Detecting stealthy attacks: Efficient monitoring of suspicious activities on computer networks

Kalutarage, H. , Shaikh, S.A. , Wickramasinghe, I.P. , Zhou, Q. and James, A.

Original citation & hyperlink:
Kalutarage, H. , Shaikh, S.A. , Wickramasinghe, I.P. , Zhou, Q. and James, A. (2015) Detecting stealthy attacks: Efficient monitoring of suspicious activities on computer networks. Computers & Electrical Engineering, volume 47 : 327–344
http://dx.doi.org/10.1016/j.compeleceng.2015.07.007

DOI 10.1016/j.compeleceng.2015.07.007
ISSN 0045-7906

Publisher: Elsevier

Detecting stealthy attacks: Efficient monitoring of suspicious activities on computer networks

Harsha K. Kalutarage[a,*], Siraj A. Shaikh[a,**], Indika P. Wickramasinghe[b], Qin Zhou[a], Anne E. James[a]

[a]*Digital Security and Forensics (SaFe) Research Group, Faculty of Engineering and Computing, Coventry University, Coventry, CV1 5FB, UK*
[b]*Department of Mathematical Sciences, Eastern New Mexico University, 1500 S Ave K Portales, NM 88130,US*

Abstract

It may take weeks or months before a stealthy attack is detected. As networks scale up in size and speed, monitoring for such attempts is increasingly a challenge; collection and inspection of individual packets is difficult as the volume and the rate of traffic rise. This paper presents an efficient method to overcome such a challenge. Data reduction has become an integral part of passive network monitoring, which could be motivated as long as it preserves the required level of precision. This paper examines the feasibility of employing traffic sampling together with a simple, but a systematic, data fusion technique for monitoring; and whether the design of the network affects on non-sampling error. Proposed approach is capable of monitoring for stealthy suspicious activities using 10%-20% size sampling rates without degrading the quality of detections.

Keywords: stealthy attacks, Bayesian, simulation, traffic sampling, anomaly detection

1. Introduction

Launching *stealthy attacks* is one of sophisticated techniques used by skillful attackers to avoid detection and can take months to complete the attack life cycle. Tools and techniques to launch such attacks are widely available. In order to detect stealthy activities it is necessary to maintain a long history of what is happening in the environment. Most systems cannot keep enough event data to track across extended time intervals for this purpose due to the performance issues and computational constraints [1, 2]. Decision to inspect each and every individual packet for security analysis may consume more resources at network

*Corresponding author
**Corresponding author
Email addresses: harshakumaralk@gmail.com (Harsha K. Kalutarage), aa8135@coventry.ac.uk (Siraj A. Shaikh)

devices for packet processing and more bandwidth for transmissions them to collection points [3]. Sophisticated computing systems may be required for analysis and storage such a huge volume of data. The performance of network can be affected by such overheads and hence to quality of the service. All these facts motivate for a *data reduction* which could be motivated as long as it preserves the required level of precision for the monitoring objectives which can be either traffic engineering, accounting or security specific.

This paper presents a study for an efficient monitoring scheme for stealthy attacks on computer networks which can consider as an early warning system. *Traffic sampling* is employed together with a simple *data fusion* technique to propose the algorithm which applies over the sampled traffic. The study has two objectives. First, investigating the feasibility of proposed method for stealthy activity monitoring; and secondly, examining whether design of the network affects on detection. The rest of the paper is organised as follows. Section 2 provides a brief overview of intrusion detection in computer systems, and explains why conventional methods which are largely developed for rapid attacks cannot be employed in stealthy activity monitoring. Section 3 presents a monitoring algorithm which identifies Bayesian approach as a method for information fusion. Sampling technique employed by the monitoring scheme is presented in Section 4. Section 5 presents a methodological way to trace anonymous stealthy activities to their approximate sources. Experimental design is presented in Section 6. Sections 7 presents experimental outcomes. Related literature is presented in Section 8. Finally, conclusions are drawn in Section 9 where further work is also suggested.

2. Security Monitoring

Computer systems are dynamic systems having many components such as clients, servers, switches, firewalls and Intrusion Detection Systems (IDSs). At each time interval these components produce large amounts of event based data which, in principal, can be collected and used for security analysis. The signature elements of an attack is scattered *spatially* and *temporally*, and often embedded within the totality of events of the distributed systems, and *motivation*[1] and *source*[2] behind some events are not always certain. In addition there are number of monitoring obstacles in such an attack scenario: evidence scarcity (weak), colluded activities, large attack surfaces, variety of users and devices, high volume high speed environments, normal variations to node behaviours and anomalies keep changing over the time [4, 5]. Due to the above challenges most of the existing anomaly detection techniques solve a specific formulation of the problem which induces by various factors such as data types and types

[1] 1. An alert of multiple login failures, 2. An execution of cmd.exe 3. An abuse of legitimate credentials either by individuals or malware.

[2] Using various proxy methods and zombie nodes. manipulation of TCP/IP elements, using relay or random routing.

of anomalies of interested, and encourage unsupervised anomaly detection techniques [6]. Proposed monitoring scheme in this paper is an effort to address most of above obstacles in one solution.

In signature based intrusion detection an attack scenario signature is needed to distinguish a given attack (say A) from other attacks (B and C) and from normal network activities. When a stealthy attack is progressing the critical challenge is how to correlate these events across spatial and temporal spaces to track various attack scenarios such as A, B and C. The detection accuracy relies on the accuracy of scenario signature as well as the accuracy of event correlation [7]. Maintaining state information of every packets and comparisons between current packets and previous all packets are needed in event correlation. Most systems cannot keep enough event data to track across extended time intervals to do this when a stealthy attack is progressing. As a result the scarcity of attack data within a short period of time allows a stealthy attacker to go undetected hiding her attempts in the background noise and other traffic. Hence using signature detection techniques for stealthy activity monitoring is a challenge.

Proposed monitoring algorithm in this paper is anomaly based. Finding nonconforming patterns or behaviours in data is referred to as anomaly detection. An intrusion is different from the normal behaviour of the system, and hence anomaly detection techniques are applicable in intrusion detection domain [6]. Intrusive activity is always a subset of anomalous activity is the ordinary belief of this idea [8, 9]. When there is an intruder who has no idea of the legitimate user's activity patterns, the probability that the intruder's activity is detected as anomalous is high. This has been formulated in [10] as a pattern recognition problem. When the actual system behaviour deviates from the normal profiles in the system an anomaly is flagged. Information fusion would be a possible method for data reduction. However given the nature of problem domain, anomaly detection techniques need to be computationally efficient to handle large sized of inputs. Hence considering any complex method, e.g. methods like Principal Components Analysis [11], for information fusion is ignored as they introduce extra computational overheads which aimed to minimise as much as possible in this work.

3. Monitoring Algorithm

The monitoring algorithm is inspired by previous work [12] which is inspired by [13]. It is an incremental approach which updates normal node profiles dynamically based on changes in network traffic (events). If some aberrant changes happen in network traffic over the time, it should be reflected in profiles as well and suspicious activities can be raised based on that profiles is the basic assumption. The algorithm has two functions: *profiling* and *analysis*.

3.1. Profiling

The profiling is the method for evidence fusion across space and time by updating node profiles dynamically based on changes in evidence. Simply put, it computes a suspicion score for each node in the system during a smaller time window w and that score is updated as time progresses to compute a node score for a larger observation window W. By just looking at an alert generated by an event it is impossible to simply judge the *motivation* (cause) behind it. Other contextual information can be used to narrow down the meaning of such an event [14]. For example, suspicious port scanning activity may have the following characteristics: a single source address, one or more destination addresses, and target port numbers increasing incrementally. When fingerprinting such traffic analysts examine multiple elements (multivariate) and develop a hypothesis for the cause of behaviour on that basis. A similar manner (multivariate approach) can be followed in the profiling to acknowledge the motivation uncertainty. Note that What and Why are two different questions. Projecting Why into What based on your own guesses is methodologically irresponsible. Hence it needs a simple, but systematic, approach to profile suspects based on motivation of activities instead of number of activities (what you see). In other words, security events must be analysed from as many sources as possible in order to assess threat and formulate appropriate responses. Extraordinary levels of security awareness can be attained by simply listening to what its all indicators are telling you [15]. Note that proposed profiling technique in this paper fuses information gathered from different sources into a single score for a minimum computational cost. It reduces data into a single value which is important to maintain information about node activities for a very long observation period W. A multivariate version of simple Bayes' formula is used for this task.

3.2. The Bayesian paradigm

The posterior probability of the hypothesis H_k given that E is given by the well-known Bayes formula:

$$p(H_k/E) = \frac{p\left(E/H_k\right).p(H_k)}{p(E)} \tag{1}$$

The hypothesis for the monitoring algorithm is built as follows. Let H_1 and H_2 be two possible states of a node in a network and define H_1 - the node acts as an attacker and H_2 - the node does not act as an attacker. Then H_1 and H_2 are mutually exclusive and exhaustive states. $\mathrm{P}(H_1)$ is an expression of belief, in terms of probability, that the node is in state H_1 in the absence of any other knowledge. Once obtained more knowledge on the proposition H_1 through multiple information sources (m indicators), in the form of evidence E={e_1,e_2,e_3,...,e_m} on attack surface including the human element, the belief can be expressed in terms of conditional probabilities as $p(H_1/E)$. Using the Bayes' theorem in Equation 1 and assuming statistical independence between information sources:

$$p(H_1/E) = \frac{\prod\limits_{j=1}^{m} p(e_j/H_1).p(H_1)}{\sum\limits_{i=1}^{2}\prod\limits_{j=1}^{m} p(e_j/H_i).p(H_i)} \tag{2}$$

When likelihoods $p(e_j/H_i)$ and prior $p(H_i)$ are known, the posterior $p(H_1/E)$ can be calculated for a given w. These posterior terms $p(H_1/E)$ can be accumulated by time to use as a metric to distinguish suspected nodes from other nodes during a W. Note that distinct types of information sources such as signature based IDSs, anomaly detection components, file integrity checkers, SNMP-based network monitoring systems can be used for this purpose. Hence the assumption on statistical independence above is reasonable. Any influence/interested technical and socio-technical indicators of changes in behaviour (e.g. changes in access patterns, differences in use of language, typing patterns, transferring large amounts of data onto or off the node, etc; if human actors are involved) can be included as input variables (i.e. elements of E) in the profiling algorithm as long as such indicators operate statistically independent. Extending proposed approach to a very large scale attack surface is easy since it is a matter of adding a new indicator (attack vector) in E. Existing domain knowledge will serve to enhance the performance of this monitoring algorithm since it takes advantage of prior knowledge about the parameters. Which is especially useful when technical data is scarce. However prior and likelihoods are the most critical parameters to this approach since Bayes' factors are sensitive to them. Proposed monitoring algorithm would be useful in monitoring threats listed in Table 1. The potential threats and their indicators in Table 1 is not exhaustive and for illustrating purpose only.

3.3. Analysis

The analysis comprised of detecting anomalous profiles in a given set of node profiles. If attacker activity pattern is sufficiently reflected by profiles then detecting anomalous profiles would be sufficient to identify attackers. This work uses a statistical method to detect anomalies. An anomaly is an observation in a dataset which is suspected of being partially or wholly irrelevant because it is not generated by the stochastic model assumed for that dataset is the underlying principle of any statistical anomaly detection technique [17]. Such techniques are based on the key assumption that normal data instances occur in high probability regions of a stochastic model, while anomalies occur in the low probability regions of the stochastic model [6]. Based on these concepts *Peer* and *Discord* analysis is proposed in this work for detecting stealthy activities in a given set of node profiles. Both techniques acknowledge the fact that baseline behaviour on networks is not necessarily stable, for example, operational or exercise deployments often mean the behaviour of nodes will potentially change dramatically. Hence, a defence method that is effective today may not remain effective for tomorrow, and any novel algorithm should account for this level of complexity. Proposed approach evolves the baseline behaviour by the time according to the other network parameters and their current states.

Scenario	Brief Description	Potential Indicators to use in E
Distant Admin	Unauthorised admin like access to servers and workstations from distant (geographically) locations. There is a "motivation" uncertainty behind this kind of behaviour as legitimate users and administrators frequently access enterprise network from endpoints which are geographically far away from the organisation.	Distance between hosts, Total bytes transferred, Service being used (e.g. RDP, SSH, VNC, telnet), Protocol, etc.
Data exfiltration	Large uploads to remote servers, once an attacker breached a network data ex-filtration can be difficult to prevent and detect as they use stealthy methods to get data back to their infrastructure during very long time periods. Typically they exfiltrate the data in batches across commonly used channels (e.g. http(s)) permitted by firewalls. Detection of this activity as early as possible would be beneficial to prevent further damage to the organisation.	Service being used (e.g. http(s)), protocol being used, session duration, amount of traffic exchanged, ratio of bytes exchanged to/from, etc.
Port Scanners	Slow randomised port scans which can be a part of an attacker's reconnaissance efforts.	Number of zero byte TCP packets/sessions, Number of one-sided UDP communications, number distinct server ports touched, number of host touched, etc.
Protocol Abuse	A popular type of tunnel communication through a common service port (e.g. ports 80 -HTTP, 53 - DNS, 443 - HTTPS) since these ports are not blocked by firewalls and other network security devices for business-critical functions.	known common ports and their expected traffic types, session duration, amount of traffic exchanged, etc.
Beacon	Monitoring for slow beacons from infected hosts to C2 servers. There is a "motivation" uncertainty behind this kind of behaviour as some innocent programs (e.g. some types of DNS traffic, regular software updates, anti-virus definition updates) also exhibit recurrent communication. Malware may try to hide behind such innocent activities (network noise). By continues monitoring helps spotting them before they can do any real damage.	traffic types (e.g. HTTP(S)), security level, version of the OS, OS is patched or not, type of the app generating network traffic, etc.

Table 1: Possible real world network scenarios that proposed method would be useful to apply [16]

3.3.1. Peer analysis

Aggregating posterior probability terms in Equation 2 over the time helps to accumulate relatively weak evidence for long periods. These accumulated probability terms $\sum_t p(H_1/E)$ (t is time), known as *node scores*, can be used as a measurement of the level of suspicion of a given node at any given time with respect to her peers as follows. A given set of node profiles, e.g. profiles corresponding to a similar peer group, is a uni-variate data set. Hence it is possible to use the uni-variate version of Grubb's test [18] (maximum normed residual test) to detect anomalous points in the set, subject to the assumption that *normal* node profiles in a given set follow an unknown Gaussian distribution [19]. The set-up where it has the distribution is very well a mixture of Gaussian. Because testing of the hypothesis for any given time is a Bernoulli trial in this work. Accumulated Bernoulli trials makes a Binomial distribution which can be approximated by a Normal distribution. For each profile score ω, its z score is computed as:

$$z = \frac{\omega - \bar{\omega}}{s} \tag{3}$$

Where $\bar{\omega}$ and s are mean and standard deviation of the data set. A test instance is declared to be anomalous at significance level α if:

$$z \geq T = \frac{N-1}{\sqrt{N}} \sqrt{\frac{t^2_{\alpha/N,N-2}}{N-2+t^2_{\alpha/N,N-2}}} \tag{4}$$

where N is the number of profile points in the set, and $t_{\alpha/N,N-2}$ is the value taken by a t-distribution (one tailed test) at the significance level of $\frac{\alpha}{N}$ and degrees of freedom $(N-2)$. The α reflects the confidence associated with the threshold and indirectly controls the number of profiles declared as anomalous [6]. Note that the threshold T adjusts itself according to current state of a network. This is a vertical analysis to detect one's aberrant behaviour with respect to her peers. In other words it compares each node's activity changes against to activity changes of her peer group. Hence it is called as *peer analysis* in this paper. This analysis technique accounts for regular variations such as diurnal, familiarity and ageing.

Looking at one's aberrant behaviour within similar peer groups (e.g. same user types, departments, job roles, etc.) gives better results in terms of false alarms than setting a universal baseline [20, 21]. Hence first classifying similar nodes into peer groups, based on behaviour related attributes/features, and then applying the monitoring algorithm is recommended. Investigations for suitable classification algorithms for this task is left as a future work.

3.3.2. Discord analysis

When a stealthy attack is progressing, malicious activities are occurring according to an on-off pattern in time. As a result, lack of agreement or harmony between points in the profile sequence of a given node can occur in a similar or different on-off fashion. This type of anomalies are known as discords [22].

In a stealthy attack environment, discords are random time context and peer analysis technique itself is not sufficient to detect them if the progression rate of malicious activities is far lower than the similar innocent activities. The objective of discord analysis in this work is to detect sub-sequences within a given sequence of profiles which is anomalous with respect to the rest of the sequence. Problem formulation occurs in time-series data sets where data is in the form of a long sequence and contains regions that are anomalous. The underlying assumption is that the normal behaviour of the time-series follows a defined random pattern, and a sub-sequence within the long sequence which does not conform to this pattern is an anomaly. In general, the purpose of this analysis is to detect one's aberrant behaviour with respect to her own behaviour regardless of her peers. Following method is proposed for discord analysis.

At the $(t-1)^{th}$ time point, using an Auto-regressive integrated moving average model $ARIMA(p, d, q)$ [23] which describes the auto-correlations in the data, 95% Confidence Interval (CI) for the t^{th} profile score is predicted. If the observed profile score at time t lies outside of the predicted CI then absolute deviation of the profile score from CI is calculated. This deviation is used as a measure of non-conformity of a given profile score to the pattern of its own sequence (group norms). These deviations average out over time to calculate the *anomaly score* for a given node. Note that this anomaly score is the average dissimilarity of profile scores with its own profile sequence of a node. This dissimilarity occurs randomly from time to time due to the deliberate intervention of the attacker. The length of the ARIMA model (i.e. n - number of previous points to be used) is critical as containing anomalous regions in input sequence makes difficult of creating robust model of normalcy. Note that keeping the length of the ARIMA model less than the minimum of time gaps between two consecutive attack activities will give better results. However since the time gap between two consecutive attack activities is unknown in advance, using a smaller observation window (i.e. slicing whole observation period into many smaller parts as much as possible) to generate short time profiles would be the better. A node does exhibit sudden changes in behaviour when compared to its past behaviour is not necessarily suspicious as it could be a regular variation of the node behaviour [20]. Proposed Discord analysis technique considers such variations as completely legitimate as it monitoring for *changes to the changing pattern* of node behaviour.

The key challenge for anomaly detection in network security domain is that the huge volume of data, typically comes in a streaming fashion, thereby requiring on-line analysis. It is essential to employ a data reduction method to overcome large-scale data handling. Employing statistical sampling would be a possible method. Despite the benefits, there is an inherent tension and debate of using traffic sampling for security specific tasks. Obviously, signature based detection methods can be seriously affected by sampling as selection of a subset of signature elements would not be sufficient to recognise a predefined pattern in a signature definition database. But in anomaly based detection, should all traffic still need to be investigated? In the abstract view, an anomaly is a deviation of a computed statistic from a norm of the normal statistics. If sampling changes the statistics of normal and anomalous traffic equally, it is reasonable to hypothesise that detection would not be affected by the sampling rate. This hypothesis is also investigated in this paper.

4. Employing sampling

Network data constitutes a potentially unlimited population continuously growing up by the time. Using multi-stage sampling with stratification is usual in large populations. This ensures that observations are picked from each of strata, even though the probability of being selected items from some stratus are very low when using simple random sampling (SRS). This feature is very useful in a security specific view. Hence, given a smaller observation window w, the traffic is sampled using the Stratification sampling technique with optimum allocation method. This sampling technique has been designed to provide the most precision for the *least cost*. If h is a traffic stratum, the best sample size n_h for stratum h during a w is given by:

$$n_h = n.\frac{[\frac{N_h.s_h}{\sqrt{c_h}}]}{\sum \frac{N_i.s_i}{\sqrt{c_i}}} \tag{5}$$

where n_h-sample size for stratum h, n-total sample size, N_i-population size for stratum i, s_i-standard deviation of stratum i, and c_i-direct cost (in terms of time, bandwidth, and computational resources) on the collection infrastructure to sample an individual element from stratum i. Note that the direct cost should be in a common unit (CU) of measurement for the amount of computational cost spending on different parameters. The time, bandwidth, memory or processor requirements that constitutes one common unit (1CU) varies based on which requirement is being measured, and how each parameter is critical and scarce to the network. Hence definition of such a unit (CU) would be subjective. For instance one can define: 1CU is memory equivalent of 128MB, 1CU is bandwidth equivalent of 56KBPS, 1CU is CPU-Time equivalent of 100 nsec etc. International unit (IU) in pharmacology is a well-known example for a similar approach for a common unit of measurement for the amount of a substance [24]. The main advantage of above sampling technique is producing the most representative sample of a population to the least cost. Hence it is the ideal sampling technique to employ with the problem as "cost" parameter can be minimised, subject to the required precision, to obtain a light-weighted monitoring scheme. The rule of thumb in stratification sampling that a population should not consist of more than six strata can be changed even into hundreds given the millions of observations in the population in this domain. Traffic classification is employed to establish the strata. Using a basic classification technique (e.g. using L4/L3 access lists and Protocols) would be enough. Stratification ensures that each traffic type is adequately represented. The SRS technique is used to select a n_h size sample from a given stratum h for a w. Random sampling techniques have a distinct advantage over other alternative methods for data reduction. It allows retention of arbitrary details while other methods for data reduction (e.g. filtering and aggregation) require the knowledge of the traffic features of interest in advance.

Each element of the population having a non-zero probability of selection is a preliminary condition for any random sampling techniques. Sampling traffic from backbones or edge routers seriously violates this condition in terms of security specific view, though it is sufficient for Traffic engineering and Accounting tasks. Since it ignores consideration of traffic within same broadcast domains, it ignores potential insider activities as well. Therefore in this work traffic is sampled at each broadcast domain, but considering the incoming traffic only. All outgoing traffic to any external network is considered as a separate broadcast domain for the purpose of traffic sampling. Considering incoming traffic only avoids selection of a given unit (packet or flow) twice for inclusion in a sample at source and destination points.

5. Tracing the Source

A common problem with many analysis tools and techniques today is that they are simply not designed for purposes of attribution[25]. Attribution of cyber activity - "knowing who is attacking you" or "determining the identity or location of an attacker or an attacker's intermediary"- is naturally a vital ingredient in any cyber security strategy [26, 27]. Although current approaches are capable of alarming suspicious activities, most of them are not suitable for this information age because when computers are under attack "who" and "why" are frequently unknown [28, 29].

The localization process becomes evermore difficult when the attacker employs various proxy methods and zombie nodes (e.g. bots), Manipulation of TCP/IP elements (e.g. IP Spoofing), using relay or random routing (e.g. Tor networks) approaches can help an attacker protecting her location. Proliferation of weakly encrypted wireless networks could also help an attacker getting anonymous locations. Tracing packets back to the source hop by hop is required in identifying sources of anonymous activities. This section presents a methodological way to trace such activities to their approximate sources by extending the above monitoring algorithm. The tracing algorithm has two functions: *tree formation* and *tree traversal*. Tree formation builds an equivalent tree structure for a given attack scenario. It enables tree traversal to move towards the attacker's physical source.

5.1. ***Tree formation:***

If the topological information is available, Tree formation is performed as follows. The victim node is the starting point. The Gateway node to victim is considered as the root of the tree and all immediate visible nodes (either internal or external) to the root are considered as children of the root. If a given child is a host node in the network then it becomes a leaf of the tree. If it is a gateway then it becomes a parent node of the tree and all immediate visible nodes to that node are attached as its children. This process is continued until the entire topology is covered (see Figure 22).

input : Topological information together with victim's location
output: Tree structure for the given attack scenario
Initialize the tree ϑ to have the root as the gateway of the victim;
List all nodes into the list τ;
/* attached each node to the tree*/;
tree-construction(ϑ,τ);
/*ϑ - Tree;
, ω - A node*/;
foreach *node* ω *in* τ **do**
    if *num-of-hops-between(ϑ,ω)==1* **then**
        insert ω into ϑ;
    end
end
foreach ϑ.*child* **do**
    tree-construction(ϑ.child,τ)
end

Algorithm 1: Tree formation for a given attack scenario.

5.2. Tree traversal:

Once the equivalent tree structure is built, *channel profile* score (z_{kt}) should be computed for each path of the tree at each step of the tree traversal algorithm as shown in Equation 7. Let

$$c_{kt} = \frac{\sum_{t} p(H_k/E)}{n_k} \tag{6}$$

where n_k is the number of nodes behind k^{th} channel. Then

$$z_{kt} = \frac{c_{kt} - \bar{c_t}}{\sigma_t} \tag{7}$$

is the Z-score of channel k at time t. where $\bar{c_t} = \frac{\sum_i c_{it}}{n}$, $\sigma_t = \sqrt{\frac{\sum_i (c_{it} - \bar{c_t})^2}{n-1}}$, and $i = 1, 2, 3, ..., n$.

To traverse a non-empty tree, perform the following operations recursively at each node, starting from the root of the tree, until suspected node is found.

1. Visit the parent node
2. Compute channel scores for all children of the parent
3. Traverse the highest channel scored sub tree if that score is above the threshold (if an attacker node is found backtrack to the parent)

4. Traverse the next highest channel scored sub trees (only sub trees above or around threshold and/or significantly deviated from rest of nodes of same parent)

The algorithm continues working towards a built tree node by node, narrowing down the attack source to one network and then to a node. At this point it is possible to run more standard trace back methods by contacting the entity which controls that network if it is beyond the analyst's control.

```
input  : A Tree constructed for anonymous stealthy attack scenario
output: A node where attacker is located
proposed-traverse(ϑ);
while not found do
  | visit node ω;
  | if node ω is a leaf then
  |   | return;
  | else
  |   | profile all children of node;
  |   | proposed-traverse(node.top_scored_child);
  |   | proposed-traverse(node.next_scored_child);
  | end
end
```

Algorithm 2: Tree traversal for a given tree.

6. Experiments

A series of experiments were conducted simulating stealthy suspicious activities in simulated networks to evaluate the proposed approach in this paper. Simulating such activities on a real network certainly gives more realistic conditions than in a simulated network. However practical constraints of the project keep away using a real world network for this purpose. Network simulator $NS3$ [30] is used to build a network topology (see Figure 1) consisting of a server farm and number of subnets of varying size. Table 2 presents a summary of specifications of event generation in simulated experiments.

A Poisson arrival model with inter-arrival time gap between two consecutive events as an exponential was assumed for events generation. Each simulation is run for a reasonable period of time to ensure that enough traffic is generated. Attackers are located at nodes in subnets. Suspicious and benign traffic were generated within and between subnets to simulate both attack and legitimate activities. Four types of suspicious activities (rate denoted by λ_a, a =1,2,3,4. in Table 2) was simulated. A stealthy attack is defined as a predefined sequence of such suspicious events executing an on-off manner. During the off period attack node acts as a healthy node. Note that "Noise" in table 2 represents the Suspicious events generated by healthy nodes, but at different rates $\lambda_n, n = 1, 2, 3, 4$. It was ensured to maintain $\lambda_a \in \lambda_n \pm 3\sqrt{\lambda_n}$ and $\lambda_n (\leq 0.1)$ sufficiently smaller for

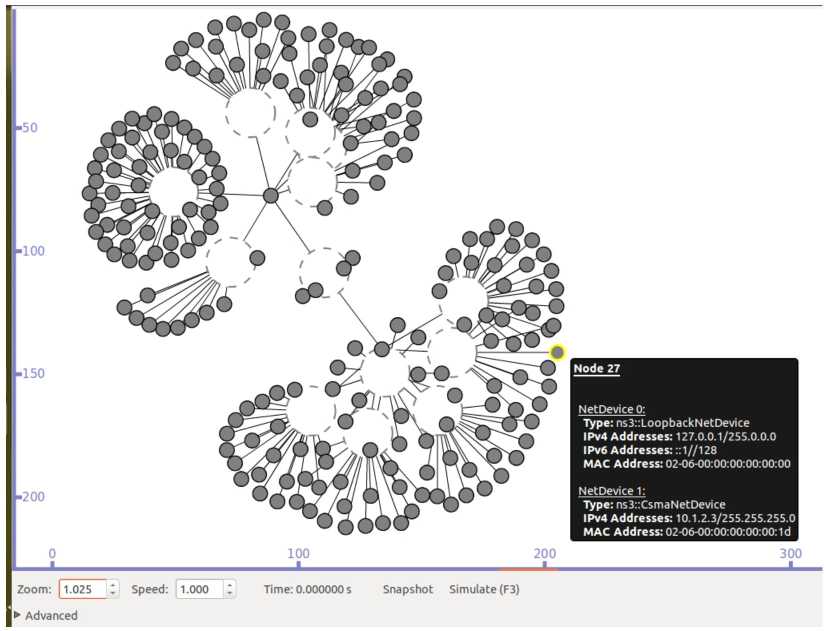

Figure 1: A screen-shot of a network topology used for experiments.

all experiments to characterise stealthy suspicious activities which aim at staying beneath the threshold of detection and hiding behind the background noise. The idea to use the above relationship for generating attacker activities was to keep them within the *normality range* of innocent activities (i.e. background noise). $\sqrt{\lambda_n}$ is the standard deviation of rates of suspicious events generated by normal nodes.

Though it did not produce all signature elements needed to characterise real attacks, representation of *suspicious events* by a subset of such characteristics (parameters) was sufficient to this work as its focus on temporal and spatial aspects of events arrivals. Note that traffic classification is sufficient to the proposed sampling method in this work, and does not require attack classifications.

Node	Event	Model	Parameters	Duration (s)	Repetitions
Attack	Legitimate	Poisson	μ_i, i=1,2,3,...,10.	3600*12*60=2592000 or above, scores are updated at every minutes (w=60s)	Between 1-100
	Suspicious		λ_a, a=1,2,3,4.		
Healthy	Legitimate		μ_i, i=1,2,3,...,10.		
	Noise		λ_n, n=1,2,3,4.		

Table 2: A summary of specifications of event generation

Basic payload information, i.e. L4/L3 access lists and Protocols such as http, ftp, udp and arp, was used for traffic classification. Traffic which cannot identify using basic payload information was pooled into a common stratum. A simple R [31] script was written to sample packets as described above. c_i in Equation 5 is set to a constant value as there is no significant difference of the cost between different type of traffics (stratum) for inclusion in a sample in simulations. Visible source of an event is always considered as the true source for experiments in this work. Prior probabilities and Likelihoods are assigned as described below.

$$p(H_1) = \frac{1}{2} = 0.5 \tag{8}$$

Equation 8 suggests there is a 50% chance for a given node to be a stealthy attacker. However, this is not the case in many situations. In networks, one node may have a higher prior belief of being suspicion than another. Since prior probabilities are based on previous experiences, $p(H_1)$ can be judged based on information gathered from contextual analysis. However if there is no basis to distinguish between nodes or groups of nodes, equally likely (i.e. same probability of occurring) can be assumed. For the experiment presented in this paper, first followed the equally likely assumption, and prior probabilities were assigned as in equation 8. Then the posterior probability of a given node at time $t-1$ is used as the prior of the same node at time t when time is progressing. This lets prior probabilities to adjust itself dynamically according to suspicious evidence observed over time.

$$p(e_j/H_1) = k_j \tag{9}$$

Equation 9 expresses the likelihood of producing event e_j by a subverted node. For the purpose of demonstration different, but arbitrary, values ($\leq$1) were assigned for k to distinguish different type of events (e_j) produced for the simulation. Likelihoods for real world implementation can be estimated as follows. If e_j is an event resulting from a certain type of known attack (e.g. a UDP scan or LAND[2] attack), then k can be assigned to one. However, k cannot always be one, as described in Section 2, as there are some suspicious events (e.g. an alert of multiple login failures) that can be part of an attack signature as well as originate from normal network activities. The question is how to estimate $p(e_j/H_1)$, i.e. the true positives, if e_j becomes such an observation. One possible solution would be to use existing IDS evaluation datasets to estimate true positives. Estimating likelihoods for real world implementation is feasible, and [32] is a good example for that which provides a detailed description of the likelihood estimation in insider detection.

According to [13], in some cases, the historical rate of occurrences of certain attacks is known and can be used to estimate the likelihood that certain events derive from such attacks or it may be sufficient to quantify these frequencies by an expert in a similar way to estimating risk likelihoods to an accuracy of an order of magnitude. Note that [13]'s claim is completely theoretical as it follows

[2]A Denial of Service (DoS) attack which sets the source and destination information of a TCP segment to be the same. A vulnerable machine will crash or freeze due to the packet being repeatedly processed by the TCP stack.

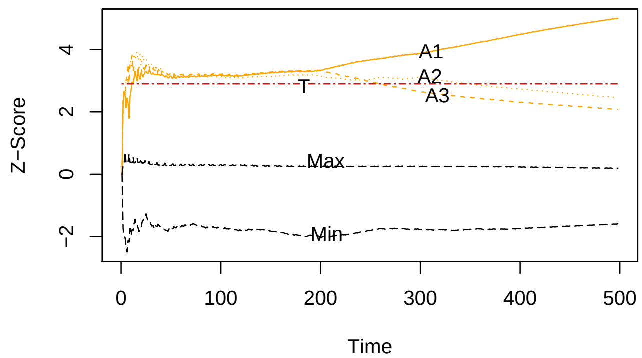

Figure 2: Z- Score graphs are sensitive to node behaviour.

the *Subjectivist*[3] interpretation of probability theory [33]. According to [14], the biggest challenge is the absence of large publicly available data sets for research and comparisons, but within an organization it is entirely possible to empirically analyse day-to-day traffic and build statistical models of normal behaviour.

7. Results

In this section, experimental results are presented. Graphical forms (e.g. Z-Score graphs) are using to present information. Visualisation helps to quickly recognise patterns in data.

7.1. Peer Analysis Outcomes

To investigate whether proposed Z-score graphs reflect the behaviour of nodes, three attacker nodes were located in a 50 size subnet. All others were innocent. Two out of three attackers stopped their attack activities at 200 and 300 time points respectively. Figure 2 presents the outcome, where $A1$, $A2$ and $A3$ are attacker nodes while Min and Max are the minimum and maximum Z-scores of normal nodes. T is the Grubbs' critical value (threshold). If an attacker node changed its behaviour, the corresponding z-score graph (see $A2$ and $A3$ in Figure 2) responses to that behaviour by changing its direction.

Peer analysis technique was tested against 24 test cases varying the subnet size between 25 and 250 and the number of attackers between 0 and 7. Peer analysis technique was capable of detecting stealthy attackers in all cases. Only

[3] There are three fundamental interpretations of probability: Frequentest, Propensity and Subjectivist. In Subjectivist, probability of an event is subjective to personal measure of the belief in that event is occurring.

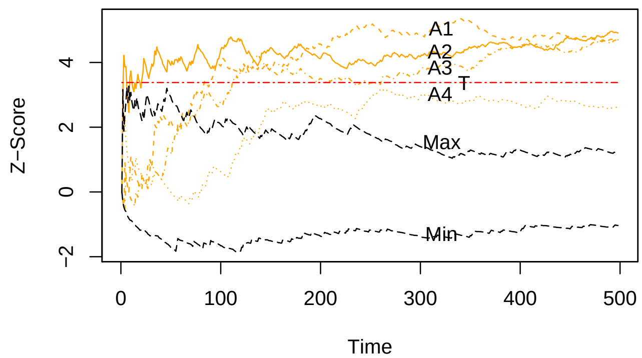

Figure 3: Z-Scores of node profiles for test case 16.

one case where four stealthy attackers were located in a hundred size subnet is presented in Figure 3. In Figure 3, nodes corresponding to $A1$, $A2$, $A3$ and $A4$ denote attackers. Min and Max denote the minimum and the maximum Z-scores of normal nodes at each time point. Aberrant node profiles A1, A2, A3 and A4 in Figure 3 always corresponded to the four stealthy attackers located in the subnet. They are above or near the threshold (T), and most importantly, there is a clear visual separation between the set of normal nodes and anomalous nodes. Hence it is possible to recognise stealthy suspicious activities using the proposed method.

Behaviour of the proposed approach in best and worst cases is also investigated. There were no attacks in best cases while all nodes were subverted in worst cases. Similar graphs, as shown in Figure 4, were obtained for both cases. Almost all the nodes are nearly below the threshold (T), and none of nodes can be seen separated from the majority. In a situation where monitoring system depends only on peer analysis technique and has seen similar graphs as in worst (or best) cases, it is safe to assume that all nodes are subverted (instead of assuming free of attackers) and doing further investigations on one or two nodes to verify. If investigated nodes are attackers, it is reasonable to consider all nodes are attackers or vice versa. However, note that Discord analysis technique is capable of detecting attackers in worst case too.

7.2. Discord Analysis Outcomes

Discord analysis technique was tested against number of test cases used for peer analysis, in addition to testing it against a special test case defined as follows. In a stealthy attack environment, discords are random time context and peer analysis technique itself would not be capable to detect them if the progression rates of malicious activities are far lower than the rates of similar

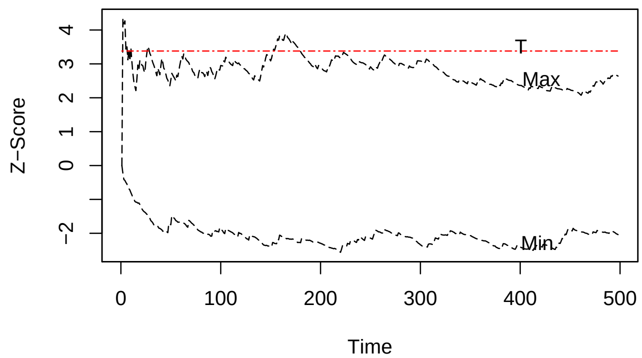

Figure 4: Z-Scores of node profiles for test case 7.

innocent activities. Therefore a small subnet consisting of five nodes including one attacker was set-up in a subnet. The attacker's activity rate was decreased until observing a node score graph like in Figure 5 where peer analysis technique itself failed to detect the attacker. In Figure 5, the attacker which is denoted by the red dotted line always keeps a very low profile score than all innocent nodes denoted by other lines (see magnified version in Figure 6). As it is seen in Figures 5 and 6, the attacker hides behind the normal nodes, and since the attacker's profile score is far lower than all normal nodes it is not detected by the peer analysis technique. The randomness of event generation can also be seen from Figure 6.

Discord analysis is capable of detecting the attacker very well in this case. First using an ARIMA(p, d, q) model 95% CI is predicted for each node in the network (see Figures 7 and 8 which are created for the attacker node and a normal node respectively). Then at each time point, anomaly score for all five nodes were calculated and converted them to Z-scores and plotted against the time line as in Figure 9. Twenty five previous points was used as the length of the ARIMA model in this case. In Figure 9, the node corresponded to A denotes the attacker. Min and Max denote the minimum and the maximum Z-scores of anomaly scores of normal nodes at each time point. T is the Grubbs' critical value (threshold) for a single outlier. As it is obvious in Figure 9 attacker node is distinguished from innocent nodes.

7.3. Network parameters

This section investigates how different network parameters: traffic volume, subnet size and number of attackers affect on monitoring of stealthy activities.

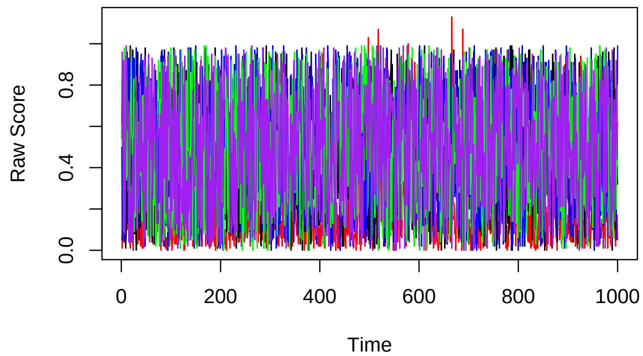

Figure 5: Hiding behind innocent nodes (See magnified version in Figure 6)

.

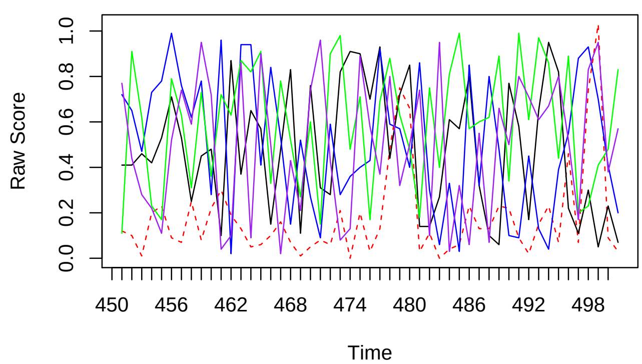

Figure 6: Magnified version of Figure 5 - the red dotted line denotes the attacker, all other lines denote innocent nodes.

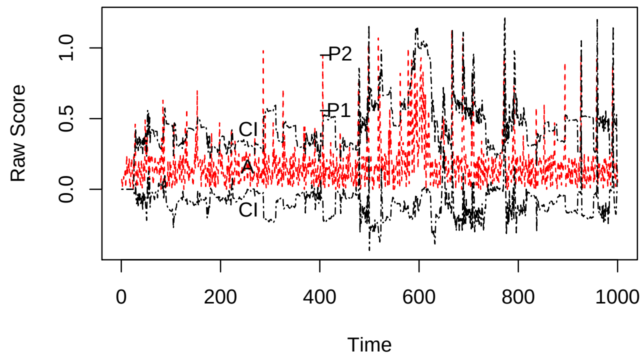

Figure 7: Node scores and 95% CI intervals for the attacker node. Black lines denote CIs while the red line denotes the attacker (A).

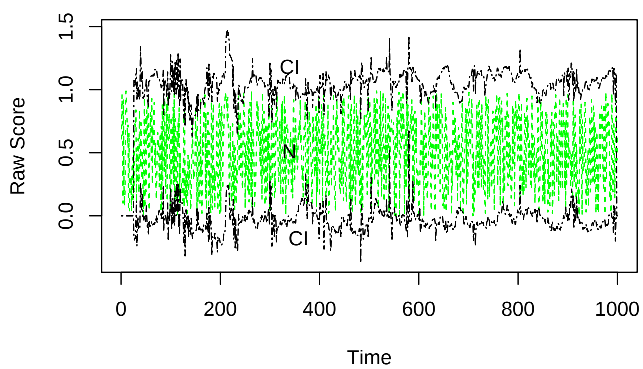

Figure 8: Node scores and 95% CIs for a normal node. Black lines denote CIs while the green line denotes the normal node (N).

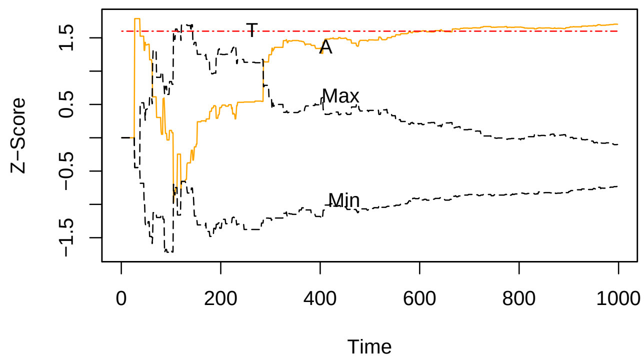

Figure 9: Z-Scores of anomaly scores for Discord analysis.

7.3.1. Traffic volume

A simple measure called detection potential is defined to explain how far an attacker node is deviated from the threshold. It helps to compare between different network conditions. The detection potential d is defined as:

$$d = z - T \tag{10}$$

on the basis of the higher the detection potential the better for the detection.

An attacker was located in a 51 size subnet and generated suspicious events. The same experiment was repeated six times by keeping all parameters unchanged, except attacker's traffic volume. If the attacker's traffic volume is V at the first time, then at each repetition the attacker's traffic volume was incremented by one time as $2V$, $3V$, ...,$7V$. For each experimental run the detection potential (deviation of node scores from the norm) was calculated, and standardised values of the detection potentials are plotted as in Figure 10. As shown in Figure 11, the detection potential is proportional to the traffic volume. The higher the traffic volume produced by an attacker is the better for her detection using the monitoring algorithm.

7.3.2. Subnet size

An attacker was located in a 500 size subnet and the same experiment was repeated six times by keeping all other parameters, except the subnet size, unchanged. Subnet size was changed to 400, 300, 200, 100, 50 and 25 at each experimental run, and Figure 12 and 13 were obtained. As shown in Figure 12, attackers have a less chance to hide behind innocent events when the subnet size decreases. The detection potential is negative exponential to the subnet size, and going beyond 100 size subnet would not make any real sense in terms of detection (see Figure 13). The smaller the subnet size is the better for detection.

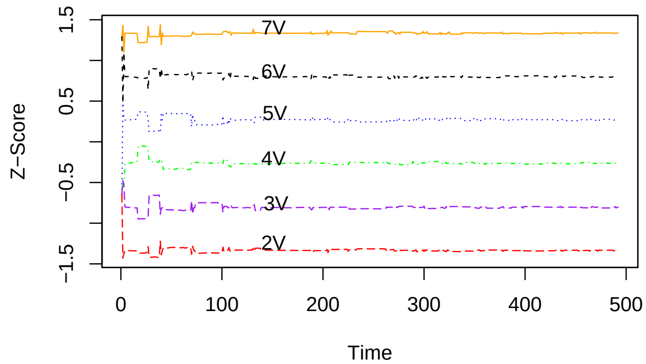

Figure 10: Z-Scores of deviations of cumulative node scores.

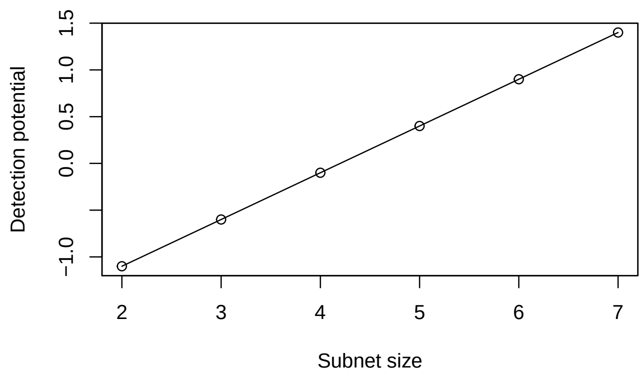

Figure 11: Traffic volume vs the detection potential.

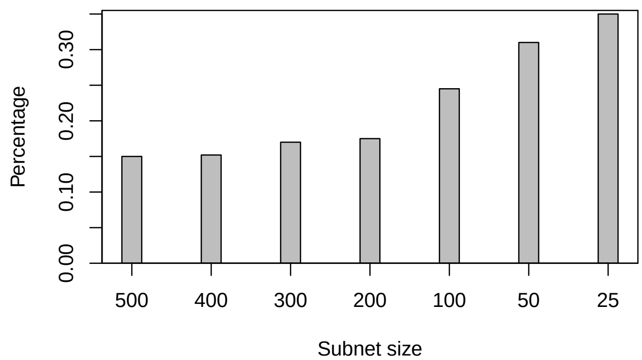

Figure 12: Percentages (%) of suspicious events generated by the attacker.

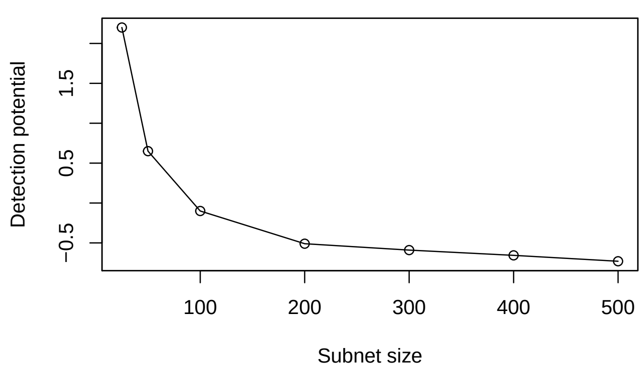

Figure 13: Subnet size vs Detection potential.

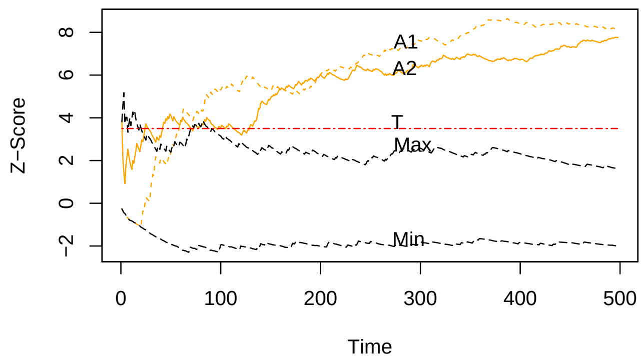

Figure 14: Z-Score graphs for same size subnets with different number of attackers (250 size subnet, two attackers).

7.3.3. Number of attackers

The same experiment was repeated many times by keeping all conditions unchanged, except the number of attackers. The outcomes of only two test cases, two and seven attackers, are presented in Figures 14 and 15. The attacker's node score is dependent on the number of attackers on her own subnet (compare attackers' Z-scores between both graphs).

7.4. Sampling results

A series of experiments have been conducted by changing the sampling rate r, hence n in Equation 5. Figures 16 and 17 present the outcomes of the proposed approach when $r = 20\%$ and $r = 10\%$ of the whole traffic N respectively. *Min* and *Max* represent the minimum and the maximum profile scores of normal nodes in the subnet where attacker node A is located. T represents the Grubbs' critical value (threshold) for attackers' subnet. As it is obvious from Figure 16, proposed algorithm together with chosen sampling technique is capable of detecting stealthy activity using a 20% size traffic sample. It is also possible using even a 10% size sample, but after a considerable time lag.

Figure 18 compares the detection potential against the sampling rate r. It is obvious that a *point of diminishing returns* is existed in Figure 18. When r is larger enough to produce a reasonable level of accuracy, making it further large would be a simply waste of resources of monitoring infrastructure? This answers the question "in anomaly based detection, should all traffic still need to be investigated?"

7.4.1. Network Design

A sampling process has two types of errors: *sampling* and *non-sampling*. Sampling error occurs because of the chance, and it is impossible to avoid but

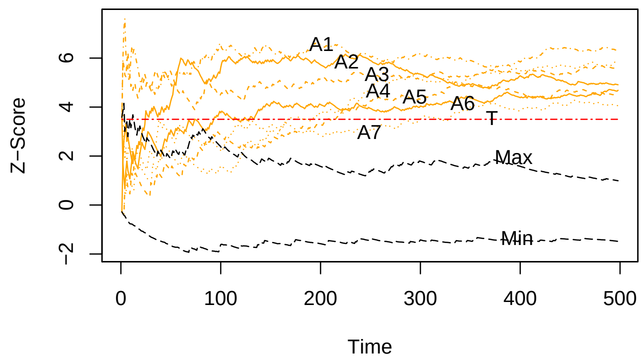

Figure 15: Z-Score graphs for same size subnets with different number of attackers (250 size subnet, seven attackers).

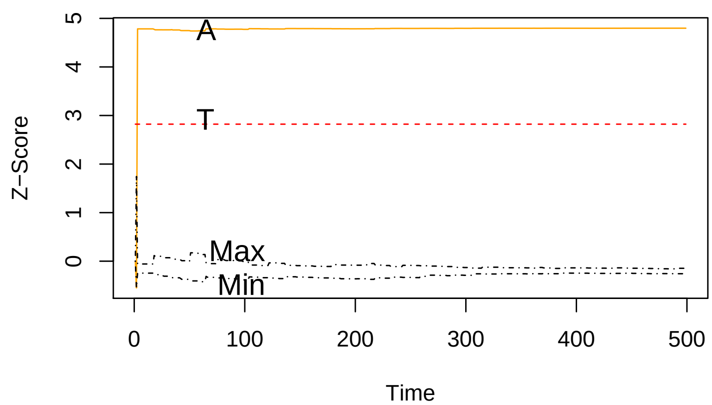

Figure 16: Running the detection algorithm over 20% size sample.

can be minimised by defining unbiased estimators with small variances. Non-sampling errors can be eliminated, and occurred due to many reasons: inability to access information, errors made in data processing, etc [34]. This section examines what impact would varying network size and subnet structure have on *Non-sampling error*. An attacker is located in a 224 size network and $\hat{\pi}$ is estimated in each case as described below. Each simulation was repeated over 100 times. Goodness-of-fit test [35] is applied to statistically test the independence

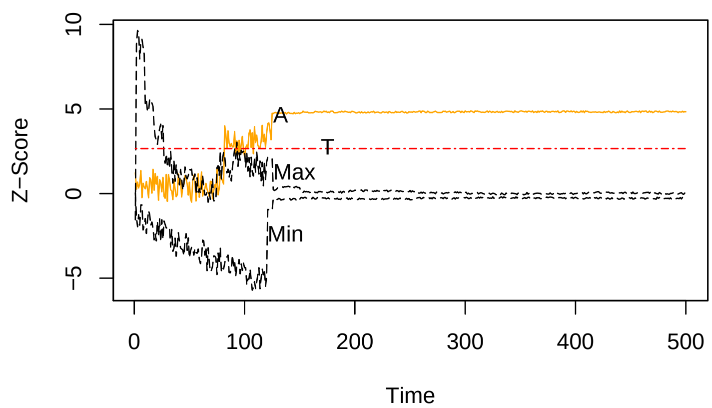

Figure 17: Running the detection algorithm over 10% size sample.

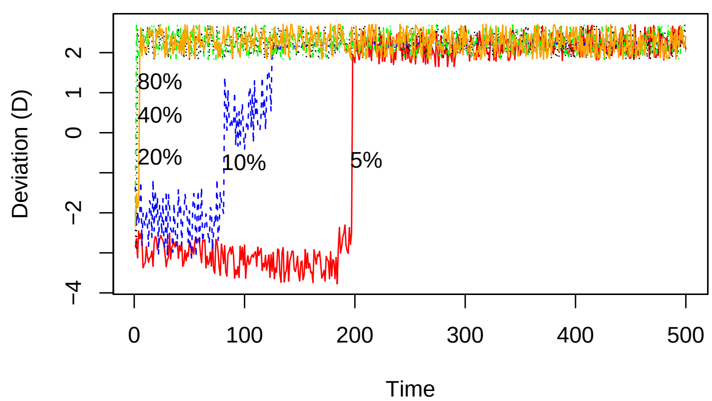

Figure 18: Detection potential vs sampling rate.

(or homogeneity) of proportion π over *sampling rates*, *number of subnets* and *subnet sizes*. If any dependency is found it is depicted in a graph (see Figures 19 and 20).

Proportion of anomaly packets ϕ is considered as the parameter of interest for this analysis and hence sample proportion π is defined as $\pi = (a/n)$; where a is the number of suspicious packets in a given sample size n. Note that

Sampling rate(r)	5%	10%	20%	40%	80%	Whole trace
$\hat{\pi}$	0.00038	0.00034	0.00036	0.00035	0.00036	0.00036
P.Value	0.0970	0.0929	0.0952	0.0971	0.9770	N/A

Table 3: Proportion over sampling rates.

proportion of illegitimate to legitimate traffic, i.e. $a : (n - a)$, is a dominating factor for likelihood of false alarms in an IDS [36]. Though the distribution of ϕ is binomial, in a network scenario, this can be approximated by a normal distribution given a overwhelm number of packets to deal with (it satisfies the conditions of $n.\hat{\pi} \geq 15$ and $n.(1-\hat{\pi}) \geq 15$). Hence, $\phi \sim Normal\left(\hat{\pi}, \sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n}}\right)$, where $\hat{\pi}$ is the observed proportion from samples. This can be used to draw inference about the unknown population proportion ϕ.

Sampling rate (r) Traffic samples at 5%, 10%, 20%, 40%, and 80% rates of the whole trace were drawn and $\hat{\pi}$ was calculated. The null hypothesis H_0 is the assertion that the sample proportion π conforms to the whole traffic proportion ϕ. The alternative hypothesis H_1 is the opposite of H_0.

$$H_0 : \forall r \ \pi_r = \phi \tag{11}$$

$$H_1 : \exists r \ \pi_r \neq \phi \tag{12}$$

$\hat{\pi}$s and p-values of testing H_0 vs H_1 are given in Table 3 where p-values are greater than the significance level $\alpha = 0.01$ for all cases. Therefore there is no enough evidence to reject the null hypothesis H_0. Hence it can be concluded that sample proportion π conforms to the whole traffic proportion ϕ. In other words π can be used to draw inference about ϕ, and chosen sampling technique is capable of producing *representative samples* to the population.

Number of subnets (b) An attacker is located in a 224 size network and same experiment was repeated for four more times by doubling the number of subnets each time (in other words each subnet was divided into two in its immediate repetition) but keeping all other conditions unchanged. The null hypothesis H_0 is the assertion that the proportion π is not affected by the number of subnets b, where b=1, 2, 4, 8, 16. The alternative hypothesis H_1 is the opposite of H_0. If k is a constant:

$$H_0 : \forall b \ \pi_b = k \tag{13}$$

$$H_1 : \exists b \ \pi_b \neq k \tag{14}$$

$\hat{\pi}$s and p-values of testing H_0 vs H_1 are given in Table 4. Since p-values are less than the significance level $\alpha = 0.01$ for some cases it is possible to conclude that there is no enough evidence to accept the null hypothesis H_0, which means that proportion is affected by the number of subnets. Figure 19

Number of Subnets(b)	0	2	4	8	16
$\hat{\pi}$	3.58E-04	2.86E-04	1.12E-04	8.52E-05	1.97E-05
P.Value	N/A	2.65E-01	6.03E-06	3.94E-07	1.04E-11

Table 4: Proportion over Number of Subnets.

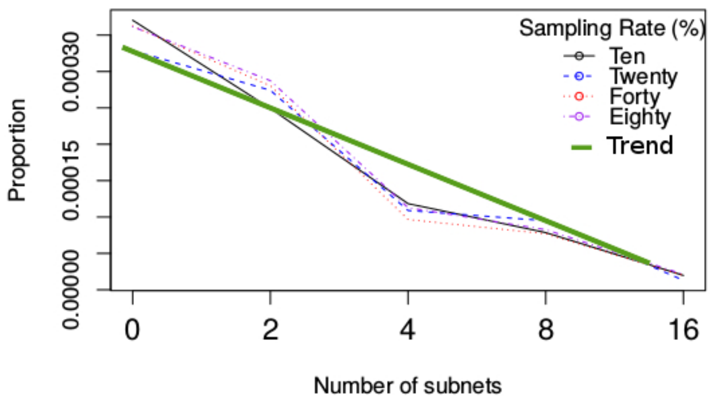

Figure 19: Proportion vs Number of subnets at each sampling rate.

presents the relationship between number of subnets b and proportion π at each sampling rate. When b is increasing $\hat{\pi}$ is decreasing (deviates from the actual value) regardless of sampling rates.

Subnet size (n) An attacker was located in a 5 nodes size subnet in the network, and $\hat{\pi}$ was calculated at each sampling rate. The same experiment was repeated by adding more nodes to produce different subnet sizes: 10, 20, 40, and 80 without changing other parameters. The null hypothesis H_0 is the assertion that the proportion π is not affected by the subnet size n, where n=5, 10, 20, 40, 80. The alternative hypothesis H_1 is the opposite of H_0. If k is a constant:

$$H_0 : \forall n \ \pi_n = k \tag{15}$$

$$H_1 : \exists n \ \pi_n \neq k \tag{16}$$

$\hat{\pi}$s and p-values of testing H_0 vs H_1 are given in Table 5. Since p-values are less than the significance level $\alpha = 0.01$ for some cases there is no enough evidence to accept the null hypothesis H_0, which means that proportion is affected by the subnet size. Figure 20 presents the relationship between subnet size n

Subnet Size(n)	5	10	20	40	80
$\hat{\pi}$	7.28E-04	8.61E-04	8.84E-05	2.06E-04	5.24E-05
P.Value	2.20E-16	2.20E-16	2.80E-01	6.39E-04	N/A

Table 5: Proportion over Subnet sizes.

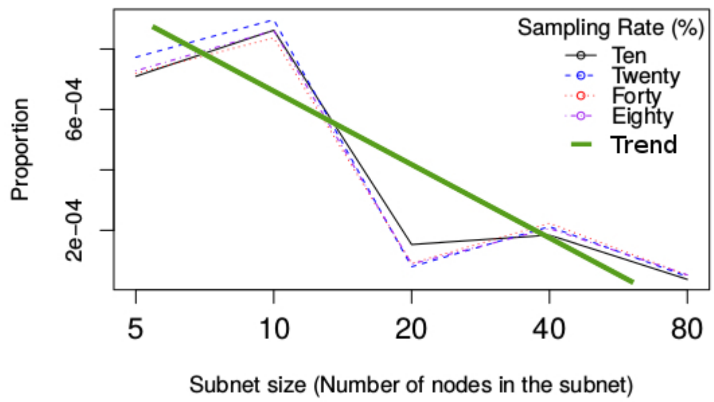

Figure 20: Proportion vs Subnet size at each sampling rate.

and proportion π, where n is increasing $\hat{\pi}$ is decreasing in overall (deviates from the actual value), regardless of sampling.

7.5. Source Anonymity

Using the topology in Figure 21, attack events were generated with anonymous source addresses in order to simulate two cases: single and multiple attackers. In the single attacker case, an attacker is located at a node in subnet *S6* and in multiple attackers case, three attackers are located one in each in three different subnets *S3*, *S5* and *S6*. Figure 22 presents the equivalent tree structure produced by Algorithm 1 for above scenario. The *root* denotes the victim node while g_{i_j} and h_{i_j} denote a gateway or a host node at level i in Figure 22. j is a node number. Dashed rectangles represent a collection of leaves corresponded to hosts in each subnet. Once the tree is obtained, Algorithm 2 is run to locate the attackers as shown in Figure 23 for single attacker, and Figure 24 for multiple attackers.

Figure 23 presents the steps of tracing process from the root of the derived tree. In Step 1, Min and Max represent the minimum and maximum Z-scores of all immediate visible nodes (11 in total, except g_{1_3}) to the root at each time point. Since that graph suggests moving towards g_{1_3}, Step 2 graph is created

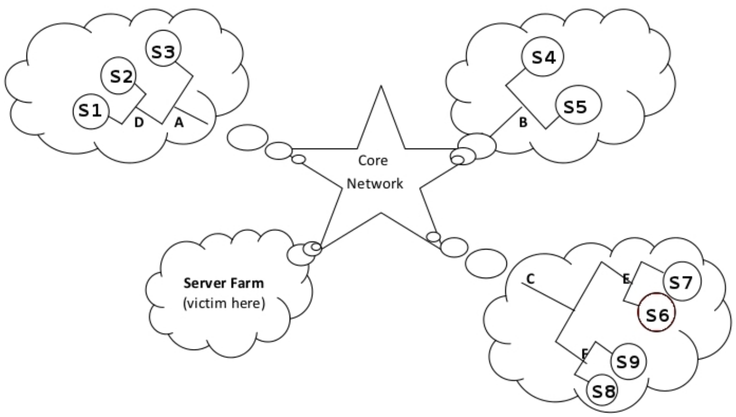

Figure 21: Network topology used for source anonymity experiment.

at node g_{1_3}, and so on. Finally search is narrowing down to the subnet *S6*. Step 4 graph is created at *S6*'s gateway node g_{3_4}, where A denotes the Z-scores corresponded to the true attacker located in that subnet. Min and Max represent the minimum and maximum Z-scores of all other nodes in subnet *S6*. T denotes the threshold which is not defined when number of data points in a set is less than three. In that case the highest scored path is chosen to move towards (see Step 2) in finding attacker or directions to her location.

A similar manner should be followed in interpreting graphs in Figure 24 obtained for multiple attackers. In that case, once an attacker is found tracing algorithm should be back tracked to its immediate parent node and should proceed with next highest Z-scored sub tree to find other suspicious nodes. After Steps 3 and 6, algorithm back tracks to the root node. Table 6 summarises travel sequences for tracing single and multiple attackers.

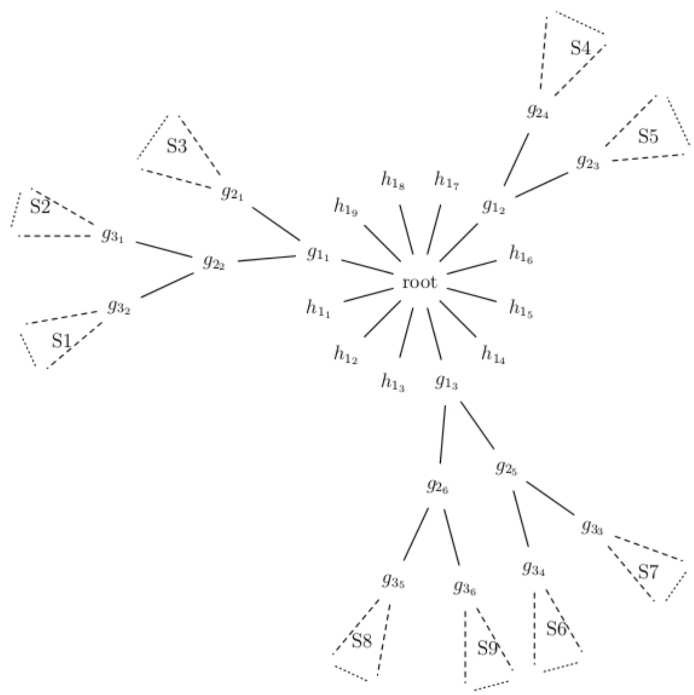

Figure 22: Equivalent tree structure for the given scenario.

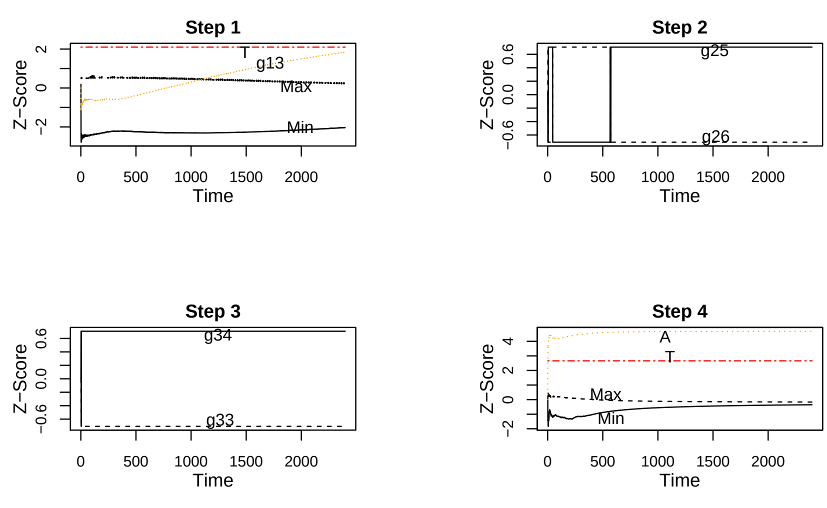

Figure 23: Tracing steps: single attacker case.

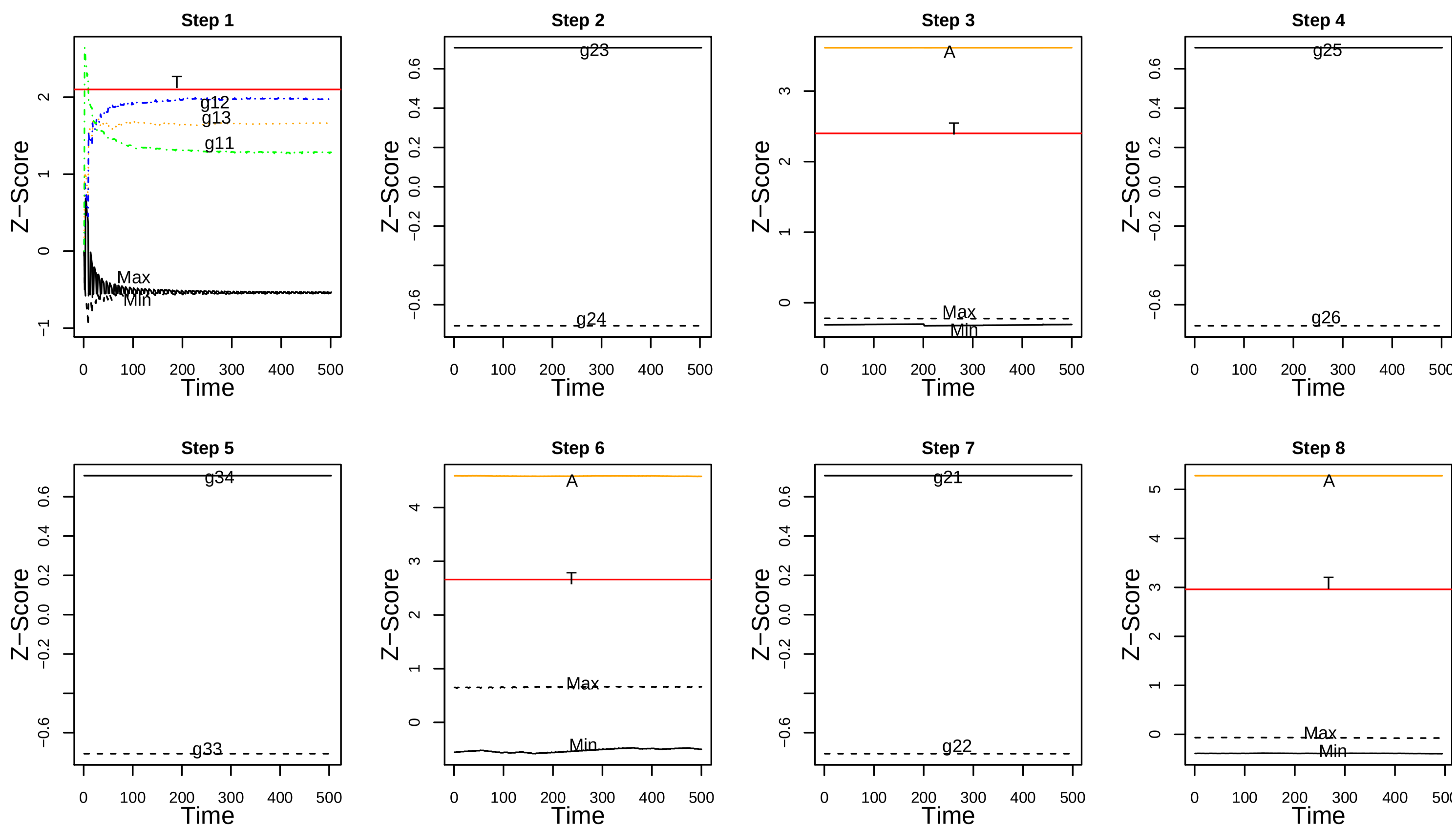

Figure 24: Tracing steps: multiple attackers case.

Scenario	Travel sequence (until all attackers are found)
Single attacker	root, g_{1_3}, g_{2_5}, g_{3_4}
Multiple attackers	root, g_{1_2}, g_{2_3}, root, g_{1_3}, g_{2_5}, g_{3_4},root, g_{1_1}, g_{2_1}

Table 6: Traversal sequences for tracing attackers.

8. Related Work

8.1. Monitoring stealthiness

A scalable solution for insider detection using Bayesian analysis is presented in [13]. Authors maintain incremental profile scores for each node in the system and distinguish suspicious nodes from normal nodes by setting a predefined baseline. If a cumulative score of a particular node is deviated from the predefined control, an anomaly is declared and that node is identified as an insider who warrant further investigation. The major drawback of this approach is setting a predefined control as the baseline. Setting predefined controls is very challenging in network security monitoring. In a network, normal behaviour keeps evolving and a current notion of normal behaviour might not be sufficiently representative in the future. Threshold needs to evolve according to the context and current state of the network. [37] integrates user's technological traits (system call alerts, intrusion detection system alerts, honey pot, systems logs, etc) with data obtained from psychometric tests (predisposition, stress level, etc) for insider detection. User profiles are used to identify the users (human actors) who warrant further investigation. [37, 38]. [39] is similar to [37]. It provides a research framework for testing hypothesises for insider threats by integrating employee data with traditional cyber security audit data. This approach is based on pattern recognition and model-based reasoning. Reasoner is the pattern recognition component which analyses the large amount of noisy data to distinguish variations from norms. Data is processed using a dynamic Bayesian network which calculates belief levels assigned to indicators and assessed the current indicators with the combination of previously assessed indicators to determine the likelihood of behaviours that represent threats. Probabilities are assigned for the Reasoner through expert knowledge. Simulation method is used to evaluate the proposed approach realising the difficulty to find real cases in this domain. When addressing non human threats it finds difficulties due to the psychological profiling components. Hence it is highly organisational dependent, and expertise knowledge is needed to fine-tune the model in order to fit with new environments. However the idea proposed in all above works to incorporate wider range of information into the monitoring process is very interesting. This idea increasingly becomes popular among security community [14].

A co-variance matrix based approach for detecting network anomalies is proposed in [40]. It uses the correlation between groups of network traffic samples. [41] is an approach which uses connection based windows to detect low profile attacks with a confidence measure. Multiple neural network classifiers to detect stealthy probes is used in [42]. Evidence accumulation as a means of detecting stealthy activities is proposed in [43]. A graph-based anomaly detection (GBAD) systems is presented in [44] to discover anomalous instances of structural patterns in data that represent entities, relationships and actions. GBAD is applied to datasets that represent the flow of information between entities, as well as the actions that take place on the information. Authors claim GBAD can apply to tackle several security concerns including identifying violation of system security policies and differentiating suspected nasty behaviour from normal behaviour. Authors acknowledged the need of reducing the time spent for main computational bottleneck. Hence these approaches are not efficient in terms of computational cost (specially for event correlation) for monitoring stealthy activities lasting in several months. Numbers of anomalous instances are far fewer than the number of normal instances is a main constraint for correlation based anomaly detection approaches [6, 45] to succeed in monitoring for stealthy attacks. Accumulating evidence according to a systematic way would help to overcome this issue.

Information visualisation has been proposed in many scholarly works [46, 47, 36, 48, 49] as a method for anomaly detection . Researches in this line often claim "having to go through huge amount of text data (packet traces, log files, etc) to gain insight into networks is a common but a tedious and an untimely task as terabytes of information in each day is usual in a moderate sized network" [48]. Therefore they propose to visualise packet flows in the network assuming that it will help network professionals to have an accurate mental model of what is normal on their own network and hence to recognise abnormal traffic. For example, [46] claims that "the human perceptual and cognitive system comprises an incredibly flexible pattern recognition system which can recognise existing patterns and discover new patterns, and hence recognising novel patterns in their environment which may either represent threats or opportunities". In principle all above works acknowledge that visualisation (by means of graphs or animation) is useful in identifying anomalies patterns. But our position, though visualisation can be motivated on this as visual cognition is highly parallel and pre-attentive than the text or speech, it does little on stealthy activities monitoring. Just presenting raw data in graphical form would not be sufficient. Visualising a traffic flow of a large network for a very long time will end up with a very complicated web of traffic flows. It would be very difficult to compare this with analyst's mental model of the netflow already made in mind. Therefore some kind of data reduction and simplification (information fusion) is needed before visualising security measures. Essentially these approaches are not either systematic or accounted for the "motivation" uncertainty behind an event.

The work presented in [50] is one of the most recent work similar using Bayesian for stealthy activities monitoring, but in a different domain detecting lone wolf terrorists. [21] combines traditional notion of Motive, Means, and Opportunity with behavioural analysis techniques to place each individual on a sliding scale of insider risk. User behaviour is compared with her own baseline and as well as the behaviours of members in their own peer groups using the Euclidean distance. A method for detecting insiders with unusual changes in behaviour by combining anomaly indicators from multiple sources of information is provided in [20]. Authors build a global model and find outliers by comparing each user's activity changes to activity changes of his peer group. [51] defines a Bayesian network model that incorporates psychological variables that indicate degree of interest in a potential malicious insider. A complex Bayesian network for capturing conditional dependencies between different attributes can be found in [52]. Using Bayesian technique and its variants for intrusion detection can be found in [53]. The relevance of information fusion for network security monitoring is widely discussed [6, 54]. A comparison of performance between Bayesian technique, Counting approach, Linear Regression and Artificial Neural Network in insider detection includes [32] which concludes that Bayesian technique is better than the other methods. Also [13] demonstrates that Bayesian approach is superior to the counting algorithm. All above approaches, except [13, 43], require storage of large volumes of event data for analysis. Systems that try to model the behaviour of individuals or protocols are forced to retain large amounts of data which limits their Scalability. Monitoring algorithm proposed in this work is different from [13, 43] by hypothesis, analysis technique and decision criteria.

8.2. Data reduction

With reference to the Sampling, objectives of network monitoring can be classified as Traffic engineering, Accounting and Security specific where accuracy requirements in each objectives are quite different. Using sampling for Traffic engineering and Accounting is widely studied [55], and already been employed by commercially available tools [56]. However those studies are not relevant to this work as our objective is a security specific. A successful sampling technique in Engineering and Accounting would not be essentially an efficient method in Security. Therefore only security related sampling works will be reviewed in this section. [57] samples malicious packets with higher rates to improve the quality of anomaly detection. High malicious sampling rates are achieved by deploying in-line anomaly detection system which encodes a binary score (malicious or benign) to sampled packets. Packets marked as malicious are sampled with a higher probability. Obviously this approach involves additional processing and storage overheads. [58] evaluates quantitatively how sampling decreases the detection of anomalous traffic. Authors use the packet volume as the parameter of interest for this analysis. That work concludes that detecting anomalies with low sampling rates is entirely possible by changing the measurement granularity, and uses relationship between the mean and the variance of aggregated flows to derive optimal granularity. Proposed analysis method in this work was impressed by this idea. [59] investigates the performance of various methods of sampling in network traffic characterisation. They use several statistics that can be used to compare two distributions for similarities, and to compare sample traces with their parent population. [60] evaluates the effect of the traffic mix on anomaly visibility using traces collected at four different border routers and using prior knowledge of two different worm types. Effects of traffic sampling on privacy and utility metrics can be found in [61]. But none of above focuses on stealthy activities. Note that methods proposed for typical rapid attacks cannot be used to monitor for stealthy activities due to several constraints including the limitations of computational resources [12, 13, 62, 63]. To the best of authors knowledge, the work presented in this paper is the first attempt to use sampling technique for stealthy activity monitoring in computer networks.

Based on the sampling frame, existing sampling proposals can be classified into two groups: packet-based and flow-based. Packet-based techniques [57, 58, 59, 60, 64, 65] consider network packets while flow-based techniques [66, 64, 67] consider network flows as elements for sampling. Packet sampling is easy to implement as it does not involve any processing before selection of samples. But in the case of flow sampling, monitored traffic is processed into flows first and then apply sampling technique on whole set of flows for drawing a sample. This requires to use more memory and CPU power of network devices. The most widely deployed sampling method in the literature is packet sampling. It is computationally efficient, requiring minimal state and counters [60]. [68] is a study of combination of packet and flow sampling. A comparison of packet vs flow sampling can be found in [66]. According to [66, 67] flow sampling is more accurate than packet sampling. However it should be noted that this not necessarily means that flow sampling is always better than packet sampling. However, suitability of a sampling method depends on the input parameters to the detection algorithm and monitoring objectives. For example, if inputs to the detection algorithm is flows, obviously flow sampling should be performed well in that scenario than sampling on any other element. [64, 65] are examples to justify that suitability of a sampling frame depends on the detection algorithm. Former investigates how packet sampling impacts on three specific port scan detection methods and the same work has been extended in later to investigate the impact of other methods. Event based and Timer based are the two possible mechanisms to trigger the selection of a sampling unit for inclusion in a sample. Event based approaches collect one elements out of N elements using the chosen sampling method. Naive 1 in N sampling strategy by Cisco NetFlow [56] is a well known example for that method. It samples one packet after every N packets. Event based approaches consume more CPU and memory of network devices as it involves some processing (counting). In a timer based approach, one packet is sampled during N time units. Though this approach is effective in terms of CPU and memory consumption, since it depends on the system timer, choosing larger Ns returns higher sampling errors due to the non-time-homogeneous nature of packets arrivals to the network.

8.3. Tracing

Tracing back is one of the most difficult problems in network security, and a lot of research being conducted in this area [69, 70]. But deterministic packet marking and out of band approaches are not relevant to this work as proposed approach in this work is a probabilistic approach. [71] controls the flooding tests network links between routers to approximate the source. To log packets at key routers and then to use data mining techniques in determining the path which packets traversed through the network is proposed in [72, 73]. The upside of this approach is traceability of an attack long after it has completed. As it is obvious, a downside is that not scalable. [74] propose to mark within the router to reduce the size of packet log and to provide confidentiality using a hash-based logging method. [75] suggest probabilistically marking packets as they traverse through routers. Authors propose router marking the packet with either the routers IP address or the edges of the path that the packet traversed to reach the router. With router based approaches, the router is charged with maintaining information regarding packets that pass through it. However above approaches are focused on DDoS attacks while this paper interests on events related to slow stealthy attacks.

9. Conclusion

Analysts find difficulties to weed through the noise of routine security events and determine which threats warrant further investigations. The profiling technique presented in this paper addresses this issue acting as early warning system. It acknowledges the motivation uncertainty to reduce the possible false alarms which prevent distraction from actual malicious activities. Proposed approach maintains long-term estimates computed on sampled data that individuals or nodes are attackers rather than retaining event data for post-facto analysis. These estimates can be used as triggers of threats which enable authorities to respond to protect systems and deter attackers, for example, by physical, procedural and technical controls such as reduction in permissions and privileges and other incident response activities. Proposed method (section 3) significantly reduces the data amounts to handle and maintain. It maintains only a number of digits equal to the number of nodes in the network to provide a unified view of the state of the network. One advantage of this monitoring strategy is combining multiple indicators not in an ad-hoc but rather in a data-driven manner. Sampling technique utilised in this work draws representative samples. However required level of sampling rate depends on several factors: detection algorithm, parameter of interest, sampling method, level of precision required, duration of monitoring, rate of attack events etc. Further research is needed to identify limitations of sampling in security of cyber physical security systems. With regards to the attribution, finding the correct origin of the activities is very important in cyber systems to locate the right person responsible with a view of persuading them not to do that again. In a situation there are multiple suspected sites to investigate prioritisation centres of attention would be a problematic. Proposed tracing algorithm would help on that, but not solved the attribution problem completely. Investigating more advanced anonymity monitoring technique (e.g. [76]) with the tracing algorithm will be interesting to develop it as more attribution oriented. This is left as future work.

References

1. Vallentin M, Sommer R, Lee J, Leres C, Paxson V, Tierney B. The nids cluster: Scalable, stateful network intrusion detection on commodity hard-

ware. In: *Recent Advances in Intrusion Detection*. Springer; 2007:107–26.

2. Vasiliadis G, Polychronakis M, Ioannidis S. Midea: a multi-parallel intrusion detection architecture. In: *Proceedings of the 18th ACM conference on Computer and communications security*. ACM; 2011:297–308.

3. Shaikh SA, Chivers H, Nobles P, Clark JA, Chen H. Towards scalable intrusion detection. *Network Security* 2009;2009(6):12–6.

4. Shaikh SA, Chivers H, Nobles P, Clark JA, Chen H. Network reconnaissance. *Network Security* 2008;2008(11):12 –6. URL: http://www.sciencedirect.com/science/article/pii/S1353485808701296. doi:http://dx.doi.org/10.1016/S1353-4858(08)70129-6.

5. Shaikh SA, Chivers H, Nobles P, Clark JA, Chen H. Towards scalable intrusion detection. *Network Security* 2009;2009(6):12 –6. URL: http://www.sciencedirect.com/science/article/pii/S1353485809700649. doi:http://dx.doi.org/10.1016/S1353-4858(09)70064-9.

6. Chandola V, Banerjee A, Kumar V. Anomaly detection: A survey. *ACM Comput Surv* 2009;41(3):15:1–15:58. URL: http://doi.acm.org/10.1145/1541880.1541882. doi:10.1145/1541880.1541882.

7. Jiang G, Cybenko G. Temporal and spatial distributed event correlation for network security. In: *American Control Conference, 2004. Proceedings of the 2004*; vol. 2. IEEE; 2004:996–1001.

8. Delooze L, Kalita J. applying soft computing techniques to intrusion detection. In: *Proc. of Cyber Security and Information Infrastructure Research Workshop, Oak Ridge National Laboratory, Oak Ridge, TN*. 2006:.

9. Patcha A, Park JM. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Comput Netw* 2007;51(12):3448–70. URL: http://dx.doi.org/10.1016/j.comnet.2007.02.001. doi:10.1016/j.comnet.2007.02.001.

10. Giacinto G, Roli F. Intrusion detection in computer networks by multiple classifier systems. In: *Proc. of International Conference on Pattern Recognition. Los Alamitos, CA*. 2002:.

11. Smith LI. A tutorial on principal components analysis. *Cornell University, USA* 2002;51:52.

12. Kalutarage HK, Shaikh SA, Zhou Q, James AE. Sensing for suspicion at scale: A bayesian approach for cyber conflict attribution and reasoning. In: *4th International Conference on Cyber Conflict (CYCON) 2012*. NATO CCDCOE; 2012:1–19.

13. Chivers H, Clark JA, Nobles P, Shaikh SA, Chen H. Knowing who to watch: Identifying attackers whose actions are hidden within false alarms and background noise. *Information Systems Frontiers* 2013;15(1):17–34.

14. Davidoff S, Ham J. Network Forensics: Tracking Hackers Through Cyberspace. Prentice Hall; 2012.

15. Drew S. Intrusion Detection FAQ: What is the Role of Security Event Correlation in Intrusion Detection? http://www.sans.org/security-resources/idfaq/role.php; n.d.

16. Advanced methods to detect advanced cyber attacks: Protocol abuse. `http://www.novetta.com/2015/02/advanced-methods-to-detect-advanced-cyber-attacks-protocol-abuse/`; 2014. Accessed: 2015-04-22.

17. Anscombe FJ, Guttman I. Rejection of outliers . *Technometrics 2* 1960;2:123–47.

18. GRUBBS RE. Procedures for Detecting Outlying Observations in Samples. *Technometrics* 1969;11(1):1–21.

19. Hagen N, Kupinski M, Dereniak EL. Gaussian profile estimation in one dimension. *Applied optics* 2007;46(22):5374–83.

20. Eldardiry H, Bart E, Liu J, Hanley J, Price B, Brdiczka O. Multi-domain information fusion for insider threat detection. In: *2013 IEEE Security and Privacy Workshops*. 2013:URL: `https://www.ieee-security.org/TC/SPW2013/papers/data/5017a045.pdf`.

21. Berk VH, Cybenko G, Souza IGd, Murphy JP. Managing malicious insider risk through bandit. In: *System Science (HICSS), 2012 45th Hawaii International Conference on*. IEEE; 2012:2422–30.

22. Yankov D, Keogh E, Rebbapragada U. Disk aware discord discovery: finding unusual time series in terabyte sized datasets. *Knowledge and Information Systems* 2008;17(2):241–62.

23. Chatfield C. The analysis of time series: an introduction. CRC press; 2003.

24. Ansel HC, Prince SJ. Pharmaceutical calculations: the pharmacist's handbook. Lippincott Williams & Wilkins; 2004.

25. Parker T. Finger pointing for fun, profit and war? the importance of a technical attribution capability in an interconnected world. https://media.blackhat.com/bh-dc-11/Parker/BlackHat-DC-2011-Parker-Finger-Pointing-wp.pdf; 2010.

26. Beidleman SW. Defining and deterring cyber war. Tech. Rep.; DTIC Document; 2009.

27. Wheeler DA, Larsen GN. Techniques for cyber attack attribution. Tech. Rep.; DTIC Document; 2003.

28. Charney S. Rethinking the cyber threat: A framework and path forward. http://download.microsoft.com/download/F/1/3/ F139E667-8922-48C0-8F6A-B3632FF86CFA/ rethinking-cyber-threat.pdf; 2009.

29. Saalbach K. Cyberwar methods and practice. *Available FTP: dirk-koentopp com Directory: download File: saalbach-cyberwar-methods-and-practice pdf* 2011;.

30. Riley GF, Henderson TR. The ns-3 network simulator. In: *Modeling and Tools for Network Simulation*. Springer; 2010:15–34.

31. R Development Core Team . R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing; Vienna, Austria; 2010. ISBN 3-900051-07-0.

32. Greitzer FL, Kangas LJ, Noonan CF, Dalton AC, Hohimer R. Identifying at-risk employees: A behavioral model for predicting potential insider threats. Pacific Northwest National Laboratory; 2010.

33. GeNIe . GeNIe-Documentation, Decision Theoritic Modelling: Probability. http://genie.sis.pitt.edu/wiki/Decision-Theoritic-Modelling:-Probability; n.d.

34. Tozal ME, Sarac K. Estimating network layer subnet characteristics via statistical sampling. In: *NETWORKING 2012*. Springer; 2012:274–88.

35. Rao J, Scott A. The analysis of categorical data from complex sample surveys: chi-squared tests for goodness of fit and independence in two-way tables. *Journal of the American Statistical Association* 1981;76(374):221–30.

36. van Riel JP, Irwin B. Identifying and investigating intrusive scanning patterns by visualizing network telescope traffic in a 3-d scatter-plot. In: *ISSA*. 2006:1–12.

37. Kandias M, Mylonas A, Virvilis N, Theoharidou M, Gritzalis D. An insider threat prediction model. In: *Trust, Privacy and Security in Digital Business*. Springer; 2010:26–37.

38. Bradford PG, Brown M, Self B, Perdue J. Towards proactive computer system forensics. In: *International conference on information technology: Coding and computing,IEEE Computer Society*. 2004:.

39. Greitzer F, Paulson P, Kangas L, Edgar T, Zabriskie M, Franklin L, Frincke D. Predictive modelling for insider threat mitigation, pacific northwest national laboratory, richland, wa, tech. rep. pnnl technical report. 2009.

40. Tavallaee M, Lu W, Iqbal SA, Ghorbani AA. A novel covariance matrix based approach for detecting network anomalies. In: *Communication Networks and Services Research Conference, 2008. CNSR 2008. 6th Annual*. IEEE; 2008:75–81.

41. Basu R, Cunningham RK, Webster SE, Lippmann RP. Detecting low-profile probes and novel denial-of-service attacks. Tech. Rep.; IEEE SMC IAS Workshop 2001; West Point, New York, USA; 2001.

42. Streilein WW, Cunningham RK, Webster SE. Improved detection of low profile probe and novel denial of service attacks. In: *Workshop on Statistical and Machine Learning Techniques in Computer Intrusion Detection*. 2002:.

43. Heberlein T. Tactical operations and strategic intelligence: Sensor purpose and placement. *Net Squared Inc, Tech Rep TR-2002-0402* 2002;.

44. Eberle W, Graves J, Holder L. Insider threat detection using a graph-based approach. *Journal of Applied Security Research* 2010;6(1):32–81.

45. Bhuyan MH, Bhattacharyya D, Kalita J. Survey on incremental approaches for network anomaly detection. *arXiv preprint arXiv:12114493* 2012;.

46. Fisk M, Smith S, Weber P, Kothapally S, Caudell T. Immersive network monitoring. In: *proc. PAM2003 Passive and Active Measurement 2003*. 2003:URL: `http://public.lanl.gov/mfisk/papers/pam03.pdf`.

47. van Riel J, Irwin B. Toward visualised network intrusion detection. In: *Proceedings of 9th Annual Southern African Telecommunication Networks and Applications Conference (SATNAC2006). Spier Wine Estate, Western Cape, South Africa*. 2006:3–6.

48. Ball R, Fink G, North C. Home-centric visualization of network traffic for security administration. In: *Proc. of the 2004 ACM Workshop on visualization and Data Mining for Computer Security*. 2004:55–64.

49. van Riel J, Irwin B. Toward visualised network intrusion detection. In: *Proceedings of 9th Annual Southern African Telecommunication Networks and Applications Conference (SATNAC2006). Spier Wine Estate, Western Cape, South Africa*. 2006:3–6.

50. Brynielsson J, Horndahl A, Johansson F, Kaati L, Mårtenson C, Svenson P. Harvesting and analysis of weak signals for detecting lone wolf terrorists. *Security Informatics* 2013;2(1):11.

51. Axelrad ET, Sticha PJ, Brdiczka O, Shen J. A bayesian network model for predicting insider threats. In: *2013 IEEE Security and Privacy Workshops*. 2013:URL: `https://www.ieee-security.org/TC/SPW2013/papers/data/5017a082.pdf`.

52. Das K, Schneider J. Detecting anomalous records in categorical datasets. In: *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM; 2007:220–9.

53. Siaterlis C, Maglaris B. Towards multisensor data fusion for dos detection. In: *Proceedings of the 2004 ACM symposium on Applied computing*. ACM; 2004:439–46.

54. Vokorokos L, Chovanec M, Látka O, Kleinova A. Security of distributed intrusion detection system based on multisensor fusion. In: *Applied Machine Intelligence and Informatics, 2008. SAMI 2008. 6th International Symposium on*. IEEE; 2008:19–24.

55. Duffield N. Sampling for passive internet measurement: A review. *Statistical Science* 2004;19(3):472–98.

56. Cisco . Cisco netflow. `http://www.cisco.com/warp/public/732/Tech/netflow`; 2013.

57. Ali S, Haq I, Rizvi S, Rasheed N, Sarfraz U, Khayam S, Mirza F. On mitigating sampling-induced accuracy loss in traffic anomaly detection systems. In: *SIGCOMM Computer Communication*. 2010:4–16.

58. Ishibashi K, Kawahara R, Tatsuya M, Kondoh T, Asano S. Effect of sampling rate and monitoring granularity on anomaly detectability. In: *In 10th IEEE Global Internet Symposium 2007*. 2007:.

59. Claffy KC, Polyzos GC, Braun HW. Application of sampling methodologies to network traffic characterization. In: *Conference proceedings on Communications architectures, protocols and applications*. SIGCOMM '93; New York, NY, USA: ACM. ISBN 0-89791-619-0; 1993:194–203.

60. Tellenbach B, Brauckhoff D, May M. Impact of traffic mix and packet sampling on anomaly visibility. In: *Proceedings of the 2008 The Third International Conference on Internet Monitoring and Protection*. ICIMP '08; Washington, DC, USA: IEEE Computer Society; 2008:31–6.

61. Fazio P, Tan K, Kotz D. Effects of network trace sampling methods on privacy and utility metrics. In: *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS)*. 2012:1–8.

62. Kalutarage H, Shaikh S, Zhou Q, James A. Tracing sources of anonymous slow suspicious activities. In: Lopez J, Huang X, Sandhu R, eds. *Network and System Security*; vol. 7873 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. ISBN 978-3-642-38630-5; 2013:122–34. URL: `http://dx.doi.org/10.1007/978-3-642-38631-2_10`. doi:`10.1007/978-3-642-38631-2_10`.

63. Kalutarage H, Shaikh S, Zhou Q, James A. Monitoring for slow suspicious activities using a target centric approach. In: Bagchi A, Ray I, eds. *ICISS 2013*; vol. 8303 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg. ISBN 978-3-642-38630-5; 2013:163168.

64. Mai J, Chuah CN, Sridharan A, Ye T, Zang H. Is sampled data sufficient for anomaly detection? In: *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. IMC '06; New York, NY, USA: ACM. ISBN 1-59593-561-4; 2006:165–76.

65. Mai J, Sridharan A, nee Chuah C, Zang H, Ye T. Impact of packet sampling on portscan detection. *NATIONAL UNIVERSITY OF SINGAPORE, SINGAPORE IN* 2006;24:2285–98.

66. Hohn N, Veitch D. Inverting sampled traffic. In: *IEEE/ACM Transactions on Networking*. 2006:68–80.

67. Bartos K, Rehak M. Towards efficient flow sampling technique for anomaly detection. In: *TMA 2012 Conference Proceedings, LNCS 7189*. Springer-Verlag Berlin Heidelberg; 2012:93–106.

68. Yang L, Michailidis G. Sampled based estimation of network traffic flow characteristics. In: *SINFOCOM 2007*. 2007:1775–83.

69. John A, Sivakumar T. Ddos: Survey of traceback methods. *International Journal of Recent Trends in Engineering* 2009;1(2):241–5.

70. Mitropoulos S, Patsos D, Douligeris C. Network forensics: towards a classification of traceback mechanisms. In: *Security and Privacy for Emerging Areas in Communication Networks, 2005. Workshop of the 1st International Conference on*. IEEE; 2005:9–16.

71. Burch H, Cheswick B. Tracing Anonymous Packets to Their Approximate Source. In: *Proc. 2000 of USENIX LISA Conference*. 2000:.

72. Sager G. Security fun with ocxmon and cflowd. *Presentation at the Internet* 1998;2.

73. Stone R, et al. Centertrack: An ip overlay network for tracking dos floods. In: *Proceedings of the 9th USENIX Security Symposium*; vol. 9. 2000:199–212.

74. Alex S, Craig P, Luis S, Christine J, Fabrice T, Beverly S, Stephen K, Timothy S. Single-packet ip traceback. *IEEE/ACM Trans Netw* 2002;.

75. Stefan S, David W, Anna K, Tom A. Network support for ip traceback. *IEEE/ACM TRANSACTIONS ON NETWORKING* 2001;9(3):226–37.

76. Backes M, Kate A, Meiser S, Mohammadi E. (nothing else) MATor(s): Monitoring the anonymity of tor's path selection. In: *Proceedings of the 21th ACM conference on Computer and Communications Security (CCS 2014)*. 2014:.