Accumulating Evidence of Insider Attacks

Howard Chivers[1], Philip Nobles[1], Siraj A. Shaikh[1], John A. Clark[2], and Hao Chen[2]

[1] Department of Information Systems, Cranfield University, Shrivenham, UK
h.chivers/p.nobles/s.shaikh@cranfield.ac.uk
[2] Department of Computer Science, University of York, York, UK
jac/chenhao@cs.york.ac.uk *

Abstract. Insider attacks are often subtle and slow, posing the problem of integrating a large volume of event data from multiple sources over a long period. This paper proposes a scalable solution to combining evidence from multiple sources, by maintaining long-term estimates that nodes are subverted for each node in the system, rather than retaining event data for post-facto analysis. These estimates are then used as triggers for more detailed investigation. We identify essential attributes of event data, allowing the use of a wide range of sensors, and show how to apply Bayesian statistics to maintain incremental node estimates without global updating or normalization. The paper provides a theoretical account of the process, a worked example, and a discussion of its practical implications.

1 Introduction

Insider attacks pose a particular threat because of the knowledge, access, and authority of their perpetrators [12]. Such attacks often involve violations of physical or operational security, or the misuse of authority; they may also involve electronic attacks, in which case the 'electronic insider' is as big a threat as a person. It may be safer for a sophisticated external attacker to subvert an electronic system, often via social engineering, than directly subvert an employee. Such attackers may use technical means to camouflage an attack, such as indirection or address spoofing [1]; however, their most potent weapon in avoiding detection is patience – the world's largest credit card fraud was achieved with a subverted internal system that avoided discovery for over 17 months [9].

Subtle attackers are unlikely to launch large-scale scans, or use known exploits; they will seek to avoid any action that can be immediately identified as an attack. However, they are likely to cause minor security events: an attacker may test known passwords, probe for services, or test new exploits, expecting to hide within the background of user errors, mistakes and other 'noise'. The problem of detecting such an attacker is therefore one of accumulating relatively

* D. Chadwick, I. You and H. Chang (Eds.): Proceedings of the 1st International Workshop on Managing Insider Security Threats (MIST2009), Purdue University, West Lafayette, USA, June 16, 2009. *Copyright is held by the author(s)*

weak evidence over a long period. This issue is one of the 'grand challenges' of the internal attacker problem: "to combine events from one or more sensors, possibly of various types" while "reduce[ing] data without adversely impacting detection" [3]. This paper provides a solution to this critical problem.

The work presented here is couched in terms of networks and systems, and the identification of a subverted node, which is part of a system that is used by a corrupt insider, or is acting as an electronic insider for some other party. However, the approach to characterizing and combining diverse sources of weak evidence is equally applicable to other problems in the insider space, such as identifying criminal or espionage threats from behavioral indicators.

This paper provides a process for combining evidence from various sources based on the application of Bayesian statistics, identifies attributes that must be available to allow the combination of evidence from different types of sensor, and demonstrates the approach with a simulated slow-attack on a network.

The paper is organized as follows: Section 2 is overview of the proposed approach, section 3 describes related work, and the evidential accumulation process is developed in section 4. Section 5 presents an example to show that this process is well behaved in simple cases, while section 6 simulates a challenging insider detection problem, and contrasts the evidence accumulation process with a common, but naive, alternative approach. Section 7 discusses results and open issues, and the paper is concluded in section 8.

2 Overview

Consider how a human investigator might approach the problem of accumulating evidence in the network of Figure 1. The network consists of nodes *(A...J)* with interconnectivity as shown. Two minor security events are detected *E1,* and *E2*; they may originate from an operating system alert, intrusion detection system, or other form of event detection (see section 3).

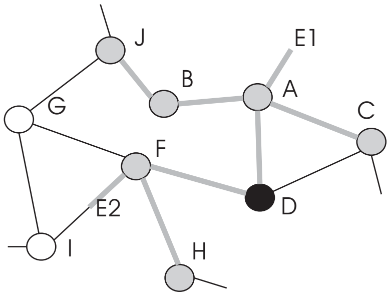

Fig. 1. Intersecting Evidence

Given information about event *E1* and the traffic in the network at the time, the investigator may determine that the nodes most likely to have originated the event are *J, B, A, C or D*. Similarly, when *E2* occurs, at a much later date, the possible originating nodes are *D, F* and *H*. Intersecting these observations suggests node *D* as a common factor, and this may be sufficient to trigger intensive monitoring to determine if it is behaving maliciously.

The data used to identify these security events and their possible sources is necessarily transient; it may not be possible to record sufficient traffic to allow this analysis retrospectively. However, it is initially sufficient to just identify nodes that score differently; in the long, slow, game, it is only necessary to 'tip off' a further investigation by identifying one or more nodes whose behaviour may be unusual. It is not essential to record the events, the traffic from which they were identified, or even the graphs that identify possible sources, provided it is possible to somehow accumulate a 'score' for each node in the system.

This approach solves one of the critical issues in identifying slow attacks: how to maintain long-term state. Systems that try to model the behaviour of individuals, systems or protocols, are forced to retain large amounts of data, which limits their scalability. In the approach described here, the state size is a small multiple of the number of nodes in the network; this state is readily distributed, and its storage is feasible, even for organizations with global networks.

The 'score' that we propose for each node is the probability that the node is subverted, based on the application of Bayesian statistics. This naturally allows incremental updating, and translation of the problem frame from events, which are related to behaviour, to individual attackers. Simpler schemes, such as the event counting used to introduce this section, can be shown to be inferior, as demonstrated by the network simulation in section 6.

In summary, we propose that to identify subtle or inside attackers:

- The primary objective is to identify nodes for further investigation.
- Long-term state is restricted to an incremental estimate of the probability that each node is an attacker.
- Node estimates are updated following every security event, taking account of transient network information that may be available at the time of the event.

This process is complementary to conventional intrusion detection using signatures or heuristics. There is no need to gradually accumulate evidence if the attack is evident; for example, high levels of network activity due to a worm or virus provide compelling evidence of an attack, and in these cases the secondary investigation is concerned with incident management, rather than confirmation.

Section 4 describes how node scores can be estimated and maintained, following a brief summary of related work.

3 Related Work

The use of a tiered approach to insider threat detection, detection followed by a more detailed forensic investigation, is proposed by Bradford et al [4]. Users are profiled according to their function, and deviation from normal behaviour triggers more intensive data collection. Sequential hypothesis testing is proposed to determine whether a process is anomalous and more intensive data collection should be initiated. However, the authors do not show an implementation of their approach, and remark that it could not be carried out for "every user regardless", but itself requires a "triggering process".

The problem is the volume of data that must be maintained, and this is also a issue with datamining approaches, which are often proposed as an adjunct to intrusion detection or audit. Research proposals to alleviate the scalability issue include improving the quality of the raw data, by discovering better behavioral indicators [11] or classifying input features [6], the latter using a Bayesian classifier. An alternative approach by Staniford et al [14] is to selectively retain anomalous network data, with the aim of identifying slow network scans. Anomalous packets are identified based on heuristics developed from real scans. Other approaches include statistical filtering, primarily to reduce false alarm rates and support visualization [7]. In essence, however, all these approaches require the storage of large volumes of event data for later analysis, and the authors themselves often identify scalability as a problem [11].

Aggregation as a means of detecting slow or stealthy attacks has been proposed by Heberlein [10]. His assumption is that slow attacks are still systematic, and the attacker will eventually repeat the attack many times, possibly against different targets. Alerts are classified, accumulated, and displayed on a visualization grid, and any persistent activity which raises alerts of the same type over a long period, can be identified. Although similarly motivated, our work differs by accumulating evidence of attackers, not of incidents, removing the restriction that attackers need to repeat similar attacks. Heberlein's algorithm is also a counting process, which we show to be inferior to statistical reasoning.

Other work directed toward the insider problem is focussed on characterising an attacker's behaviour. The security indicators ('events') used may range from an individual's buying and travel preferences, to electronic alerts. For example, Burford et al [5] propose a comprehensive framework of 'observables' that are used to build a model of individuals' behaviour via graph theory. Eberle et al [8] develop graphs of behavioral events, such as phone calls, to identify subgraphs of normal behaviour, which are used to search for similar but anomalous occurrences. These approaches offer the advantage of modeling the potential attacker, and providing interesting insights into observable behaviour; however, their application may be limited by the computational cost of graph matching over large datasets, as well as by data scalability.

Most of the work described above is still formative; network intrusion detection, however, is established in the literature and supported by both open and propriety products [2]. An intrusion detection system (IDS) uses a behavioral model of a system or protocol and detects anomalous events by either recognizing predefined signatures, or by heuristics. Both approaches have strengths and weaknesses, but despite the usefulness of IDSs in practice, they are hampered by a lack of scalability, and tend to generate large numbers of false positive alerts [2]. From the perspective of this paper, IDSs are an effective way of generating events which may indicate an attack, but are unable to maintain sufficient state to identify slow attacks. An IDS is not the only possible source of security events; for example, the behavioral events referenced above, operating system audit trails, and even Honeypots [13], which are security traps with no operational functionality, are all possible sources of security events.

In summary, the challenge of integrating information from many sources in order to identify patient internal attackers is still an important open question [3].

3.1 Updating Evidence

This section develops the detailed theory necessary to achieve the method outlined in section 3: to collapse the problem of attacker identification to updating a single score for each network node, or user. The section first outlines the evidential scenario, and the attributes required to characterize security events. Standard Bayesian updating is summarized, followed by the development of the process for updating evidence of insider attacks. Finally, the practical issue of relating this process to real security events is discussed.

Definitions

Node: This paper uses network terminology, without loss of generality to broader types of human or attack behavior. A node is a network component, such as a user's end system, a router, or a user.

Event: An event is an alert that indicates a possible security violation; it may be an anomalous phone call, a failed connection, or something more certain, such as a known electronic exploit.

The evidential scenario is presented in Figure 2. Node (a) is a network node, and (x) is an event which is detected somewhere in the network; there is some evidence that identifies the nodes that may have originated the event.

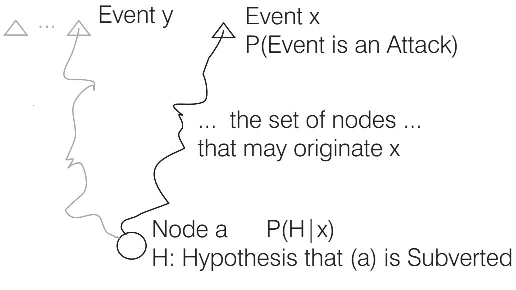

Fig. 2. Evidential Scenario

Event (x) may indicate an attack. Some security events are almost certainly attacks; however, there are many more that may be user mistakes, backscatter, or other forms of network 'noise'. For example, an attempt to connect to a non-existent webserver is often a simple mistake, but could also be an attack probe.

In addition to uncertainty about the extent that an event is an attack, there may also be uncertainty about the source of the event. For example, the attacker may be able to spoof its network address, or the event may only be traceable to a subnetwork. In order to accumulate evidence from a wide range of different sources, they must be characterized by uniform parameters that describe these various attributes. We propose that the difference between different security events can be characterized by three parameters:

- *P(Attack):* the probability that an particular event (x) is actually caused by an intentional attack.
- *The Causal Node Set:* the set of network nodes that could have caused the event, even if it was a not an attack.
- *P(Causal):* the probability that the causal node is actually within the node set.

Given a sequence of events characterized by these parameters, we wish to investigate the hypothesis that a particular node is subverted, or acting as the agent of an attacker. We will first summarize the standard approach to Bayesian updating, then show how it can be applied in this case.

4 Bayesian Updating

Bayesian updating provides an estimate of the probability that hypothesis H is true, given an event, (x).

$$P(H|x) = \frac{P(x|H) \cdot P(H)}{P(x)} \tag{1}$$

This theorem uses *P(x|H)*, the probability of event *(x)* given that the hypothesis is true, to update the initial ('prior') estimate of the probability that the hypothesis is true, *P(H)*. Simple updating of this type is often used in medical diagnosis; given knowledge of the probability of a symptom (the Event) given a disease (the Hypothesis), it provides a principled estimate of the likelihood of the disease given the symptom. It is essentially this change of reference frame – from symptom to cause – that is needed to identify internal attackers from their behaviour.

The denominator, *P(x)*, the probability of the event, is effectively a normalising factor. In many cases, including ours, it is difficult to estimate; however, by making use of $P(H|x) + P(\neg H|x) = 1$, it is straightforward to eliminate *P(x)* and derive the following standard variant of Bayes' theorem:

$$P(H|x) = \frac{P(x|H) \cdot P(H)}{P(x|H) \cdot P(H) + P(x|\neg H) \cdot P(\neg H)} \tag{2}$$

Practical applications of Bayes theorem often combine several sources of evidence. Achieving a usable update formula for multiple evidential events requires an assumption of conditional independence – that individual security events do not cause each other. The derivation is given in standard texts on Bayes. For example, the formulae, which gives the revised probability of the Hypothesis, H, given two items of evidence, *(x)* and *(y) is*:

$$P(H|x,y) = \frac{P(x|H) \cdot P(y|H) \cdot P(H)}{P(x|H) \cdot P(y|H) \cdot P(H) + P(x|\neg H) \cdot P(y|\neg H) \cdot P(\neg H)} \quad (3)$$

This pattern can be extended to cope with multiple items of evidence.

4.1 Combining Evidence from Security Events

The evidential scenario is described at the start of this section; in detail, we define:

S The set of all nodes in the system.

#S The total number of nodes in the system.

a,b... Particular network nodes. $a, b, ... \in S$

$\mathbf{H}_a$ The Hypothesis that we wish to update: that a particular node (a) is subverted, or being used to mount an attack within the system.

E The set of all Security Events generated over time.

x,y... Particular events that may provide evidence of an attack. $x, y, ... \in E$

$\mathbf{P}_x$**(Attack)** The probability that a particular event (x) actually originates from an intentioned attack.

$\mathbf{C}_x$ The Causal set of nodes associated with event (x); in other words, the set of nodes that may have originated the event.

#$\mathbf{C}_x$ The number of nodes in set C_x

P($\mathbf{C}_x$**)** The probability that C_x includes the node that originated the event. In other words the accuracy with which C_x is estimated.

The parameters P_x*(Attack)*, C_x*,* and *P(*C_x*)* were introduced in the introduction to this section as the attributes needed to characterize an event.

We wish to update an estimate of the probability of H_a, following event (x). In equation 2, above, the prior probabilities $P(H), P(\neg H)$ will depend upon the node (e.g. the prior probability of subversion of a server may be significantly different to that of a user client), but will otherwise be constant, so we can write $P(\neg H_a) = 1 - P(H_a)$

However, to obtain an estimate of $P(x|\neg H_a)$it is necessary to take into account that it may not be possible to attribute an event to a single node (a), but only identify a set of nodes, C_x, from which the event may have originated. Unlike the prior probability, this is dependent on the event, as well as on the node, since different events will be generated by different sensors with different capabilities and views of the network.

There are three types of node to consider: the node currently being updated, (a), other nodes in the set C_x, and other nodes in the system that are not in C_x. As a consequence, for an event (x), there are two alternative hypotheses to H_a:

$\mathbf{R}_{a,x}$ That node (a) is not an attacker, but is within C_x.
$\mathbf{I}_{a,x}$ That node (a) is not an attacker, and is outside C_x.

Since $R_{a,x}$ and $I_{a,x}$ are disjoint, we can write:

$$P(x|\neg H_a) = P(x|R_{a,x}) \cdot P(R_{a,x}|\neg H_a) + P(x|I_{a,x}) \cdot P(I_{a,x}|\neg H_a) \tag{4}$$

If a node is not an attacker, we can expect the probabilities of the two alternative hypotheses to be a simple function of the numbers in each set. Substituting $P(R_{a,x}|\neg H_a) = \frac{\#C_x}{\#S}$ and $P(I_{a,x}|\neg H_a) = \frac{\#S-\#C_x}{\#S}$into equation (4), we obtain:

$$P(x|\neg H_a) = P(x|R_{a,x}) \cdot \frac{\#C_x}{\#S} + P(x|I_{a,x}) \cdot \frac{\#S - \#C_x}{\#S} \tag{5}$$

Substituting (5), and the expression for $P(\neg H_a)$given above into the normalized version of Bayes theorem given in equation (2), we obtain:

$$P(H_a|x) = \frac{P(x|H_a) \cdot P(H_a)}{P(x|H_a) \cdot P(H_a) + [P(x|R_{a,x}) \cdot \frac{\#C_x}{\#S} + P(x|I_{a,x}) \cdot \frac{\#S-\#C_x}{\#S}] \cdot [1 - P(H_a)]} \tag{6}$$

Defining:

$$\Delta_{a,x} = \frac{P(x|H_a)}{P(x|R_{a,x}) \cdot \frac{\#C_x}{\#S} + P(x|I_{a,x}) \cdot \frac{\#S-\#C_x}{\#S}} \tag{7}$$

Rearranging equation (6) and substituting in $\Delta_{a,x}$ gives:

$$P(H_a|x) = \frac{\Delta_{a,x} \cdot P(H_a)}{\Delta_{a,x} \cdot P(H_a) + [1 - P(H_a)]} \tag{8}$$

This update formulae can be extended to multiple events under the assumption of conditional independence, similar to equation (3); the resulting update formulae becomes:

$$P(H_a|x, ...y) = \frac{\Delta_{a,x} \cdot ... \cdot \Delta_{a,y} \cdot P(H_a)}{\Delta_{a,x} \cdot ... \cdot \Delta_{a,y} \cdot P(H_a) + [1 - P(H_a)]} \tag{9}$$

This is the final update formulae. Bayes updating is often presented in this form; the application specific elements are contained in the definition of $\Delta a,x$, which will now be further developed by substituting in the information that characterizes security events. We will consider each element of equation 7 in turn.

P(x|H$_a$) is the probability that an event (x) occurs given that the identified node is actually subverted. Since we will update in response to events, then for each event $P_x(Attack)$, provides a plausible value, which only discounts the possibility that false alarms may have originated from a genuine attacker.

P(x|R$_{a,x}$) is the probability that the node is not subverted, but that it lies within the set of nodes that may have originated the attack. In terms of our model parameters, this corresponds to $[1 - P_x(Attack)] \cdot P(C_x) \cdot \frac{\#C_x}{\#S}$

P(x|I$_{a,x}$) is the probability that the node is not subverted, but that it lies outside the set of nodes that may have originated the attack. Of course, we would wish to correctly identify the range of nodes that may originate the attack, in which case this probability will be zero; however, in the real world we must take into account the possibility that C_x cannot be enumerated with certainty, giving a value of: $[1 - P_x(Attack)] \cdot [1 - P(C_x)] \cdot \frac{\#S - \#C_x}{\#S}$

Substituting these parameters into $\Delta_{a,x}$, we obtain:

$$\Delta_{a,x} = \frac{P_x(Attack)}{[1 - P_x(Attack)] \cdot \left[P(C_x) \cdot \left(\frac{\#C_x}{\#S}\right)^2 + [1 - P(C_x)] \cdot \left(\frac{\#S - \#C_x}{\#S}\right)^2\right]} \tag{10}$$

This ratio is used to update the probability that a node is an attacker.

4.2 Updating in Practice

Equations (9) and (10) provide the necessary theory to achieve the objective of discarding the details of security events, while retaining a simple score for each node which summarises the evidence that the node is an attacker. A naive algorithm to achieve this would be:

1. Initialize each node with its prior probability, $P(H_a)$. Values may be suggested by survey data; in large systems a prior assumption that a small number of nodes (e.g. 10 of 10,000) are likely to be subverted is reasonable. This parameter is of most value if different nodes have significantly different prior probabilities, for example the difference between a router and a laptop.
2. For each security event:
   (a) Establish the distinguishing parameters (the probability that it is an attack, the nodes that may have originated the attack, and the probability that the node set contains the attacker).
   (b) Calculate Δ from equation (10).
   (c) Multiply the node score by Δ for each node in the set, but not for any others in the system.
3. When required, substitute the node score (the product of all Δs) into equation (9) to obtain the probability that the node is an attacker.

A feature of accumulating evidence in this way is that, assuming the evidence collected is generally useful (a true positive is more likely than a false positive), then over a long period the probabilities converge towards unity. However, we are only concerned with comparative scores, in order to identify nodes that are distinctive and require further investigation. In practice, then, it is sufficient to use Logarithmic scores, simply adding Log(Δ) to each node indicated by an event. Equation (9) can still be reconstructed from this information, but more usually, the highest node score is chosen for further investigation.

The reader may be wondering about the value of calculating Δ at all at this stage, since we simply add its logarithm to the score for indicated nodes. However, this differs significantly from a simple counting algorithm, where the score for each node is incremented when it is identified as the possible source of a security event. The update value, Δ, characterizes exactly how much evidence is provided by each event. This important distinction is illustrated in the worked example presented in section 6.

5 Simple Example

Before showing a simulation of a realistically difficult example (see section 6), this section explores if the evidential accumulation process has intuitively appealing behaviour; in particular, given a single sub-network, in which the sender can be readily identified:

- Does the evidential process identify an attacker sending at a slightly higher rate than the background of errors from normal nodes?
- If the rate of attack increases, is the process stable, and does it enable the attackers to be identified earlier?
- Does the process accommodate multiple attackers with different rates of attack (i.e. can one node hide behind another's attack)?

We assume a single sub-net of 50 nodes, in which the originating node of an event can be identified (i.e. C_x=1); we assign $P(Attack)$ an arbitrary probability of 0.083. Time is divided into slots (e.g. single minutes) and the average background rate of random innocent events that may be misinterpreted as attacks is 1/50 per node – in other words, one event per minute. Three nodes within the sub-net are designated attackers, and they generate random attacks at rates of 2, 4 and 8 times the total background rate.

The scores resulting from simulating this scenario are shown in Fig. 3. All three attack nodes are well distinguished from the background level of events, which is indicated by the 'other nodes' result, which is the score for a typical innocent node. As would be expected, if the attack rate is higher, the discrimination improves. The accumulation of evidence is well behaved, and the higher rate nodes do not interfere with the accumulation of evidence relating to attackers operating at a lower rate.

6 Insider Attack Simulation

This section presents an example of evidence updating in practice. The example shows that complex network propositions can be accommodated straightforwardly, and contrasts the principled accumulation of evidence with a simple counting scheme. This example includes features that are common in this problem space, including:

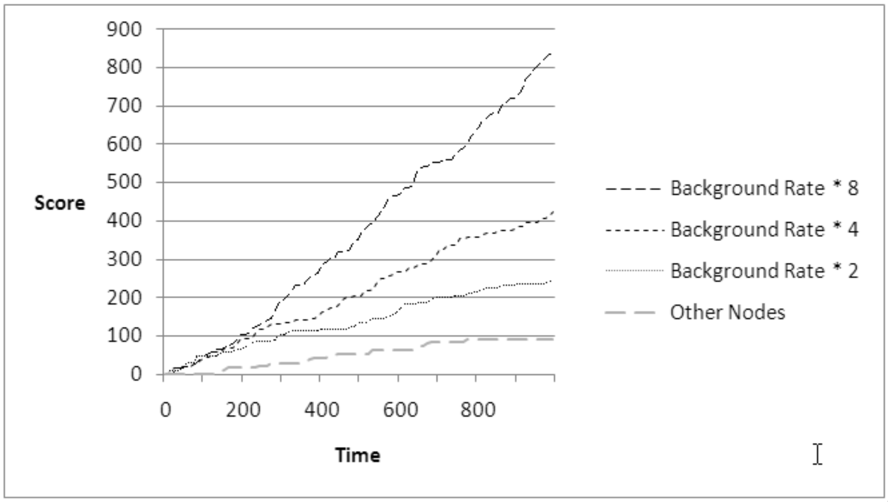

Fig. 3. Simple attacker scenario in a single sub-network

- Sensors with different capabilities; for example, certainty of detection and ability to identify source nodes.
- Attackers whose rate of attack is below the background rate of false positive alerts for the system.
- Attacks that employ address spoofing.

An important practical issue is the estimation of the three parameters that characterize a security event; relating these to actual systems and assessing the need for accuracy is subject to ongoing study. To date it has been possible to achieve realistic results by assigning *P(Attack)* as a fixed value for a given sensor within a deployment context, and by creating a simple rule-set that maps the network connection associated with an event to a set of nodes, giving C_x and $P(C_x)$, depending on the configuration and protocol.

The network used in this example is given in Fig. 4. This network has 200 nodes, most of which are user systems located in four separate client sub-networks. Two of these sub-networks have nodes that been subverted and are attacking the system. The purpose of dividing the clients into several sub-nets (apart from the fact that this is a common configuration) is to contrast the detectability of attackers in different sized sub-networks, given that we assume that in many cases it will be possible to identify only the sub-net from which an attack originated. This arrangement allows us to investigate the scores accrued for an attack node (3 or 53) versus other nodes in the same sub-net, and nodes in a control sub-net of the same size and performance with no attacker.

Most of the traffic in the system is between the clients and servers, via the core network. Router and firewall detail is not shown, and because the object is to investigate evidence accumulation rather than event generation we model a two unspecified types of security event: those that can be detected within client

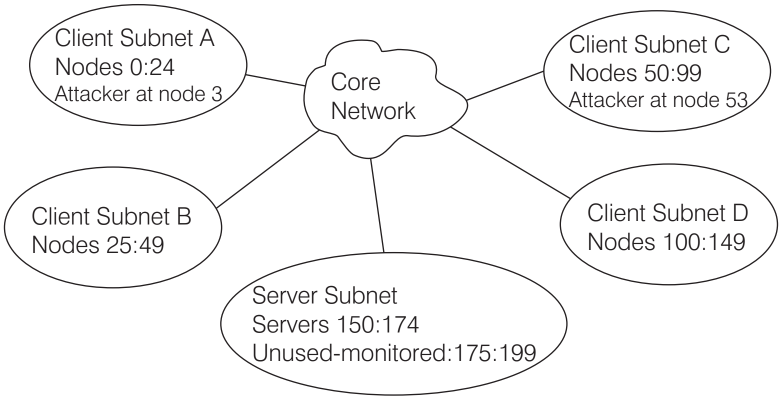

Fig. 4. Test Network

sub-networks, and events in the server farm. For example, an event could be an attempt to connect to an exploitable network port.

Attackers are expected to generate security events at a rate that is much lower than the background rate of 'mistakes' by normal clients, in order to remain undetected. In the simulation below, time is measured in arbitrary clocks (e.g. minutes), and the probability of a normal client generating a security alert in any time slot is 1/150; in other words the system suffers an average of one false alarm every minute. In contrast, attackers generate events at a rate of 1/25; one event every 25 minutes.

In addition to the low attack rate, to further avoid detection, attackers use address spoofing. Events detected outside the sub-net containing the attacker can only be assigned to the whole sub-net. Only events identified within the sub-net containing the attacker (i.e. directed toward nodes within that sub-net) can be traced to a specific node.

An outline calculation illustrates the difficulty of this problem. Consider the attacker in sub-net A. Viewed from outside, the sub-net can be expected to generate innocent background events (false alarms) at a rate of one every 6 minutes (*P()=25 * 1/150*). The events generated by the attacker are distributed at random across the network, so of these, 25/200 are towards the attacker's own sub-network, and 175/200 are visible externally. This results in an externally visible attack every 29 minutes (*P()=1/25 * 175/200*), and these events can only be identified with the whole sub-net. Events targeted at random to nodes within the sub-net can be identified to a particular attacker, but these occur at a rate of only one every 200 minutes (*P()=1/25 * 25/200*). Of course, given this information the reader could devise a solution to identify the attacker, but the problem addressed here is how to use all the available information when the location of the attacker and the traffic patterns are unknown in advance.

In summary, the event parameters used in the simulation are:

C_x contains all the nodes in the source sub-net, unless the destination of the network message that caused the event is in the same subnet as the source, in which case C_x contains just the source address.

$P(C_x)$ is set to unity, since C_x includes all the possible source nodes.

$P(Attack)$ is set to 0.33 for all locations except the server nodes, for which a value of 0.083 is assigned. (These are arbitrary, for the sake of demonstration. It seems plausible that an incident at a location to which most of the traffic is directed is less likely to be an attack, but in practice that is dependent on the actual event. The only special feature in the choice of value is avoiding fractions such as 25/150 that match the system topology and may produce anomalous results in a small system. Varying these parameters result in different scores, but not at the expense of overall discrimination.)

A network simulator was used to generate random traffic as outlined above, and the scores for the resulting security events were accumulated as described in section 4. The results are shown in Fig. 5.

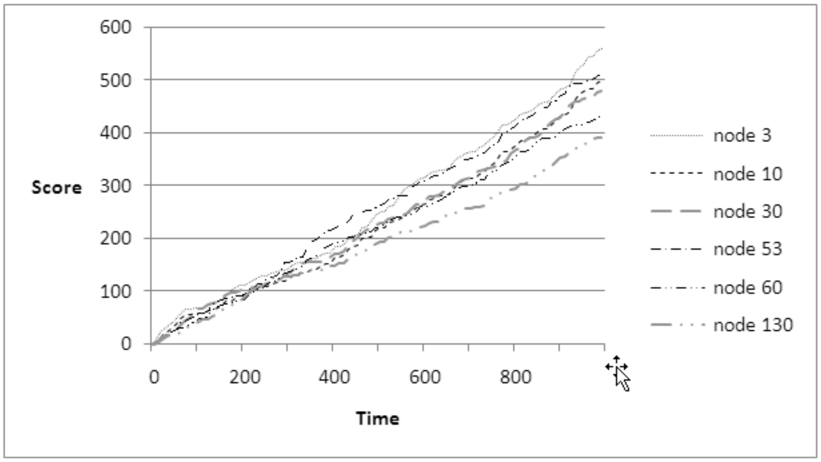

Fig. 5. Network Simulation Results

Fig. 5. shows node scores as they are accumulated. The nodes shown are attackers (3,53), representative nodes in the same sub-nets (10,60), and representative nodes in the same sized sub-nets with no attackers (30,130). Nodes (3,10,30) are from 25-node sub-nets, and nodes (53,60,130) are from 50-node sub-nets, which contain a significant proportion of the nodes in the network.

The results show that insider attacks can be clearly distinguished from background noise in the system. A longer running simulation, given in Fig. 6., provides a view of the asymptotic performance of the process.

For each size of sub-net the proposed scoring clearly distinguishes the attacker as an individual, and the sub-net containing the attacker, from the control sub-

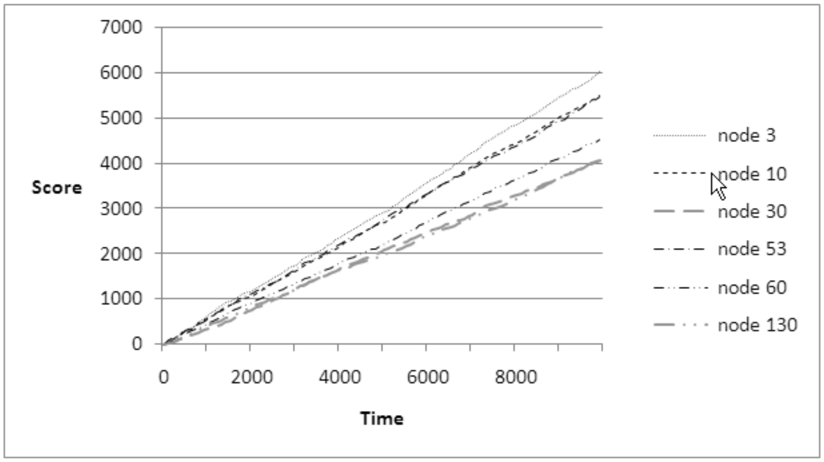

Fig. 6. Long Term Performance

net with no attacker. The distinction between different sized sub-nets is not fully maintained: both attackers are well distinguished from both control networks, however, the smaller sub-net containing an attacker is not well distinguished from the individual attacker in the larger sub-net. In practice, grouping the small sub-net with the two attackers does not present a problem, since it still provides a correct diagnosis of the attacks, that can be subject to further investigation. We conjecture that the issue here is not that the scoring method is inherently biased between different sizes of C_x (that is, any more than the actual evidential content varies with C_x), but that the larger sub-networks in this system are a substantial fraction of the system size.

The effectiveness of this Bayesian approach can be judged by comparison to the counting algorithm used to introduce section 2, and adopted by some researchers. Assuming that the same deductions can be made from the security events (specifically that C_x is the same for each event), the result of using a counting approach, where node scores are simply incremented if they are identified, is given in Fig. 7.

On a realistic problem, the counting approach fails in almost every respect. Attackers are not distinguished from other nodes in their sub-net, and there is little difference between a sub-net containing an attacker, and a control sub-net with no attacker. Instead, the primary distinction is between nodes on the basis of network size; essentially the larger sub-nets generate more background traffic, so receive a proportionately higher score.

7 Discussion

The proposed updating process is effective because it relates event evidence to the hypothesis that the node (or user) is an attacker. This change of reference

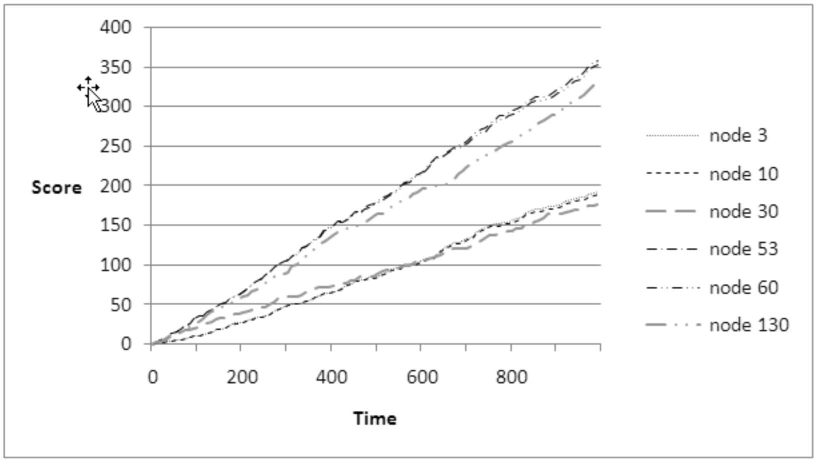

Fig. 7. Counting Algorithm Performance

frame allows event data to be discarded, while retaining the weight of evidence for attackers. The process scales linearly with the number of nodes in the system, and is likely to be applicable to a very wide range of systems and circumstances.

The updating ratio, Δ, can be thought of as the ratio of true positives to false positives. However, Bayes has been used, rather than the simple probability ratios that would be suggested if information theory was employed, in order to effect the change of viewpoint from the event to the attacker. Δ takes account of ancillary information such as the number of nodes that are indicated by the event, and the degree of certainty in their estimation.

Δ can be used as a figure of merit for sources of information; essentially, if Δ is consistently fractional for a sensor, then the resulting events will degrade the quantity of available information, rather than improve it.

The attributes described in section 4 (probability of attack, possible sources, and likelihood that the attacker is in this set) are not specific to any particular type of event generator, and can be applied at different levels of abstraction, if necessary within the same system.

There are a number of practical considerations that are subject to ongoing study. The first implementation decision is which real components are regarded as 'nodes': should nodes model all network components, just routing components and endpoints, or just endpoints such as clients, servers or users? To date, only endpoint nodes have been considered; this decision is based on the prior probability of network components originating attacks, and the convenience in associating events with their possible sources.

A key practical issue is how to determine which nodes are a potential source of any particular event, and to what degree. Ideally this assessment would be evidence-based using recent network history, but although this is feasible in principle, it is an open question if this can be achieved in practice. However, even simple strategies, such as the one used in section 6, provide demonstrable benefit.

This research is ongoing, and other open issues include the sensitivity of the assignment of $P(Attack)$ for disparate sensors, and the possibility of decision criteria other than the maximum score function used above.

8 Conclusion

This paper provides a solution to a critical problem in insider attacker discovery: how to combine events from multiple sensors, and manage the data explosion that is otherwise needed to support the identification of long-running attacks.

The key concept is to move away from maintaining models and evidence of behaviour, and instead maintain an incremental assessment for every user/node in the system that the node is an attacker. This approach is extremely scalable; the updating algorithm is soundly based in Bayesian statistics, and avoids the need for global updating or normalization. The approach is well behaved, in the sense that higher volumes of attack make detection easier, and in a worked example which includes several of the difficulties faced in practice, it significantly outperforms counting algorithms (see section 6).

In addition, this work identifies the attributes or parameters that need to be standardized for disparate sources of security event to be combined, allowing the use of a wide range of different sensors, at different levels of abstraction. The key criteria for a sensor (see section 7) is that it tends to provide information rather than add confusion, and a side effect of the updating process presented here is a criteria for deciding when this is the case.

Research on this approach is ongoing, both using simulation and relating the work to real sensors; some of the open questions are described in section 7.

References

1. CERT incident note IN-98-05: Probes with spoofed IP addresses, 24 November 1998.
2. Rebecca Bace and Peter Mell. Intrusion detection systems (IDS). Technical Report SP 800-31, National Institute of Standards and Technology (NIST), 2001 2001.
3. Richard C. Brackney and Robert H. Anderson. Understanding the insider threat. Technical Report Proceedings of March 2004 Workshop, RAND National Security Research Division, 2004.
4. Phillip G. Bradford, Marcus Brown, Josh Perdue, and Bonnie Self. Towards proactive computer-system forensics. In *International Conference on Information Technology: Coding and Computing (ITCC 2004)*, pages 648 – 652. IEEE Computer Society, 2004.
5. John F. Buford, Lundy Lewis, and Gabriel Jakobson. Insider threat detection using situation-aware MAS. In *11th International Conference on Information Fusion*, pages 1–8, Cologne, Germany, 2008. IEEE Xplore.
6. Srilatha Chebrolua, Ajith Abrahama, and Johnson P. Thomas. Feature deduction and ensemble design of intrusion detection systems. *Computers and Security*, 24(4):295–307, 2004.

7. Jeffrey B. Colombe and Gregory Stephens. Statistical profiling and visualization for detection of malicious insider attacks on computer networks. In *The 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, pages 138–142. ACM Press, 2004.
8. William Eberle and Lawrence Holder. Insider threat detection using graph-based approaches. In *Cybersecurity Applications & Technology Conference For Homeland Security (CATCH)*, pages 237–241. IEEE Computer Society, 2009.
9. Dan Goodin. TJX breach was twice as big as admitted, banks say. The Register, 24 October 2007.
10. Todd Heberlein. Tactical operations and strategic intelligence: Sensor purpose and placement. Technical Report TR-2002-04.02, Net Squared, Inc., 9 September 2002 2002.
11. Nam Nguyen, Peter Reiher, and Geoffrey H. Kuenning. Detecting insider threats by monitoring system call activity. In *2003 IEEE Workshop on Information Assurance*, pages 18–20, United States Military Academy, West Point, 2003. IEEE Computer Society.
12. Marisa Reddy Randazzo, Dawn Cappelli, Michelle Keeney, Andrew Moore, and Eileen Kowalski. U.S. secret service and CERT coordination center/SEI insider threat study: Illicit cyber activity in the banking and finance sector. Technical report, Software Engineering Institute, Carnegie Mellon University, August 2004.
13. Lance Spitzner. Honeypots: Catching the insider threat. In *19th Annual Computer Security Applications Conference (ACSAC '03)*, pages 170–179. IEE Computer Society, 2003.
14. Stuart Staniford, James A. Hoagland, and Joseph M. McAlerney. Practical automated detection of stealthy portscans. *Journal of Computer Security*, 10(1/2):105–136, 2002.