phising attempts and fraud schemes. Login activity outside office hours may trigger a policy violation for Splunk, based on the premise that insiders usually act during this time-framework [24]. In a similar vein, and based on observations of insider activity, the traveller's problem is monitored by organisations; the same remote access account cannot be active in two or more devices whose geographical location (or logical location) is *different*. The rational behind this policy is that attackers often make use of the credentials of their colleagues either to elevate their privileges or to cover their tracks [11].

All the aforementioned policies can be implemented as tripwires and once a violation occurs an alert may be raised by a detection system. Research, however, has identified that there exist patterns of behaviour which insiders exhibit when conducting specific attacks [6,11]. In [6] the authors provide critical paths which insiders usually follow to execute intellectual property theft, fraud or sabotage. Adopting a similar approach, researchers in [11] have analysed more than 120 real cases of insider theft to unveil patterns and provide several attack graphs of insider activity. One can imagine that a more sophisticated system would actively detect the occurrence of these patterns. It is thus, essential to create statefull tripwires which will only be triggered if a specific sequence of policy violations occurs.

Concluding our review, it is widely accepted that clear and enforced policies provide the best guidance for mitigating the risk of insider threats, by both deterring some and by helping the accidental threat to avoid creating problems. If employees are able to correctly and consistently follow policies, it is hypothesised that this will result in a stronger security culture within an organisation, which should in turn help to control risk. However, this will not work for all cases of insider threat, it will only serve to limit the noise that detection methods face. The task of detection of insiders remains. We might hope that the policies outlined above will provide some inspiration for tripwires, since we will be interested in those actors who continue to act against the policy. We next turn to the literature on formalisations of the insider-threat problem.

3 Formalising the insider-threat problem

Halpern et al [25] demonstrated that policies describing the conditions under which actions are permitted or forbidden may be formalised with first-order logic. Applying Weber's sociological process of "un-

derstanding explanation", the authors in [26] use Higher Order Logic to describe this process and, as an example of modeling human behaviour, they analyse an insider-threat scenario. The authors are able to formally describe the classes identified in CERT's taxonomy such as "Precipitating Event or catalyst, the individual's Personality Characteristics, Historical Behaviour, Psychological State, Attitude Towards Work, Skill Set, Opportunity, and Motivation to Attack" [6].

Adopting a similar approach, Kammüller et al. [27] present one of the first attempts to formalise organisational policies for insider threats. They provide the language to formalise structural information and they try to establish a series of actions that would result in a system state where the negated formalised version of the policy is true. The different sequences of actions resulting in the violation of the policy can be used to identify attack vectors and refine access control systems.

Ivanova et al [28] suggest a model capturing the infrastructure of organisations, such as rooms and locations as nodes in directed graphs. Actors and assets are also represented as nodes. In their model, actors can perform actions on locations and these actions are constrained by policies providing the necessary credentials to actors, formalised as predicates. The language is rich enough to describe situations denoting that an actor is an employee of a service provider and has specific credentials. Policies are describing actions that should be prohibited, and once invalidated they identify the actors who may exhibit this behaviour, the set of actions which may render the system in an undesirable state and the locations in the enterprise where this state may occur.

In [29], researchers adopt a two-fold approach formalising a taxonomy of insider threats presented by Nurse et al [10] and estimating the likelihood of an employee conducting an insider attack. The approach adopts Bayesian networks to model the human disposition and Markov Decision Processes to model the possible actions of towards a successful attack. The framework is able to formalise concepts such as personal characteristics of an attacker and skills required to execute an attack.

Crampton et al. [30] adapt role-based access-control languages to the insider-threat problem. The premise is that trusted users of the system cannot actually always be trusted, thus there are novel requirements for formalisation which can take account of more operational context. The authors suggest the formalisation of access requests in the form of tuples (s,o,a,c) where s ranges over subjects, o ranges over objects (i.e. protected resources), a ranges over actions (such as read, write, etc.), and c ranges over contextual information (such as whether the user is accessing the system remotely). Tuples form subsets which are then interpreted as predicates referred to as request predicates. Policies either accept or deny these requests. Finally, Magklaras et al. make use of Domain Specific Language to construct an Insider Threat Prediction Specification Language [31,32]. The aim is to capture the misuses of IT policy violations, indicative of insider activity.

Reflecting on the relevant literature, all the formalisations are expressive enough to either formalise organisational policies in terms of access control or describe situations indicative of insider threat. There is not, however, a clear understanding of what data these formalisations require and how they may be used to provide a triggering event (such as a tripwire in the sense we consider in this paper) indicating a policy violation or a behavioural pattern of interest for the purposes of a detection system. It is this gap that this paper attempts to fill by providing a tripwire formalisation closely related to the detection system we have designed over the last two years. The formalisation takes into account the features created by the system and the anomalies detected. The next section provides a brief description of the overall architecture of our detection tool.

4 CITD system architecture

The system architecture is presented in Figure 1 and is detailed in [33]. The available logs for the company are parsed to the specific input format of the system. All events are identified, allowing to

correlate the information from the same user. From this information the system creates a history-tree of users' past behaviour and observations, and constructs the baseline for the anomaly detection. For each input event, the process adds the appropriate information to the user's tree, and extracts features from the event and the existing tree's content. For instance, if this is the first time that the system observes the user accessing a particular file, it will report this into the vector of features and update the statistics accordingly.

From the vector of features, the system is calculating three types of alerts named L1, L2 and L3. L1 alerts corresponds to the tripwires (including policy violations) described in this paper, and are calculated after each observed event. L2 alerts are specific deviations from the expected behaviour, while L3 alerts are non-specific deviations from the observed behaviour. The main difference between L2 and L3 alerts is that L2 have a defined anomaly threshold, while L3 alerts use machine learning to calculate that threshold. At the end of each day, L2 and L3 anomaly detection is run to compare each user's behaviour with the expected behaviour for the user, and the expected behaviour for other users who have the same job category (role). If the anomaly score is greater than the threshold, we raise an alert for that user (for a specific anomaly type).

Anomaly types are specified in the tool and describe sets of features that have some relevance for the anomaly. For instance, the anomaly file exfiltration can collate the features: new file accessed, number of files accessed, unusual login time, unusual file access time. All these features can point to a possible data exfiltration threat, and thus, they are relevant for this anomaly. Our system has been applied on datasets obtained from an organisation and the results of this study can be found in [34]. It was this experience obtained from getting access to the datasets of the organisation as well as the structure of the synthetic scenarios that guided the development of our grammar presented in the next Section 5.

5 Tripwires identified for formalisation

Sections 2 and 3 provided an overview of white papers [18–22] published by organisations and the academic literature identifying strategies and policies for mitigating insider threats. In this section we identify and present the most common policies widely adopted by different institutions, as well as the most prevalent attack patterns of insider activity. These will be used for validating our formal language for describing tripwires in Section 6.

We are developing support for two types of tripwire: *policy-violation* and *attack-pattern* tripwires. These two types of alerts in our detection system are similar in that they do not require anomaly detection techniques in order to fire; both are simply looking for specific behaviours and patterns directly presented in the data. This also means that they will not suffer from false-positive alerts, as they will only fire when it is certain that a behaviour of interest has been exhibited somewhere on the system. However, the degree to which the behaviour or events are concerning (indicate a significant threat) will be open to interpretation by the organisation. Since the nature of insider threats vary amongst different organisations (some organisations are prone to IP theft others to sabotage), so will the relevance and usefulness of the tripwires. Of course, as the threat landscape is constantly changing, so too will the usefulness of tripwires.

More specifically, the first type of tripwire alerts (policy violations) captures tripwires which we have identified in publicly available documents and are widely applied in other organisations. The second type of tripwire alerts (attack-patterns) captures a sequence of events of interest based on known insider attacks. Previous research has identified attack graphs by analysing more than one hundrent real cases of insider attacks [11].

Taking each in turn. The common policies we observe and intend to use in validating the utility of our formalisation are:

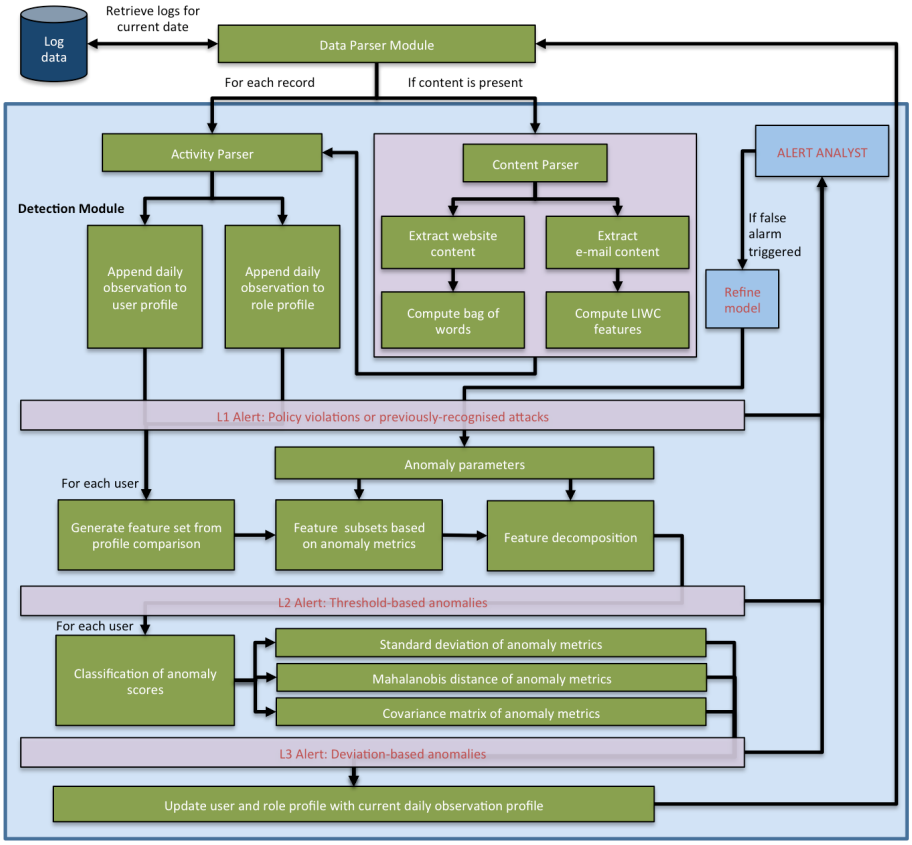


Figure 1: Architecture of the CITD insider-threat detection system

- Physical impossibility (e.g. multiple VPN connections from geographically diverse IP addresses)
- Exceed threshold of accesses to black-listed websites
- Use of unauthorized USB sticks
- Exceed threshold of unsuccessful authentication attempts to
  - Email account access
  - Service access
  - File access
- Exceed threshold of emails to personal accounts with attachment file names matching corporate files
- Exceed threshold of emails containing attached files of significant size
- Exceed threshold of total size of email attachments sent in a given time-frame

Policies are of great interest as they can easily be flagged in near real-time. The output format of the L1 tripwire alerts in our system will be tuples in the form:

⟨User, Name of policy, Severity of alert [1], Number of violations in the current day⟩

Regarding tripwires based on attack-patterns which capture sequential actions of interest, we will use the most commonly identified patterns in [11]. It is crucial to capture specific patterns which insiders follow to exfiltrate information or to sabotage the organisation's functionalities. These actions do not always represent significant deviations from the employee's normal behaviour, rendering the detection based on anomaly detection systems challenging; the sequence, however, in which these actions are executed is of high importance and may provide useful information for further analysis of the employee's behaviour.
The output format of the sequential actions in our system is:

⟨User, Name of attack-pattern tripwire, Severity of alert[1], Number of violations in the current day⟩

The most common attack patterns we identified from our work [11] are:

- Unusual login time → Unusual file access activity → Unusual number of emails with attachments sent to personal email account
- Unusual login time → Unusual file access activity → USB stick insertion
- Access to a colleague's VPN credentials → Unusual login time → Unusual number of files deleted
- Unusual visits to competitors' sites → Unusual time login → Unusual file access activity
- Drop in file access activity → Drop in web activity → Drop in email correspondence → Unusual visits to competitors' sites
- Increased use of social media → Unusual login time → Unusual file access activity → Unusual email activity

6 Formalising Tripwires

We outline a formal language to ensure that the requirements we capture for both types of tripwire alerts are clear and unambiguous. The formal language is based on the architecture of the system. It provides functions which transform data from different types of logs, features of the detection system and alert outputs in the form of tuples. These tuples provide information which is critical for the implementation of L1 alerts. The types of data[2] available during our first trial of the system [34] guides the grammar and our examples presented here, however, does not restrict the language and its expressiveness. If further datasets are available, then all the formalisations can be easily extended to accommodate for the new types of data, by simply replacing the types of data presented here with the novel ones.

We define a set, *Raw_Data*, of functions for transforming raw data into tuples. More specifically, $Raw\_Data = \{f_{email}, f_{web}, f_{file}, f_{VPN}, f_{log\text{-}on}, f_{log\text{-}off}, \ldots\}$. Each f in *Raw_Data* maps an individual log entry of the appropriate type into a valid *raw data* tuple.

In a similar vein, we define a set of functions named, $Features = \{f_1, f_2, \ldots, f_n\}$[3], containing a function for each feature extracted by the detection system. Each f_n maps the value of the counter to an instance of the n^{th} class of features into a valid *feature* tuple.

[1] Yellow, orange, red based on thresholds defined by organisations
[2] During the trial we acquired access to file logs, email logs, VPN logs, proxy logs and logs indicating login and logoff activity.
[3] n is the number of features generated by the CITD system. This number varies based on the available datasets.

Finally, we define a set of functions named, $F_{Alert} = \{f_{\text{New\_email}}, f_{\text{Email}}, f_{\text{New\_file}}, f_{\text{File}}, f_{\text{New\_log-on}}, f_{\text{Log-on}}, f_{\text{User}}, f_{\text{New\_Hourly}}, f_{\text{Hourly}}, f_{\text{New\_web}}, f_{\text{Web}}, f_{\text{New\_log-off}}, f_{\text{Log-off}}, f_{\text{Role}}, \ldots\}$. Each function f_X in F_{Alert} maps the data attached to an X-alert into a valid *alert tuple*.

A *raw-data tuple* is defined to contain five fields; $\langle user, device, activity, key, metadata\rangle$; where:

- $user \in \boldsymbol{U}$, all the employees in an organisation
- $device \in \boldsymbol{D}$, all devices organisations might create logs for
- $activity \in \boldsymbol{A}$ {file, log-on, log-off, VPN, email, web, insert, remove}
- $key\_data \in \boldsymbol{K}$, an attribute:value binding
- $metadata \in \boldsymbol{M}$, a sequence of such bindings

Attributes are drawn from a set of tags $\boldsymbol{G}$={Filename, URL, Email_to, Email_from, Source_IP,...} and values are drawn from a space $\boldsymbol{V}$=$Time \uplus Size\_of\_file \uplus Duration \uplus IP \uplus Size\_of\_attachement \uplus Email\_to \uplus Email\_from \uplus Category\_for\_websites \uplus \ldots$ In this set we include all the available types of data that might be tagged with an attribute.

It is evident that some tuples are not valid because the metadata values are not relevant for the activities they capture (i.e. we cannot have a file activity and an attribute Email_to attached to it). We therefore define a set of valid tuples $\boldsymbol{J}$.

For example, consider the raw VPN logs in an APACHE format [35], as commonly captured by organisations. These are of the form [VPN_name, date_utc, messsage_id, source_ip, user_id, duration, xmt, rcv, internal_ip, reason_for_disconnect]. Let us apply the f_{VPN} function to a data entry with the following value:

raw_vpn_1; *raw_vpn_1* = [sngidc-vpn-cluster-1, 2014-10-10 03:32:35, ASA-4-113019, 69.89.31.226, lbegum1962,
25:02:01, 7426402, 4758369, 192.168.1.50, Idle Timeout].

$f_{VPN}(raw_vpn_1) = \langle lbegum1962, \perp, VPN, \langle$Source_IP : 69 : 89 : 31 : 226, Duration : 25:02:01, Date:2014-10-10 03:32:35$\rangle\rangle$, which is a valid tuple in $\boldsymbol{J}$.

Here, the device field is missing from the data logs. We obtain a set comprising tuples; we assign to the user named lbegum1962, a VPN activity which originated from a device with 69.89.31.226 as an IP address; the first tuple indicates that the activity lasted for 25 (hours):02 (minutes):01 (seconds) and the second tuple that it started at 2014-10-10 03:32:35.

From our experience with logs from trial partners and synthetic scenarios [34], there are cases where the metadata values or the device name is missing; in most of these cases these values are irrelevant to the tripwires we intend to design. Therefore, we allow for *raw data* tuples to be defined with the other fields only and have the device or metadata fields replaced with a $\perp$ symbol, representing *undefined*. In cases where missing values are crucial for an implementation of a tripwire, further triangulation of logs of different types will be required (i.e. find from the internal IP address assigned to the VPN log, which device was assigned to it by examining the logs recording which IP addresses are mapped to the mac addresses of devices).

The features we extract to feed into machine learning algorithms are counting occurrences in the raw data of some of the tuples in $\boldsymbol{J}$ over a period of time (usually set to 24 hours). Our system calculates more than 550 features, depending on the datasets available, we therefore define another type of tuple to allow us to exploit the features directly. The *features* tuple is defined to contain three fields, namely the name of the user, the feature we are interested in and the value attached to the instance of the n^{th} feature. More formally, $\langle user, n,$ value of feature $n\rangle$.

Finally, for the attack-pattern tripwires, we require access to L3 alerts to capture intermediate steps of unusual activity. We therefore define an alert tuple which contains three fields, namely the name of the user, the name of the alert being raised, the severity of the alert. More formally, $alert\_tuple = \langle user, alert, severity \rangle$.

Regarding L1 alerts, when we recognise that a tripwire has been triggered we raise the alert to the main detection engine, which increments its count. Write **Flag** $\langle$ *user*, *alert*, *severity*$\rangle$, where *alert* is the name of the policy we are raising a flag for. Since the engine is keeping count of the alerts, when we observe an L1 tuple a count field will be included $\langle$ *user*, *alert*, *severity*, *count* $\rangle$, where *alert* is the name of the tripwire we are raising a flag for and *count* is the number of **Flag** tuples the system has observed so far in the current day for the same tripwire.

For functions, lists of activities, attributes and metadata, it is important to point out that we do not claim to have provided an exhaustive list; what we present here is merely representative. Over time we expect our library of formalised tripwire descriptions to evolve. In terms of implementation and use, the realisation in tripwires will depend upon the data each organisation has available.

In the next section we present a tripwire grammar based on the most common polices identified in the literature. In Section 6.2 we present extensions to the grammar to include tripwires for attack-patterns based on the attack graphs created during our project [11].

6.1 Grammar for policy violation tripwires

Here we provide the formalisations of the policies described in Section 5.

The physical impossibility

The first example of tripwire captures physical impossible VPN accesses (multiple VPN connections from different regional IP addresses). These types of violations may be indicative of stolen VPN credentials and elevation of access privileges. This formalisation is based on the initial trial partner datalogs for VPN. Suppose $\boldsymbol{K}$ contains *Source_IP*, *Duration*, *Date*. Let $u \in \boldsymbol{U}$ and $ip_1, ip_2 \in \boldsymbol{V}$ and are values of the *Source_IP*, $date_1, date_2 \in \boldsymbol{V}$ and are values of *Date* and $duration_1, duration_2 \in \boldsymbol{V}$ and are values of *Duration*. If we observe $\langle u, \perp, VPN, \langle$Source_IP:$ip_1$, Duration:$duration_1$, Date:$date_1\rangle\rangle$ and $\langle u, \perp, VPN, \langle$Source_IP: ip_2, Duration:$duration_2$, Date:$date_2\rangle\rangle$, such that $ip_1 \neq ip_2$ AND $date_1 \leq date_2$ AND $date_1 + duration_1 \geq date_2$ then **Flag** $\langle u, Physical\_impossibility, Red\rangle$ AND the system will increase this specific alert count. Notice that we do not assume that we have information about the name of the device, hence the $\perp$ wildcard symbol is used. If the VPN session has not been terminated, the duration value is ∞.

This is a strict tripwire since it does not allow a person to connect from two different devices at the same time. Alternatively, we could consider that the *Date* attributes will not differ more than 12 hours and compare only the first n numbers of the IP address, depending on how restrictive the organisation requires the distance between the devices using the same VPN access to be.

Number of accesses to black-listed websites

Most organisations provide black-listed websites which either indicate the existence of malware or inappropriate content. This list may also contain websites indicative of financial problems, competitors' websites, etc. The formalisation is based on the data logs we obtained from X_5.

Let B be the set of all the black-listed websites. Suppose $\boldsymbol{K}$ includes *URL* and *Date*. For $u \in \boldsymbol{U}$, $url \in \boldsymbol{V}$ and $date \in \boldsymbol{V}$, if we observe the tuple $\langle u, \perp, website, \langle$URL:*url*, Date:*date* $\rangle\rangle$ such that $url \in B$ then **Flag** $\langle u, Black\_List, Orange \rangle$.

Usage of social media

Most organisations have policies regarding the usage of social media. Research results from our research [11], as well as publicly available reports [16] showed that employees who were either coerced or flattered by third parties to instigate insider attacks, increased their usage of social media. The formalisation is based on the data logs we obtained from one of our industrial partners.

Let S be the set of all the social media websites. Suppose $\boldsymbol{K}$ includes URL and $Date$. For $u \in \boldsymbol{U}$, $url \in \boldsymbol{V}$ and $date \in \boldsymbol{V}$, if we observe $\langle\ u, \perp, website, \langle\ \mathsf{URL}{:}url, \mathsf{Date}{:}date\ \rangle\rangle$ and $url \in S$ then **Flag** $\langle\ u, Social\_media\_usage, White\ \rangle$. Notice that when a social-media website is detected in the website logs we raise a flag of white severity. Once the threshold is exceeded, we raise an alert of an appropriate severity: when we observe the alert tuple $\langle user, Social\_media\_usage, White, 50\rangle$, **Flag** $\langle user, Social\_media\_usage, Orange\rangle$.

Use of unregistered USB sticks

There are a number of organisations which allow the usage of registered USB sticks only. Potentially, usage of an unregistered USB stick may imply data exfiltration attempts or risk of malware infection. The formalisation of this policy is based on the insert logs used in the synthetic scenarios.

Let B be the set of all the registered USB devices. Suppose $\boldsymbol{K}$ contains USB, $u \in \boldsymbol{U}$ and $usb \in \boldsymbol{V}$. If we observe $\langle\ u, \perp, insert, \langle \mathsf{USB}{:}usb\rangle\rangle$ and $usb \notin B$ then **Flag** $\langle\ u, Unregistered\_USB, Orange\rangle$. Note that for this tripwire we do not require information about the device where the USB was inserted nor other attributes regarding the USB device.

Number of unsuccessful authentication attempts

Many organisations record the number of failed authentication attempts and may choose to act if a user surpasses a specific threshold. Usually, logs are recorded for access to systems, email accounts and internal file systems. In our engagement with the initial trial partner we acquired logs which recorded unsuccessful authentication attempts to obtain access to files and we will define a policy to calculate how many failing attempts employees make in a specific period of time. The formalisation example is based on data logs provided by the initial trial partner.

Suppose $\boldsymbol{K}$ contains $Filename$, $Date$ and $Verified\_access$. For $u \in \boldsymbol{U}$, $file \in \boldsymbol{V}$, $date \in \boldsymbol{V}$ and 100 and 200 $\in \boldsymbol{V}$, if we observe $\langle\ u, \perp, file, \langle \mathsf{Filename}{:}file, \mathsf{Verified\_access}{:}200, \mathsf{Date}{:}date\rangle\rangle$ then **Flag** $\langle\ u, unsuccessful\_attempts\_policy, White\ \rangle$. If we observe $\langle\ u, unsuccessful\_attempts\_policy, White, 20\ \rangle$ we raise **Flag** $\langle\ u, unsuccessful\_attempts\_policy, Orange\rangle$.

In the logs we acquired from the trial partner [34], the denial-of-access to a file is captured with the value 200 in the appropriate field. Otherwise, if the access is granted, it is denoted with the value 100. The rationale in this formalisation is similar to the one described in the violation of social-media policy. We raise an alert only when a threshold is exceeded (in this case the threshold is set to 20).

Emails to personal accounts with large attachment files

One of the most common policies, which organisations across sectors have adopted, restricts the size of attachment files sent over email communication. Data exfiltration over email is a popular method followed by insiders. Our related tripwire formalisation below is based on the logs obtained from the initial trial partner.

Suppose $\boldsymbol{K}$ includes $Email\_to$, $Attachment\_size$, $Attachment\_name$ and $Date$. For $u \in \boldsymbol{U}$, $email\_to$, $size$, $name$ and $date \in \boldsymbol{V}$, if we observe $\langle\ u, email, \langle\ \mathsf{Email\_to}{:}email\_to, \mathsf{Attachment\_size}{:}size, \mathsf{Attachment\_name}{:}name, \mathsf{Date}{:}date\ \rangle\rangle$ and $size \geq 2GB$ then **Flag** $\langle\ u, Email\_attachment, Red\ \rangle$.

Another variation of this tripwire could focus on the name of the file instead of the size. The rationale behind this would be to ensure that sensitive files cannot be attached and send over by email. This formalisation would require an organisation to provide a set of sensitive files S. More specifically: For $u \in \boldsymbol{U}$, *email_to*, *size*, *name* and $date \in \boldsymbol{V}$, if we observe $\langle$ u, *email*, $\langle$Email_to:*email_to*, Attachment_size:*size*, Attachment_name:*name*, Date:$date$ $\rangle\rangle$ and *name* $\in S$ then **Flag** $\langle$ u, *Email_attachment_name*, *Red* $\rangle$.

Additionally, we could design a stricter tripwire to identify emails being sent to personal accounts provided by Gmail, Yahoo or Hotmail services. We would need to maintain a list of personal emails named, *EMAIL*, and whether the value of the attribute *email_to* observed in the data logs exists in this set.

Suppose $\boldsymbol{K}$ includes *Email_to*, *Attachment_size*, *Attachment_name* and *Date*. For $u \in \boldsymbol{U}$, *email_to*, *size*, *name* and $date \in \boldsymbol{V}$, if we observe $\langle$ u, *email*, $\langle$Email_to:*email_to*, Attachment_size:*size*, Attachment_name:*name*, Date:$date$ $\rangle\rangle$ and *email_to* $\in$ *EMAIL* then **Flag** $\langle$ u, *Personal_account*, *Red* $\rangle$.

6.2 Grammar for attack-pattern tripwires

The attack pattern-tripwires differ from the policy violation tripwires in two ways; the first is the larger number of interesting behaviours observed in attack patterns and, more importantly, the sequence with which these behaviours occur. The second difference is the temporal element, capturing the time intervals between these behaviours. Thus, the formalised tripwires can be used to describe parts of the attack patterns. We need, however, to extend the grammar presented in Section 6.1 to allow us to maintain states of interesting behaviour for all the employees; when the first behaviour in the attack pattern is observed in the data logs, within a given time, the state would change to indicate the following behaviour of interest the system should be looking to observe. The alert is raised when all behaviours of interest are observed in the data logs, with the sequence denoted in the attack pattern and within a given time-framework.

We define a set of abstract states Σ, which reflects the state of progress of each attack pattern for a specific user. We also define a universal initial state S_0 as the starting point of all the attack patterns and a set of final states named Σ_f comprising of states (S_{fn}) denoting the end of an attack pattern. The initial and final states are part of Σ. We also define a set Σ_i to capture intermediate states, denoting situations where there are more than one activities required to reach the next state, the sequence of which is irrelevant. Consider the example where an attack pattern requires 3 steps to completion with a single point as an outcome; there is an initial state S_i, a final state S_f, two states S_1 and S_2, and $\Sigma = \{S_0,\ S_1,\ S_2,\ S_3,\ S_f\}$. When we reach the final state we flag the alert with the appropriate data. This is captured as **Flag** $\langle User\ N, Name\ of\ pattern, Severity \rangle$.

We define a set Φ to denote valid transitions between states. A transition is a triple of the form (S_n, t, S_{n+1}), where $S_n \in \Sigma - \Sigma_f$, $S_{n+1} \in \Sigma$ and t is the triggering event which belongs to a set T of triggering events. The triggering event is a valid tuple in the $\boldsymbol{J}$ set or a *timeout* event. A timeout event is created when a specified time between two triggering events has passed, however the required event leading to the next state has not been observed. Time constraints between state-transitions will allow us to create windows of opportunity for an attack pattern to be executed and emphasise on situations of interest.

Regarding transitions, each one comprises two attributes:

- The first attribute *trigger* captures events required to commence a transition. In practice, this will be expressed by a pattern the tuple should match, which may contain values observed in previous tuples.

- The second attribute is the *time* variable which defines the time interval within which triggering events must be observed.

The following notation is used for the transitions:

$$\phi : S_n \xrightarrow{(trigger,\ time)} S_{n+1}$$

Next, we formalise the most prevalent attack patterns we have identified in our work. As for the policy based tripwires, we do not pretend that this set of examples is exhaustive, merely representative for the purposes of this paper.

Exfiltration of files via email

In this attack pattern the insider logs in at an unusual time, downloads files and sends these to their personal email account. There are four different states and six transitions to be defined. More specifically, we have the initial state S_0; the final state S_f, which in our attack pattern is a violation of a policy defined in Section 6.1 regarding the usage of personal email; state S_1 and S_2. Triggering events leading from S_0 to S_1 are alerts regarding unusual time logins, whereas transitions from S_1 to S_2 require the observation of the values of specific features focusing on file behaviour.

The transitions in the Φ set are built from:

- $\phi_0 : S_0 \xrightarrow{(\langle User\ N,\ log{-}on\ anomaly \geq Red\rangle,\ \perp)} S_1$
- $\phi_1 : S_1 \xrightarrow{(\langle User\ N,\ file\ anomaly \geq Orange\rangle,\ 5\ hours)} S_2,$
- $\phi_2 : S_1 \xrightarrow{(\langle User\ N,\ new\ file\ anomaly \geq Orange\rangle,\ 5\ hours)} S_2$
- $\phi_3 : S_1 \xrightarrow{(\langle User\ N,\ feature\ file \geq average\ of\ role\rangle,\ 5\ hours)} S_2$
- $\phi_4 : S_1 \xrightarrow{(\langle User\ N,\ feature\ new\ file \geq average\ of\ role\rangle,\ 5\ hours)} S_2$
- $\phi_5 : S_2 \xrightarrow{(\langle User\ N,\ Email\_Usage\rangle,\ 1\ hour)} S_f$
- $\phi_6 : \forall\, S_n \in \Sigma - S_0,\ S_n \xrightarrow{timeout} S_0$

Finally, $\Phi = \phi_1 \cup \phi_2 \cup \phi_3 \cup \phi_4 \cup \phi_5 \cup \phi_6$. Note that there are four possible transitions from the first step to the second step of the attack pattern. Any one of the four behaviours of interest would suffice to transit from the first state to the next one. The ϕ_6 transition ensures that if there is a *timeout* event at any point of the attack pattern, we will return to the initial state.

Exfiltration of files via removable devices

In this attack pattern the insider logs in at an unusual time, downloads files and stores these to their removable device. There are four different states and six transitions to be defined. More specifically, we have the initial state S_0; the final state, which will be achieved if an alert for the use of a USB device is observed and is defined as S_f; the first step S_1requires as a trigger a log-on anomaly; the second step S_2 requires the observation of the values of specific features focusing on file behaviour.

The transitions in the Φ set are built from:

- $\phi_0 : S_0 \xrightarrow{(\langle User\ N,\ log{-}on\ anomaly \geq Red\rangle,\ \perp)} S_1$
- $\phi_1 : S_1 \xrightarrow{(\langle User\ N,\ file\ anomaly \geq Orange\rangle,\ 24\ hours)} S_2,$
- $\phi_2 : S_1 \xrightarrow{(\langle User\ N,\ new\ file\ anomaly \geq Orange\rangle,\ 24\ hours)} S_2$

- $\phi_3 : S_1 \xrightarrow{(\langle User\ N,\ feature\ file \geq average\ of\ role \rangle,\ 5\ hours)} S_2$
- $\phi_4 : S_1 \xrightarrow{(\langle User\ N,\ feature\ new\ file \geq average\ of\ role \rangle,\ 5\ hours)} S_2$
- $\phi_5 : S_2 \xrightarrow{\langle User\ N,\ Insert\ alert > Orange \rangle,\ 1\ hour)} S_f$
- $\phi_6 : \forall\ S_n \in \Sigma - S_0,\ S_n \xrightarrow{timeout} S_0$

Finally, $\Phi = \phi_1 \cup \phi_2 \cup \phi_3 \cup \phi_4 \cup \phi_5 \cup \phi_6$. Note that there are four possible transitions from the first step to second step of the attack pattern. Any one of the four behaviours of interest would suffice to transit from the first state to the next one. The ϕ_6 transition ensures that if there is a *timeout* event at any point of the attack pattern, we will return to the initial state.

Sabotage by deleting sensitive files

In this attack pattern the insider logs in at an unusual hour and deletes an unusually large number of sensitive files. It is likely that the insider will try to obtain VPN credentials from a colleague to cover their tracks. Here, we provide an example of how to formalise attack patterns that may have more than one variation, meaning that the insider may omit some steps (usually because they do not choose to cover their tracks or because they already possess the appropriate access credentials as part of their job).

More specifically, we have the initial state S_0; transition to the final state S_f requires the observation of an alert regarding the deletion of files; transition to the first step S_1 requires the observation of a violation of the physical impossibility alert; to reach the the second state S_2, this pattern requires the observation of a log-on anomaly. The transitions in the Φ set are built from:

- $\phi_0 : S_0 \xrightarrow{(\langle User\ N,\ Physical\ Impossibility \rangle,\ \perp)} S_1$
- $\phi_1 : S_0 \xrightarrow{(\langle User\ N,\ log{-}on\ anomaly \geq Orange \rangle,\ \perp)} S_2$
- $\phi_2 : S_1 \xrightarrow{(\langle User\ N,\ log{-}on\ anomaly \geq Orange \rangle,\ 24\ hours)} S_2$
- $\phi_3 : S_2 \xrightarrow{(\langle User\ N,\ delete\ file\ anomaly \geq Orange \rangle,\ 24\ hours)} S_f$
- $\phi_4 : \forall\ S_n \in \Sigma - S_0,\ S_n \xrightarrow{timeout} S_0$

Finally, $\Phi = \phi_1 \cup \phi_2 \cup \phi_3 \cup \phi_4$. Note that there are two possible transitions from the initial state. One which leads to step one and an alternative which leads directly to step two. There is a 24 hour window to observe the VPN policy violation, a 24 hour window to observe the log-on anomaly, provided there was a VPN violation, and a 48 hour window to observe the same event from the initial state. The ϕ_4 transition ensures that, if there is a *timeout* event at any point of the attack pattern, we will return to the initial state. In the case where the insider may choose to use the credentials of a colleague, we will be knowingly raising an alert for the employee whose credentials were used. Further investigation will be required to identify the real insider. Currently that would be typically undertaken by a human-centric investigation, however, we observe that there may well be other online activities related to the insider which could provide metadata that can be automatically cross-correlated in order to provide some intelligence on who the insider actually is.

7 Conclusions and Future Work

There is a growing concern in both the academic community and industry that the risk which insider threats pose to organisations is even higher than external threats. Researchers and security practitioners have suggested various ways to deter and detect these threats, focusing on identifying security policies and developing anomaly detection systems. There are not many instances, however, where both approaches have been applied in tandem in a detection context. Such hybrid approaches can explore the benefits that clear and transparent policies offer to the security culture of organisations with corresponding monitoring -analytic techniques of network data in the form of tripwires.

In this paper we reflected on security policies that organisations have adopted to mitigate the problem of insider threat to build a library of widely accepted tripwires. We further presented a grammar for capturing such tripwires and validated our approach by formalising a series of common security policies. Our grammar is closely related to the detection system designed for the needs of the CITD system and provides the basis for automatic code generation for tripwire alerts. More specifically, we described a language that can formalise two types of tripwires: those triggered when a violation of a security policy occurs; and those triggered when evidence of a known attack pattern is found. It is our belief that the construction of such a framework will allow to unambiguously describe and share the body of knowledge and practice around insider threats. We hope that our framework will provide the opportunity to practitioners to implement tripwires in their operational contexts and enrich our grammar further based on their experiences.

We have engaged with the industry and started to apply our system to real data, obtained from three multi-national organisations operating in different environments. We intend, in the near future, to validate the effectiveness of the tripwires formalised in this paper by testing our system on real datasets, where there is a known case of insider activity. Another avenue of future work involves conducting interviews with security employees who are responsible for designing policies for organisations, in order to identify new tripwires which could be used to detect insider threats and further enrich our library. An interface based on the formal language that allows analysts to write their own bespoke tripwires and run these on our detection system or their own would also be of value.

The constantly changing landscape of insider threats and the dynamic legislative environment renders the use of transparent security policies more crucial than ever. When these policies are complemented with a tripwire and an anomaly detection system for insider threats, organisations can better manage and mitigate the risk from within. We hope that the provision of a grammar for describing tripwires is a decisive step towards providing enhanced protection from the harms caused by insiders. We will continue to evolve the library of tripwires and explore how to make this available within the security community.

Acknowledgements

This research was conducted in the context of a collaborative project on Corporate Insider Threat Detection, sponsored by the UK National Cyber Security Programme in conjunction with the Centre for the Protection of National Infrastructure, whose support is gratefully acknowledged. The project brings together The Department of Computer Science and The Said Business School at the University of Oxford, and the Department of Psychology at the University of Warwick.

References

[1] M. Bishop, D. Gollmann, J. Hunker, and C. W. Probst, *Insider Threats in Cyber Security*. Springer, 2010.

[2] H. Poll and A. Kellett, "Vormetric insider threat report," https://www.sans.org/reading-room/whitepapers/monitoring/insider-threat-mitigation-guidance-36307 [Online; Accessed on March 1, 2017].

[3] PriceWaterhouseCoopers, "Managing insider threats," http://www.pwc.com/us/en/increasing-it-effectiveness/publications/managing-insider-threats.html [Online; Accessed on March 1, 2017].

[4] Centre for the Protection of National Infrastructure, "Holistic management of employee risk," http://www.cpni.gov.uk/Documents/Publications/2012/2012021-homer.pdf [Online; Accessed on March 1, 2017].

[5] D. M. Upton and S. Creese, "The danger from within." *Harvard business review*, vol. 92, no. 9, pp. 94–101, September 2014.

[6] D. M. Cappelli, A. P. Moore, and R. F. Trzeciak, *The CERT guide to insider threats: how to prevent, detect, and respond to information technology crimes (Theft, Sabotage, Fraud)*. Addison-Wesley, 2012.

[7] Y. Chen and B. Malin, "Detection of anomalous insiders in collaborative environments via relational analysis of access logs," in *Proc. of the 1st ACM Conference on Data and Application Security and Privacy (CODASPY'11), San Antonio, Texas, USA*. ACM, February 2011, pp. 63–74.

[8] P. Parveen and B. Thuraisingham, "Unsupervised incremental sequence learning for insider threat detection," in *Proc. of the 2012 IEEE International Conference on Intelligence and Security Informatics (ISI'12), Washington, DC, USA*. IEEE, June 2012, pp. 141–143.

[9] E. Ted, H. G. Goldberg, A. Memory, W. T. Young, B. Rees, R. Pierce, D. Huang, M. Reardon, D. A. Bader, E. Chow *et al.*, "Detecting insider threats in a real corporate database of computer usage activity," in *Proc. of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data mining (KDD'13), Chicago, Illinois, U.S.A.* ACM, August 2013, pp. 1393–1401.

[10] J. R. Nurse, O. Buckley, P. A. Legg, M. Goldsmith, S. Creese, G. R. Wright, and M. Whitty, "Understanding insider threat: A framework for characterising attacks," in *Proc. of the 2014 IEEE Security and Privacy Workshops (SPW'14), San Jose, California, USA*. IEEE, May 2014, pp. 214–228.

[11] I. Agrafiotis, J. R. Nurse, O. Buckley, P. Legg, S. Creese, and M. Goldsmith, "Identifying attack patterns for insider threat detection," *Computer Fraud & Security*, vol. 2015, no. 7, pp. 9–17, July 2015.

[12] I. Agrafiotis, A. Erola, M. Goldsmith, and S. Creese, "A tripwire grammar for insider threat detection," in *Proc. of the 8th ACM CCS International Workshop on Managing Insider Security Threats (MIST'16), Vienna, Austria*. ACM, October 2016, pp. 105–108.

[13] National Institute of Standards and Technology, "Framework for improving critical infrastructure cybersecurity," http://www.nist.gov/cyberframework/upload/cybersecurity-framework-021214-final.pdf [Online; Accessed on March 1, 2017].

[14] Securities Industry and Financial Markets Association, "Overview of insider threat best practices guide," https://www.sifma.org/uploadedfiles/issues/technology_and_operations/cyber_security/insider-threat-best-practices-guide.pdf [Online; Accessed on March 1, 2017].

[15] B. Balakrishnan, "Insider threat mitigation guidance," https://www.sans.org/reading-room/whitepapers/monitoring/insider-threat-mitigation-guidance-36307 [Online; Accessed on March 1, 2017].

[16] L. Flynn, C. Huth, R. Trzeciak, and P. Buttles, "Best practices against insider threats for all nations," in *Proc. of the 3rd Worldwide Cybersecurity Summit (WCS'12), New Delhi, India*. IEEE, October 2012, pp. 1–8.

[17] National Counterintelligence and Security Centre, "National insider threat policy," https://www.ncsc.gov/nittf/docs/National_Insider_Threat_Policy.pdf [Online; Accessed on March 1, 2017].

[18] D. S. Eric Carter, "Security policies for Mid-Essex hospital services," http://www.meht.nhs.uk/about-us-/policies-and-guidelines/ [Online; Accessed on March 1, 2017].

[19] C. Mitchell, "Information security policy: NHS England," https://www.england.nhs.uk/wp-content/uploads/2013/06/ig-policy-1.1.pdf [Online; Accessed on March 1, 2017].

[20] Queen Mary, University of London, "Information security policies: Queen Mary, University of London," http://www.its.qmul.ac.uk/governance/policies/index.html [Online; Accessed on March 1, 2017].

[21] South West Yorkshire Partnership NHS Foundation Trust, "Information security policy: South West Yorkshire Partnership NHS Foundation Trust," http://www.southwestyorkshire.nhs.uk/about-us/corporate-information/policies/ [Online; Accessed on March 1, 2017].

[22] M. West, "Information security policy: The Queen Elizabeth Hospital," http://www.qehkl.nhs.uk/IG-Documents/information-security-policy.pdf [Online; Accessed on March 1, 2017].

[23] SPLUNK, "Detecting insider threats: How splunk software is used to safeguard financial data," http://www.splunk.com/pdfs/customer-success-stories/customer-profiles/detecting-insider-threats.pdf [Online; Accessed on March 1, 2017].

[24] SPLUNK, "Using splunk user behaviour analytics," http://www.splunk.com/en_us/products/premium-solutions/user-behavior-analytics.html [Online; Accessed on March 1, 2017].

[25] J. Y. Halpern and V. Weissman, "Using first-order logic to reason about policies," *ACM Transactions on Information and System Security (TISSEC)*, vol. 11, no. 4, pp. 21:1–21:41, July 2008.

[26] J. Boender, M. G. Ivanova, F. Kammüller, and G. Primiero, "Modeling human behaviour with higher order logic: Insider threats," in *Proc. of the 2014 Workshop on Socio-Technical Aspects in Security and Trust (STAST'14), Vienna, Austria*. IEEE, July 2014, pp. 31–39.

[27] F. Kammüller and C. W. Probst, "Invalidating policies using structural information," in *Proc. of the 2013 IEEE Security and Privacy Workshops (SPW'13), San Francisco, California, USA*. IEEE, May 2013, pp. 76–81.

[28] M. G. Ivanova, C. W. Probst, R. R. Hansen, and F. Kammüller, "Attack tree generation by policy invalidation," in *Proc. of the 9th IFIP International Conference on Information Security Theory and Practice (WISTP'15), Heraklion, Crete, Greece*, ser. Lecture Notes in Computer Science, vol. 9311. Springer, Cham, August 2015, pp. 249–259.

[29] T. Chen, F. Kammüller, I. Nemli, and C. W. Probst, "A probabilistic analysis framework for malicious insider threats," in *Proc. of the 3rd International Conference on Human Aspects of Information Security, Privacy, and Trust (HAS'15), Los Angeles, California, USA*, ser. Lecture Notes in Computer Science, vol. 9190. Springer, Cham, August 2015, pp. 178–189.

[30] J. Crampton and M. Huth, "Detecting and countering insider threats: Can policy-based access control help," in *Proc. of the 5th International Workshop on Security and Trust Management (STM'09), Saint Malo, France*, September 2009. [Online]. Available: https://www.doc.ic.ac.uk/~mrh/security/CramptonInsiderWorkshop.pdf

[31] G. Magklaras and S. Furnell, "Insider threat specification as a threat mitigation technique," in *Insider Threats in Cyber Security*, C. W. Probst, J. Hunker, D. Gollmann, and M. Bishop, Eds. Springer US, 2010, pp. 219–244.

[32] G. Magklaras, S. Furnell, and P. J. Brooke, "Towards an insider threat prediction specification language," *Information Management & Computer Security*, vol. 14, no. 4, pp. 361–381, 2006.

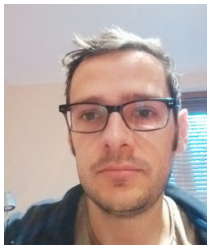
[33] P. A. Legg, O. Buckley, M. Goldsmith, and S. Creese, "Automated insider threat detection system using user and role-based profile assessment," *IEEE Systems Journal*, June 2015. [Online]. Available: https://doi.org/10.1109/JSYST.2015.2438442

[34] I. Agrafiotis, A. Erola, J. Happa, M. Goldsmith, and S. Creese, "Validating an insider threat detection system: A real scenario perspective," in *Proc. of the 2016 IEEE Security and Privacy Workshops (SPW'16), San Jose, California, USA*. IEEE, May 2016, pp. 286–295.

[35] T. A. software foundation, "Log files," https://httpd.apache.org/docs/1.3/logs.html [Online; Accessed on March 1, 2017].

Author Biography

Ioannis Agrafiotis is a Research Fellow in the Department of Computer Science at the University of Oxford. His research include risk analysis in the cyber domain, insider-threat detection, situational awareness and the usage of social media in the post-truth era. Ioannis received an EPSRC scholarship and completed a PhD in Engineering (Warwick, 2012), focusing on formal methods for information privacy. He also holds an MSc in Analysis, Design and Management of Information Systems from LSE (2008) and a BSc in Applied Informatics (University of Macedonia, 2006).



Arnau Erola is a cyber security expert with strong background in information privacy. He obtained a MSc in Computer Security from the University Rovira i Virgili (Tarragona) in 2009 and a PhD in 2013. Dr. Arnau Erola is currently a post doctoral researcher at Cyber Security Oxford at the University of Oxford, working in several projects on information privacy and defence systems. He is author of several international journal articles on online privacy and anonymity protocols to achieve it and intrusion detection mechanisms.

Michael Goldsmith is a Senior Research Fellow at the Department of Computer Science and Worcester College, Oxford. With a background in Formal Methods and Concurrency Theory, Goldsmith was one of the pioneers of automated cryptoprotocol analysis. He has led research on a range of Technology Strategy Board and industrial or government-funded projects ranging from highly mathematical semantic models to multidisciplinary research at the social-technical interface. He is an Associate Director of the Cyber Security Centre, Co-Director of Oxford's Centre for Doctoral Training in Cybersecurity and one of the leaders of the Global Centre for Cyber Security Capacity-Building hosted at the Oxford Martin School, where he is an Oxford Martin Fellow.

Sadie Creese is Professor of Cybersecurity in the Department of Computer Science at the University of Oxford. She is Director of Oxford's Cyber Security Centre, Director of the Global Centre for Cyber Security Capacity Building at the Oxford Martin School, and a co-Director of the Institute for the Future of Computing at the Oxford Martin School. Her research experience spans time in academia, industry and government. She is engaged in a broad portfolio of cyber security research spanning situational awareness, visual analytics, risk propagation and communication, threat modelling and detection, network defence, dependability and resilience, and formal analysis. She has numerous research collaborations with other disciplines and has been leading inter-disciplinary research projects since 2003. Prior to joining Oxford in October 2011 Creese was Professor and Director of e-Security at the University of Warwick's International Digital Laboratory. Creese joined Warwick in 2007 from QinetiQ where she most recently served as Director of Strategic Programmes for QinetiQ's Trusted Information Management Division.