When Gossip is Good: Distributed Probabilistic Inference for Detection of Slow Network Intrusions

Denver Dash
Intel Corporation*
Pittsburgh, PA
denver.h.dash@intel.com

Branislav Kveton
Intelligent Systems Program
University of Pittsburgh
bkveton@cs.pitt.edu

John Mark Agosta
Intel Corporation*
Santa Clara, CA
john.m.agosta@intel.com

Eve Schooler
Intel Corporation*
Santa Clara, CA
eve.m.schooler@intel.com

Jaideep Chandrashekar
Intel Corporation*
Santa Clara, CA
jaideep.chandrashekar@intel.com

Abraham Bachrach
University of California, Berkeley
abachrach@gmail.com

Alex Newman
Rensselaer Polytechnic Institute
newmaa2@networks.ecse.rpi.edu

Abstract

Intrusion attempts due to self-propagating code are becoming an increasingly urgent problem, in part due to the homogeneous makeup of the internet. Recent advances in anomaly-based intrusion detection systems (IDSs) have made use of the quickly spreading nature of these attacks to identify them with high sensitivity and at low false positive (FP) rates. However, slowly propagating attacks are much more difficult to detect because they are cloaked under the veil of normal network traffic, yet can be just as dangerous due to their exponential spread pattern. We extend the idea of using collaborative IDSs to corroborate the likelihood of attack by imbuing end hosts with probabilistic graphical models and using random messaging to gossip state among peer detectors. We show that such a system is able to boost a weak anomaly detector D to detect an order-of-magnitude slower worm, at false positive rates less than a few per week, than would be possible using D alone at the end-host or on a network aggregation point. We show that this general architecture is scalable in the sense that a fixed absolute false positive rate can be achieved as the network size grows, spreads communication bandwidth uniformly throughout the network, and makes use of the increased computation power of a distributed system. We argue that using probabilistic models provides more robust detections than previous collaborative counting schemes and allows the system to account for heterogeneous detectors in a principled fashion.

Intrusion Detection

Worms pose an increasingly serious threat to network security. With known worms estimated at reaching peak speeds of 23K connections per second, and theoretical analysis citing higher speeds, the entire Internet risks infection within tens of minutes (Moore *et al.*, 2003). Because of the virulence of fast scanning worms in the Internet, there is a premium on automated countermeasures. To wait for human intervention is problematic because of the mismatch between the timescale of worm spread and that of human software patching efforts.

Several automated methods have been proposed to prevent, or at least mitigate, the damage caused by worm infections. While some of these methods are focused on prevention, others are aim for detection and containment. Of particular interest in the latter class are intrusion detection systems(IDSs) which rely on searching for signatures of known malware in traffic sent to a network or a node. Examples include SNORT (Roesch, 1999) and Bro (Paxson, 1999). A significant drawback with these traditional IDSs is that they are defenseless against heretofore unknown worms, a.k.a., "day-zero" exploits. More recent work in the area remedies this by building such signatures "on the fly" by searching for frequent anomalous common substrings in observed network traffic (Newsome, Karp, & Song, 2005; Kim & Karp, 2004). An alternative approach is to look for the basic, invariant feature of any scanning worm—the need to make connections to remote machines in search of future victims (Weaver, Staniford, & Paxson, 2004; Eskin, 2000; Lazarevic *et al.*, 2003; Portnoy, Eskin, & Stolfo, 2001). Here, the emphasis is shifted from the task of worm prevention to detecting the worm infection, and subsequently on worm containment. Most of these non-signature based techniques rely on the fact that worms are reproducing quickly.

As the methods to detect worms become increasingly sophisticated, the worm designers react by making worms harder to detect and stop. Worms released over the past year have tended to the extremes: getting either much faster to allow rapid spread, or much slower to prevent detection. The latter approach places an increasing burden on detection methods to effectively pick out and isolate worm traffic from the baseline created by normal traffic seen at a host. While the slower rate does offer some respite to the network operator(s) (if detected, the worms can be contained with relatively little collateral damage), the detection is extremely challenging due to the fact that slow worms can hide under the veil of normal traffic. Although locally a worm may be propagating very slowly, if it can manage to reproduce more than once before being detected on a local host, it will still grow at an exponential rate.

Yet another challenge in dealing with worms is that individual entities can only see a partial picture of the larger network wide behavior of the worm(s). IDSs deployed in

*Intel Research & Communications Technology Labs

select networks might not see any worm traffic for a long time and perhaps see it only when it is too late. Collaboration is seen as a way to remedy this; systems that allow multiple IDSs to share information have been shown to provide greater "coverage" in detection (Bailey *et al.*, 2005; Nojiri, Rowe, & Levitt, 2003; Anagnostakis *et al.*, 2003; Agu Rajab, Monrose, & Terzis, 2005). The existing collaboration based schemes rely on relatively simple counting and thresholding schemes. Collaborating detectors are assumed to be homogeneous, which might not always be the case,[1] and again they typically rely on the speed of worms to make detections.

In this paper, we describe an approach to host-based IDS using distributed probabilistic inference. The starting point in our work is a set of *weak* host-based IDSs, referred to as *local detectors* (LDs), distributed throughout the network. We allow the hosts to collaborate and combine their weak information in a novel way to mitigate the effect of the high false positive rate. LDs raise alarms at a relatively high frequency whenever they detect even a remotely plausible anomaly. Alarms spreading in the network are aggregated by *global detectors* (GDs) to determine if the network, as a whole, is in an anomalous state, e.g., under attack. A similar system of distributed Bayesian network-based intrusion detection is presented by Gowadia, Farkas, & Valtorta (2005). Their work uses a novel procedure called "soft-evidential update" to smooth beliefs during inference. Our work is distinct from theirs in that we (1) separate the local detection model from the global detection model to improve modularity of the system, (2) identify a clear need for such a system (detecting weak/slow network-wide anomalies), (3) empirically test our approach on real data, and (4) demonstrate empirically that our approach is superior to common existing methods.

Our main contribution in this paper is a probabilistic framework that aggregates (local) beliefs to perform network-wide inference. Our primary findings are:

- We can detect an order-of-magnitude slower worm than could be detected by using LDs alone at a FP rate of one per week. [2]
- Our framework shows good scalability properties in the sense that we achieve a fixed false positiv system, independent of the network size.
- Our probabilistic model outperforms previc tive counting schemes and allows the syste for heterogeneous detectors in a principled f

While the methods we describe are quite ge plicable in a wide variety of network settings, results operate over a subset of the Intel enterp In the following sections, we describe the archi system, discussing the advantages and disadva many design points, and we present empiric demonstrate several of the advantages to the system we propose.

Architectural Model

In answer to the challenges posed in the previous section, we propose a system composed of three primary subcomponents, shown in Figure 1: the LDs which make host-level detections, the GDs which make system-wide detections, and the information sharing system (ISS) to share state between the LDs and the GDs.

The LDs live at the end-hosts and are designed to be *weak* but *general* classifiers which collect information and make "noisy" conclusions about anomalies at the host-level. This design serves several purposes:

1. Analysis of network traffic at the host level compares the weak signal to a much smaller background noise-level, so can boost the signal-to-noise ratio by orders of magnitude compared to an IDS that operates within the network.
2. Host-based detectors can make use of a richer set of data, possibly using application data from the host as input into the local classifier.
3. This system adds computational power to the detection problem by massively distributing computations across the end-hosts.

An important design decision of this system is where to place the GDs in the network, and this decision goes hand-in-hand with the design of the ISS. There are at least two possibilities: centralized placement (Figure 1-a) or distributed placement (Figure 1-b). Both placements have advantages and drawbacks; however in this paper, we propose, like the LDs, distributing the GDs onto the end-hosts. The advantages of this approach are:

1. Distributing reduces congestion by preventing link-implosion that can occur at a single centralized detector.
2. One GD in charge of securing the whole system is a single-point of failure and is a vulnerable target for would-be attackers.
3. A collection of GDs on the end-hosts can in principle be used as a distributed ensemble classifier (e.g., using boost-

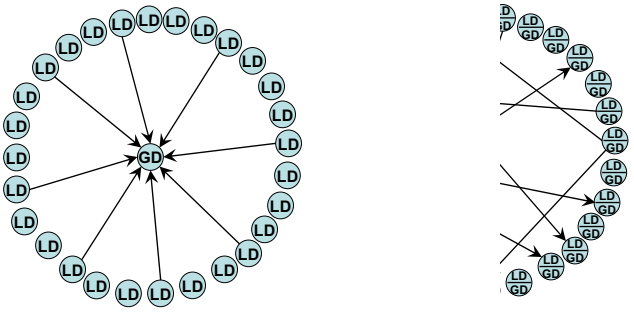

Figure 1: In centralized corroboration, (a), all LDs report to a single central GD. In the fully distributed version, (b), each end-system has both an LD and a GD; nodes share their states with peers.

[1] For instance, a system may employ a range c some detectors may be more trusted than others.

[2] By contrast, the Intel network operations cent vestigates 2-3 false positives each day.

ing or bagging), and distributing this ensemble calculation can again allow more computational power to be brought to bear on the global detection problem.

On top of this *a priori* rationale, we show empirically that in fact a single GD with full information of the system state can perform worse than a simple ensemble of many GDs with partial information when detecting day-zero attacks for which the models are not correctly tuned.

The ISS uses a network protocol to communicate state between the LDs and the GDs. For the purposes of this paper we assume that each LD communicates its state by beaconing to a random set of GDs at regularly spaced epochs. There are many important and interesting research questions about what an ideal ISS should look like. For example, messages could be aggregated from host-to-host to allow exponential spreading of information. However it is beyond the scope of this paper to deal with this issue in depth. Here we assume that no message aggregation is taking place, each LD relays its own state to M hosts, chosen at random each epoch.

In the following sections, we examine in detail the LDs and GDs used by our system.

The Local Detectors

For the purposes of this paper, we define a LD as a binary classifier that sits on the local host and uses information about the local state or local traffic patterns to classify the state of the host as *normal* or *abnormal*. We assume that the LDs are *weak* in the sense that they may have a high false-positive rate, but are *general*, so are likely to fire for a broad range of anomalous behavior. In the context of intrusion-detection systems, because of the high volume of traffic in modern networks, what may appear to be a relatively small FP rate could by itself result in an unacceptable level of interruptions, so could be classified as weak.

The LD implementation we use in this paper is a heuristic-based detector that analyzes outgoing traffic and counts the number of new outgoing connections to unique destination addresses and outgoing ports; alerts are raised when this number crosses a configured threshold. Figure 2 shows this statistic over 37 hosts in the Intel corporate network over a five week period. This figure shows that setting the threshold of this statistic above 200 connections per 50s interval would have caught the Blaster, Slapper, Code Red II, Slammer and Witty worm outbreaks, and would have resulted in about 2 FPs over the entire period and over all hosts. These worms are relatively easy to detect with this heuristic because their propagation rates stand out orders of magnitude higher than typical connection rates in the enterprise. However, in line with our goal of detecting extremely slow worms, we push these LD thresholds to an orders-of-magnitude lower value (4 connections per 50 seconds, shown as the left-most dotted line in Figure 2), well within the bulk of the normal traffic distribution. If a LD by itself was used with this threshold it would have generated thousands of false alarms over the five-week period; however, it would have successfully detected worms operating at a much slower rate. Thus a simple way to create a weak, general LD is to drastically reduce the threshold of some standard heuristic, although other standard anomaly detection techniques can be used as well.

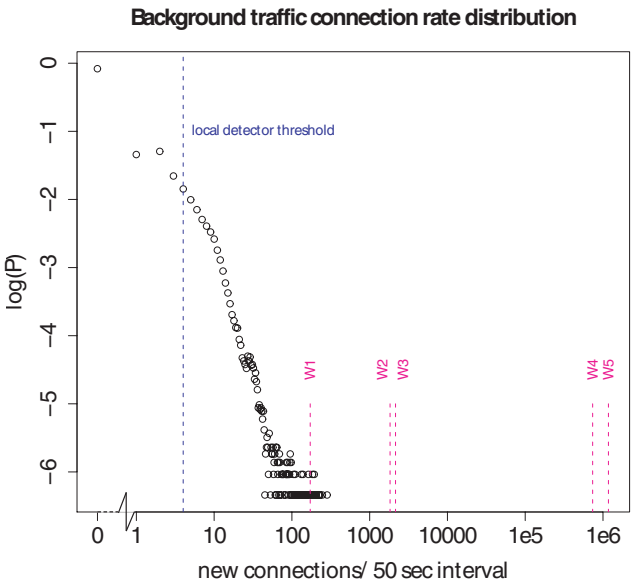

Figure 2: The distribution of the number of new connections to unique destination address-port-pairs per 50s time interval (CPI). The dashed line indicates the threshold (4 CPI) our LDs use to raise local alerts. The propagation rates of some previous worms are also shown: W1=Blaster, W2=Slapper, W3=Code Red II, W4=Slammer, and W5=Witty.

The space of LDs can roughly be partitioned into two classes: those that are *inward-looking* and those that are *outward-looking*. Inward-looking detectors analyze the state of the host itself, including outgoing traffic, looking for anomalous behavior of an infected host. Outward-looking detectors look at incoming packets to decide if an external threat is attempting to make contact with the host. The distinction between outward and inward-looking detectors is important because the dynamics of a system of inward-looking detectors is qualitatively different from outward-looking. We elaborate more on this point in the next Section.

The Global Detectors

Once the information of LD states has been collected at various locations, there remains the question of how to use that information to draw conclusions about a global attack taking place. We consider four possible GD models for this task:

1. The *PosCount* detector aggregates the local information simply by summing the number of positive (anomalous) counts received from the LDs and thresholding the value.
2. The *CuSum* detector (cf., Bissell, 1984) is a standard technique from statistical process control to detect changes in the trend of a statistic.
3. The *change-point dynamic Bayesian network* (CP-DBN) is a simplified causal model that models an attack as occurring uniformly across the population or not at all.
4. The *epidemic dynamic Bayesian network* (E-DBN) models the dynamics of a system that is being swept by an epidemic (exponentially growing) outbreak.

All of these detectors take as input a binary subset L_t of LD observations at time t and output some *signal*, S_t, which is some measure of how likely a global anomaly is to be occurring at time t. A fixed threshold is applied to S_t to decide if a GD fires. The system of GDs makes up an ensemble, and there are many interesting ensemble techniques such as boosting (Schapire, 2001) that could be employed to combine beliefs among this ensemble; however, we do not pursue these possibilities in much detail here. In this paper, we use the $\max$ function as our aggregation function, i.e., we combine the individual signals S_t^l into the aggregate signal $S_t^{max} = \max_l S_t^l$, which is thresholded to determine whether a global alarm is raised or not.

We consider the PosCount and CuSum detectors to be baseline techniques as they have been extensively used for similar applications in the past. The last two we propose as better alternatives, the CP-DBN being especially useful for a system of outward-looking LDs and the E-DBN model for inward-looking LDs. We briefly elaborate on these models below.

The PosCount detector is a simple detector that has been widely used in other distributed detection algorithms that have performed well in detecting fast worms using outward-facing LDs. This technique aggregates by counting events (in our case, the number of positive local detections received over a given time interval). We also apply smoothing to this model by using a sliding window average to smooth epoch-to-epoch variations. Thus for a sequence of N binary (0/1) observations L_t^T observed between time $t-T$ and time t, the signal S_t^{PosCt} generated by PosCount is given by

$$S_t^{PosCt} = \frac{1}{N} \sum_{l_i \in L_t^T} l_i$$

This model is very simple and the information it needs to draw conclusions is quite compact, requiring much smaller messages to communicate local state information. However, this method does not take into account the fact that the network may consist of heterogeneous LDs, some of which may be stronger than others. For example, one instance of a very accurate detector firing is sufficient to raise a global alarm with a low FP rate; whereas weaker detectors may require extensive corroboration.

The CuSum detector is extensively used in statistical process control to detect a deviation from the mean of some statistic. The CuSum variant that we use here considers each binary (0/1) observation $l_i \in L_t$, and recursively computes the signal

$$S_i^{Cus} = max(0, S_{i-1}^{Cus} + l_i - (P+Q)/2),$$

where P and Q are the assumed true positive (TP) and FP rates for the LD with observed state l_i. The CuSum model thus provides an ad hoc mechanism to account for heterogeneous LDs (i.e., LDs with varying TP and FP rates).

The final two models we consider are based on Dynamic Bayesian networks (DBNs) (Dean & Kanazawa, 1990), which are a principled formalism for expressing independence relations while modeling temporal stochastic processes. This approach is motivated by the work of Cooper *et al.* (2004) which used causal Bayesian networks to monitor a large population for disease outbreaks. The signal S_t generated by a DBN model M is a log-posterior odds of an anomaly A_m at the most likely time m given all the evidence L_t^T observed from time $t-T$ to time t:

$$S_t^{DBN} = \log \frac{P(A_m \mid L_t^T, M)}{P(\textit{no anomaly} \mid L_t^T, M)}$$

Because their signal is based on the posterior probability, DBN models have the benefit of incorporating the $\{P, Q\}$ values of the LDs in a principled way.

The "change-point" CP-DBN model assumes that up to some time τ_{cp}, the network as a whole is not in an anomalous state; whereas after τ_{cp} the network is. There are anomalies for which this type of step-function dynamics is expected. For example, when the network address space is being port-scanned by an external threat, then a system of outward-looking LDs would all fire near-simultaneously, causing an abrupt rise in the number of local detections. Other anomalies that might trigger this type of dynamics are network routing mis-configurations and downed servers, which would affect the whole system simultaneously. Typically these types of events would be detectable with an outward-looking LD that is noticing unusual behavior outside an individual host, e.g. by looking at anomalies in the distribution of incoming connections. The CP-DBN model structure is shown in Figure 3. The binary variables $A_i \in \{T, F\}$ cor-

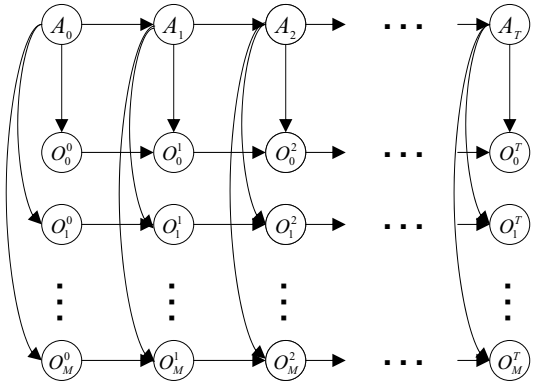

Figure 3: The CP-DBN model.

respond to the event that an attack has taken place at time slice i or not. The variable $O_l^i \in \{on, off\}$ corresponds to the event that detector l is *on* or *off* at time i. There are M total LDs in the population.

This model assumes that once an attack takes place at time i, it remains in place for the duration of the model horizon (N). Thus, the transition probabilities for the A_i variables are written as:

$$P(A_{i+1} = T \mid A_i = T) = 1 \quad (1)$$
$$P(A_{i+1} = T \mid A_i = F) = P_A \quad (2)$$

This model also assumes that for the duration of the time horizon, once a LD fires it remains on; for this we need the explicit arc from $O_l^i \rightarrow O_l^{i+1}$. If LD l has a TP rate of P_l and

a FP rate of Q_l, then the transition matrix of the O_l^i variables can be written as:

$$P(O_l^{i+1} = on \mid O_l^i = on, A_{i+1} = *) = 1 \quad (3)$$

$$P(O_l^{i+1} = on \mid O_l^i = off, A_{i+1} = T) = P_l \quad (4)$$

$$P(O_l^{i+1} = on \mid O_l^i = off, A_{i+1} = F) = Q_l \quad (5)$$

It is easy to show that under assumptions of sparse observations (no LD is observed more than once in the interval $[t-T, t]$), this model reduces to a naive Bayes model, where each observed LD variable l has parameters that depend on the time slice at which the feature is observed and the assumed TP rate P_l and FP rate Q_l of detector l. Using this naive approximation, inference with this model can be performed in time linear in the number of observations.

The *epidemic* E-DBN models the spread of exponentially growing anomalies within the system. This type of dynamics is relevant for inward-looking LDs during the spread of an anomaly, and thus is most relevant for detection and containment of a worm that has infected the system. To model this growing trend, we construct a DBN over random variables $A_1, \ldots, A_T$, $N_1, \ldots, N_T$, S, and $O_1, \ldots, O_T$, where T denotes the length of the time horizon, $A_t = \{0, 1\}$ is the anomaly state at time t, $N_t = \{0, \ldots, N\}$ is the number of infected computers, S is the spreading rate of a worm, and $O_t = \{0, \ldots, N\}$ is the number of observed host detectors that fired. The variables A_t, N_t, and S are unobserved. Note that both N_t and O_t are only statistics in the state and observation spaces. Thus, we do not represent each host explicitly as in the CP-DBN model. The transition model between the unobserved state variables is defined as:

$$P(A_{t+1}, N_{t+1} \mid A_t, N_t, S) = P(A_{t+1} \mid A_t) P(N_{t+1} \mid N_t, A_{t+1}, S), \quad (6)$$

where $P(A_{t+1} \mid A_t)$ models the dynamics of the binary anomaly state, and $P(N_{t+1} \mid N_t, A_{t+1}, S)$ models our beliefs in the number of infected computers given the hypothesis A_{t+1}. A graphical representation of our model is shown in Figure 4.

Assuming a worm attack, the growth rate in the number of infected computers ΔN_{t+1} is modeled by a binomial (Chen, Gao, & Kwiat, 2003):

$$P(\Delta N_{t+1} \mid N_t, A_{t+1} = 1, S) = P_{\text{binom}}\left(\Delta N_{t+1} \mid N - N_t, 1 - (1 - 1/N)^{SN_t}\right), \quad (7)$$

where $N_{t+1} = N_t + \Delta N_{t+1}$. To explain the stochastic generative process in Equation 7, we assume that a worm attacks N hosts at random and every attack succeeds with probability 1. The chance of a single host not being hit by an individual attack equals to $(1 - 1/N)$. Consequently, the chance of not being hit by SN_t independent attempts is $(1-1/N)^{SN_t}$, and thus the chance of a hit is $1-(1-1/N)^{SN_t}$. Therefore, the number of newly infected computers follows a binomial distribution with mean $[1 - (1 - 1/N)^{SN_t}](N - N_t)$.

We put no restriction on the form of the transition function $P(N_{t+1} \mid N_t, A_{t+1} = 0, S)$. Although it is natural to expect no infected computers if the network is not attacked, we can employ the hidden state N_t to explain spatial correlations of observations. Note that these correlations are likely to have different temporal patterns than the trend in Equation 7. The transition matrix between N_t and N_{t+1} can be learned from historical data, for instance by the expectation-maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977).

The likelihood of o_t detectors firing when n_t computers are infected is modeled by a binomial:

$$P(O_t = o_t \mid N_t = n_t) = P_{\text{binom}}(o_t \mid N, p), \quad (8)$$

where $p = (n_t p_{\text{tp}} + (N - n_t) p_{\text{fp}})/N$. This approximation follows from $P_{\text{binom}}(n \mid N, p) \approx \mathcal{N}(n \mid Np, Np(1-p))$ and the fact that the sum of two independent normal variates is a normal variate.

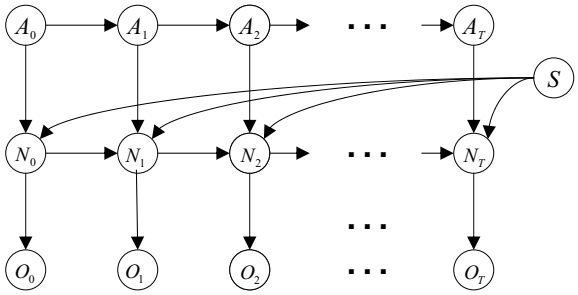

Figure 4: The E-DBN global model for detecting worm-like spreading of anomalies.

An obvious alternative to the generative DBN models we propose here is to use standard discriminative models such as SVMs or decision trees as the GDs. The use of these models is complicated by the fact that we do not possess any labelled positive cases on which to train a decision boundary, because we are interested in detecting "day-zero" events that by definition have never occurred in the past. Even using previous instances of anomalies with semi-supervised learning does not solve this dilemma because the positive instances we have observed in the past will by definition not contain day-zero threats. Instead, we make use of a wealth of historical negative instances, and tune our generative model thresholds to produce a given expected FP rate. An additional advantage of causal models is that they provide a principled approach to incorporating background knowledge that could be essential in detecting various classes of day-zero attacks, as illustrated by the different dynamics modeled by the CP-DBN and the E-DBN models. Another example of useful background knowledge is signal variation based on time of day and day of the week. The data that we use in our experiments demonstrates very strong diurnal patterns, and our models would likely benefit from such information, although we do not consider this particular optimization in this paper.

Simulation Studies

In this section we present a set of results that highlight the advantages of the system we are proposing. The algorithms above were tested on five weeks of traffic data for 37 hosts within our enterprise network. As detailed in the previous

section, the LDs we used were inward-looking detectors that scanned outgoing packets and counted the number of UDP packets[3] to unique destination IPs, itemized by port. The histogram for this statistic aggregated over all machines for all five weeks is shown in Figure 2. We operated the LD at a low threshold of 4 connections per 50 second interval. To decrease run times, the CP-DBN implementation we used assumed sparse observations and thus was reduced to the naive approximation as discussed previously.

The simplicity of our LDs made simulating worm behavior on top of real network traffic trivial. We modeled a worm with two parameters: the spread rate S which indicated how many hosts the worm attempted to spread to per unit time, and the address density with which the worm was able to spread. For example, if the worm is randomly choosing addresses from the entire 32-bit address space and we have a network of 37 hosts, then the address density will be $37/2^{32}$. All results shown here use an S of 1 connection per 20 seconds, and set the address density to 1/1000. We have many results for which these two parameters have been varied; the performance is relatively insensitive to the worm speed but much more sensitive to the address space density. Thus a worm with a perfect "hit-list" and a divide-and-conquer strategy would likely do a pretty good job of defeating this system. Our simulations, implemented in C++, superimposed artificial worm traffic on the real enterprise traffic by releasing worms, letting the proportion of infected hosts grow to 1 in the population, then repeating 20 times. To test for FP rates we simply used the enterprise traffic with no worm traffic superimposed. The results were averaged over all runs.

All GD types tested, except the PosCount model, required an estimate for P and Q, TP and FP rates of the LDs, respectively. All models assumed for these experiments that the system was comprised of a set of homogeneous detectors. This was a correct assumption in so far as each LD used the identical heuristic and an identical threshold, but may not have been true in the sense that the traffic processed by each LD was distinct, possibly resulting in different FP rates. We used homogeneous LDs so that our results would not be biased in favor of the DBN models which were expected to deal with heterogeneous detectors better (we save a quantitative measurement of how heterogeneous detectors affect the various techniques for future work). For all results we used values $P = 0.6$ and $Q = 0.2$, although, we verified empirically that for the homogeneous case, these parameters could vary by an order of magnitude while having almost no impact on the resulting performance curves.

We measure performance of the GDs by sweeping through all GD thresholds and plotting the percentage of machines infected when a global detection is made (presumably we are able to quarantine infected machines at this point) versus the FP rate. Our results were invariant to wide parameter variations precisely because we sweep through all threshold values. Much like an ROC curve, changing the P and Q values widely will cause a shift in the threshold, but

[3]Many of these experiments have been repeated with TCP packets, and the results so far have all been qualitatively similar.

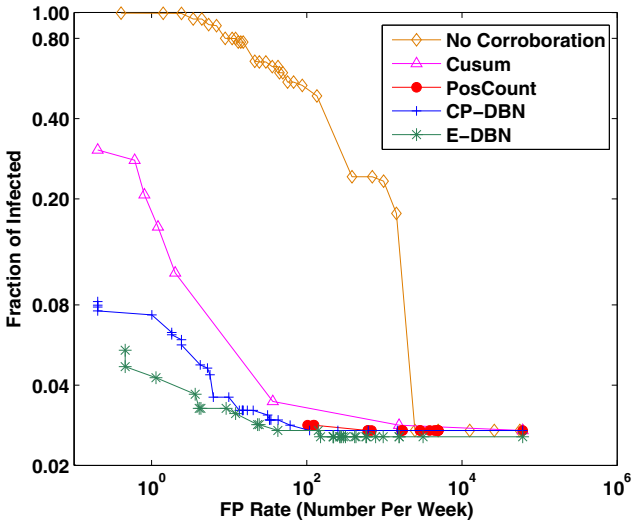

Figure 5: The performance of the 4 GD models on 5 weeks of enterprise data compared to an individual LD detector.

will have relatively little impact on the shape of the curve. In practice, the Q value should be measured on historical data and the threshold should be calibrated to the desired FP rate via cross-validation. The P value cannot be measured directly since we are attempting to detect day-zero events. Having the estimated P value significantly different from the actual P value is expected to cause a decrease in sensitivity in the global detector performance. One method for dealing this problem is to perform Bayesian averaging over P values, an approach we are pursuing.

Figure 5 shows the results for all GDs when the LDs pass one message per 10 second epoch. The ideal point on this curve is in the lower-left corner where a low FP rate is achieved but a global alarm is triggered before any hosts are infected. Both DBN models clearly outperformed the baseline GD models. The PosCount detector[4] at a FP rate of 100 per week will only raise a detection after the entire network is infected. The CuSum detector is able to operate at our target FP rate of 1 per week, but it detects at a much higher infection percentage ($> 20\%$) than the DBN models. The CP-DBN, which is not designed to detect an epidemic spread can still achieve the 1 FP per week while allowing only 8% infection, while the E-DBN model which was specifically designed for this scenario, can detect with an infected-host percentage of about 4%. The top line shows the results that are obtained as you sweep through the LD thresholds with no corroboration.

There are several questions about the scalability of these results. One question that is useful to answer is whether we can achieve similar absolute FP rates as the number of hosts scales, or whether the FP rate increases as the number of

[4]Although not clear from the graph, the PosCount curve essentially becomes a vertical line going to 1.00 at a FP rate of 100 per week.

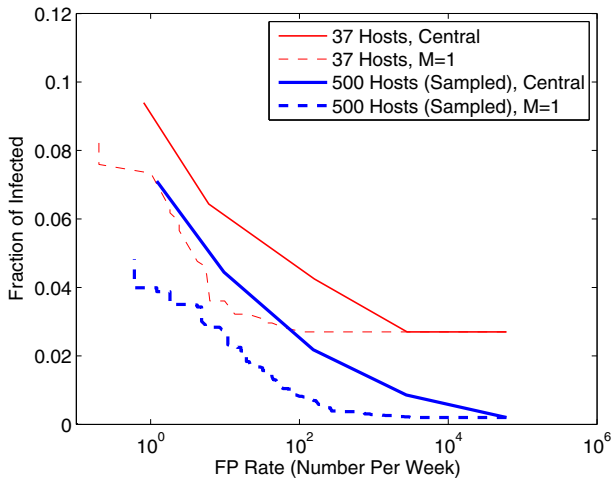

Figure 6: CP-DBN results when data is bootstrapped to simulate 500 hosts. Both centralized and distributed approaches show improvements compared to 37 hosts.

hosts increases. We tested this question by bootstrapping the enterprise data to 500 hosts by picking one of the 37 hosts at a random time and using a ring buffer to maintain a continuous stream of data. With this methodology we can increase the effective population of our 37 hosts to an arbitrary number. The result of this experiment for the CP-DBN model for 500 hosts is shown in Figure 6. This figure shows the original 37 host results side-by-side with the 500 host results. As shown here, the 500 hosts results performed even better than the 37 hosts for the CP-DBN model. We also tested the ring buffer methodology by sampling only 37 hosts–these results were slightly worse than the results on the original data indicating at least that there does not appear to be a systematic improvement-bias introduced by sampling. These results provide evidence that the absolute FP rate can scale with the network size. There are other scaling issues in this system that we have not fully explored, such as how to develop a scalable ISS.

Figure 6 also shows the results of the CP-DBN model that is fully centralized, i.e., at each epoch, *all* hosts report in to a single GD. Although messaging for this technique has serious scalability issues, we believed it would form an informative baseline to evaluate the amount of information needed to be passed per epoch to achieve good results. The possibly surprising fact for the CP-DBN is that a single version of CP-DBN with N observations received per epoch actually performed worse than N CP-DBNs with on average 1 observation received per epoch. This effect was tested more stringently in Figure 7 where fraction of hosts infected at the 0-FP point is plotted versus the number of messages passed per 10s epoch. The number of messages was varied from 1/64 (1 message per 64 epochs) to 4. The point labelled "C" shows the results of a single centralized GD. These results show that, for the CP-DBN model and for the parameters used in our experiments, there exists an optimal message spread rate at about 1 message per 4 epochs.

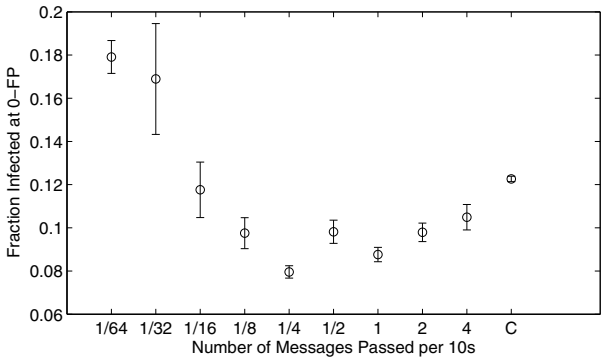

Figure 7: Many aggregators with fewer messages can outperform a single centralized aggregator with full information.

There are a few possible explanations for these results. One is the simple fact that the naive approximation for the CP-DBN model assumes sparse observations. As the number of messages passed per epoch increases, this assumption can be violated. In the centralized case this assumption is guaranteed to be violated. Another possible explanation is that, since the centralized detector does not have a correct estimate of the P values (presumably a common occurrence for day-zero threat), it can become overconfident with much information, and actually perform worse. Similar experiments with the E-DBN model show that this effect is less prominent, but still present. This fact provides support for the latter hypothesis: because the E-DBN model is closer to the epidemic outbreak being modeled, the over-confidence should be less severe; however, it still does not possess an accurate P value, so it should still be present. There is yet another effect to consider. The centralized detector with many observations will obviously possess a higher variance on likelihood ratios than will the max over many global detectors with just a few observations (there are exponentially fewer combinations available to the latter model). The higher variance may translate to poorer detection time. More work is needed to disambiguate these hypotheses.

Conclusions

We consider the problem of detecting very weak or slowly-spreading "day-zero" network-wide anomalies. We argue that to separate the weak signal from the background traffic, it is useful to split the data into smaller data streams, and that the obvious place to do that is at the end-host. We argue that causal generative models provide a natural mechanism to aggregate data for unknown day-zero problems due to the ability to account for background knowledge and heterogeneous LDs in a principled fashion. We introduced the distinction between inward-looking and outward-looking LDs, and showed that systems of the former possess qualitatively different dynamics than systems of the latter. We introduced two DBN models that are useful for aggregation of LD information, one suitable for inward-looking and one suitable for outward-looking LDs, and we showed

that both of these models outperformed state-of-the-art techniques on real traces from an enterprise network using a simple inward-looking set of homogeneous LDs. Finally, our results raised questions regarding what should be the optimal amount of information to share when the GD being used differs from the real world.

There are many future directions to take this work. One obvious direction is to understand to what degree heterogeneous local detectors will impact the various techniques. To do so requires a cross-validation step to calibrate the global thresholds according to the assumed local TP and FP rates. More extensive analysis and empirical work needs to be done to understand the results of Figure 7. There are other advantages to moving detection onto the hosts, such as the ability to analyze internal application state data to improve local detections. It would be worthwhile to explore more elaborate local detectors based on these other sources of data. There is the issue of how to train the local detector parameters online—it might be feasible to use the GDs to soft-label the data in an EM-type algorithm to tune LD parameters in real time. Finally, there is the issue of how to deal with the absence of positive training instances; selecting reasonable and tractable priors to allow model averaging over P is one obvious direction.

Acknowledgments

We would like to thank Nina Taft and Toby Kohlenberg for much feedback and encouragement for this work. We would also like to thank David Durham, Uday Savagaonkar, Ravi Sahita, Gayathri Nagabhushan and Priya Rajagopal for the use of their local detector code in our experiments. Finally, we would like to thank Emily Fisher and David Thomas for their help in collecting enterprise traces for use in our experiments.

References

Agu Rajab, M.; Monrose, F.; and Terzis, A. 2005. On the effectiveness of distributed worm monitoring. Technical report, John Hopkins University.

Anagnostakis, K.; Greenwald, M.; Ioannidis, S.; Keromytis, A.; and Li, D. 2003. A cooperative immunization system for an untrusting internet. In *Proceedings of the 11th IEEE International Conference on Networks (ICON)*.

Bailey, M.; Cooke, E.; Jahanian, F.; Nazario, J.; and Watson, D. 2005. The internet motion sensor: A distributed blackhole monitoring system. In *Proceedings of the Network and Distributed System Security Symposium Conference 2005*.

Bissell, A. 1984. *An Introduction to CuSum Charts*. The Institute of Statisticians.

Chen, Z.; Gao, L.; and Kwiat, K. 2003. Modeling the spread of active worms. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*.

Cooper, G.; Dash, D.; Levander, J.; Wong, W.-K.; Hogan, W.; and Wagner, M. 2004. Bayesian biosurveillance of disease outbreaks. In *Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence (UAI-04)*, 94–103. Arlington, Virginia: AUAI Press.

Dean, T., and Kanazawa, K. 1990. A model for reasoning about persistence and causation. *Comput. Intell.* 5(3):142–150.

Dempster, A.; Laird, N.; and Rubin, D. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B* 39(1):1–38.

Eskin, E. 2000. Anomaly detection over noisy data using learned probability distributions. In *Proceedings of the International Conference on Machine Learning, 2000*.

Gowadia, V.; Farkas, C.; and Valtorta, M. 2005. Paid: A probabilistic agent-based intrusion detection system. *Computers & Security* 24(7):529–545.

Kim, H.-A., and Karp, B. 2004. Autograph: Toward automated, distributed worm signature detection. In *USENIX Security Symposium*, 271–286.

Lazarevic, A.; Ozgur, A.; Ertoz, L.; Srivastava, J.; and Kumar, V. 2003. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of Third SIAM International Conference on Data Mining*.

Moore, D.; Paxson, V.; Savage, S.; Shannon, C.; Staniford, S.; and Weaver, N. 2003. The spread of the sapphire/slammer worm. Technical report, CAIDA.

Newsome, J.; Karp, B.; and Song, D. X. 2005. Polygraph: Automatically generating signatures for polymorphic worms. In *IEEE Symposium on Security and Privacy*, 226–241.

Nojiri, D.; Rowe, J.; and Levitt, K. 2003. Cooperative response strategies for large scale attack mitigation. In *Proceedings of the 3rd DARPA Information Survivability Conference and Exposition, April 2003*.

Paxson, V. 1999. Bro: a system for detecting network intruders in real-time. *Comput. Networks* 31(23-24):2435–2463.

Portnoy, L.; Eskin, E.; and Stolfo, S. J. 2001. Intrusion detection with unlabeled data using clustering. In *Proceedings of the ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001)*.

Roesch, M. 1999. Snort - lightweight intrusion detection for networks. In *LISA '99: Proceedings of the 13th USENIX conference on System administration*, 229–238. Berkeley, CA, USA: USENIX Association.

Schapire, R. 2001. The boosting approach to machine learning: An overview. In *MSRI Workshop on Nonlinear Estimation and Classification*.

Weaver, N.; Staniford, S.; and Paxson, V. 2004. Very fast containment of scanning worms. In *USENIX Security Symposium*, 29–44.