Special issue: The future of software engineering for security and privacy

Research Paper

CAMNEP: An intrusion detection system for high-speed networks

Martin REHÁK[1], Michal PĚCHOUČEK[2], Karel BARTOŠ[3], Martin GRILL[4], Pavel ČELEDA[5], and Vojtěch KRMÍČEK[6]

[1,2,3,4]*Department of Cybernetics and Center for Applied Cybernetics, Czech Technical University in Prague*

[5,6]*Institute of Computer Science, Masaryk University*

ABSTRACT

The presented research aims to detect malicious traffic in high-speed networks by means of correlated anomaly detection methods. In order to acquire the real-time traffic statistics in NetFlow format, we deploy transparent inline probes based on FPGA elements. They provide traffic statistics to the agent-based detection layer, where each agent uses a specific anomaly detection method to detect anomalies and describe the flows in its extended trust model. The agents share the anomaly assessments of individual network flows that are used as an input for the agent's trust models. The trustfulness values of individual flows from all agents are combined to estimate their maliciousness. The estimate of trust is subsequently used to filter out the most significant events that are reported to network operators for further analysis. We argue that the use of trust model for integration of several anomaly detection methods and efficient representation of history data shall reduce the high rate of false positives (legitimate traffic classified as malicious) which limits the effectiveness of current intrusion detection systems.

KEYWORDS

Intrusion detection, network behavior analysis, multi-agent system, trust, anomaly detection

1 Introduction

Increasing Internet traffic obliges the backbone operators and large end users to deploy high-speed network links to match the bandwidth demands. The increase of bandwidth is apparent not only on backbone links, but the consumer hosts are more and more connected with bandwidth capacity that was available only for enterprise clients few years ago. However, besides all beneficial effects, this new high-bandwidth infrastructure presents novel challenges in the domain of security and robustness, as the manual oversight of such high traffic volumes is nearly impossible and only the events of extraordinary scale are typically reported [10].

Received September 13, 2007; Revised November 29, 2007; Accepted December 11, 2007.

[1)] *mrehak@labe.felk.cvut.cz*, [2)]*pechouc@labe.felk.cvut.cz*, [5)]*celeda@ics.muni.cz*, [6)]*vojtec@ics.muni.cz*

DOI: 10.2201/NiiPi.2008.5.7

Our research aims to give the operators of these networks a highly autonomous network intrusion detection system (NIDS), which will use network behavior analysis techniques as outlined in recent classification [17]. Network behavior analysis systems are not based on the signature matching, but use the network traffic statistics acquired in form of TCP/IP flows to identify relevant malicious traffic, such as vertical scanning (used to determine the services offered by a host), horizontal scanning (used to map the network for on-line hosts, and used for worm and malware propagation [5]), denial of service attacks and other relevant events. Our approach does not aim to detect targeted compromise of single hosts (this service being provided by IDS deployed on LANs), but concentrates on the events that are significant on the network level, providing broader perspective on current threats and allowing the opera-

tors to address the most significant ones. Furthermore, the methods outlined in our system aim to detect the activity of the hosts in their networks that were taken over by an attacker (typically using zombie networks [5]) and are used to stage further zombie recruiting, DoS attacks [10] or spam propagation.

Our work is based on the analysis of the network traffic statistics. We aggregate the network data to capture the information about network flows, unidirectional components of TCP connections (or UDP, ICMP equivalent) identified by shared source and destination addresses and ports, together with the protocol, and delimited by the time frame used for data acquisition (see Section 3.1). This information provides no hint about the content of the transmitted data, but by detecting the anomalies in the list of flows acquired over the monitoring period, we can detect irregularities and possible attacks.

2 Related work

In order to detect an attack from the flow information on the backbone level, especially without any feedback from the affected hosts, it is necessary to analyze the behavioral patterns in the traffic data, compare them with normal behavior and conclude whether the observed anomalies are caused by attacks, or are mere false alarms. This approach to Network Intrusion Detection, typically based on the flow information captured by network flow probes is currently an important field of research into *Anomaly Based Intrusion Detection*, or more specifically *Network Behavior Analysis*. Numerous existing systems, based on traffic volume analysis modeled by *Principal Component Analysis* (PCA) methods [7], models of entropy of IP header fields for relevant subsets of traffic [9], [22], or just count of the flows corresponding to the selected criteria [6] offer each a particular valid perspective on network traffic.

The MINDS system [6] represents the flow by basic NetFlow aggregation features (srcIP, srcPrt, dstIP, dstPrt, protocol) and complements them by the number of the flows from the same srcIP, to the same dstIP and their combinations with dstPrt and srcPrt respectively. These properties are assessed both in time and number of connections defined windows, to account for slow scanning. The anomalous traffic is detected by means of clustering, using local outlier factor, and the operator can use dedicated system modules to support him when creating the rules describing identified attacks, so that they can be applied to future traffic.

The system proposed by Xu et al. [22] for traffic analysis on backbone links also uses the NetFlow based identity 5-tuple. The context of the single connection is defined by the normalized entropy of srcPrt, dstPrt and dstIP dimensions of the set of all connections from the given host in the current time frame. These three dimensions define a feature space, where the method applies predefined rules to separate legitimate and malicious traffic.

Two methods adopted from the work of Lakhina et al. [8], [9] share several common properties: they were originally designed for detection of anomalous *origin-destination flows*, defined as a sum of all flows arriving from one network and leaving to another network. In order to identify the anomalous behavior, both variants use *Principal Component Analysis* (PCA) to model the normal traffic, using the past flow sets to predict traffic volumes [8]. Differences between predicted and observed values are then used to identify the anomalies.

In the alternative approach [9], the PCA method is used to model the normal and residual entropy of the destination IP addresses and source and destination ports of all flows originating from the same srcIP. Subtracting the modeled values from the observed data subtracts normal traffic entropies and allows us to concentrate on anomalous residual entropy. Technically, the approach is analogous to the previous method, the only difference is in the fact that we model three parameters instead of one, and that these parameters are based on entropies, rather than volumes.

Besides the above mentioned sample of backbone anomaly detection mechanisms, there are numerous research and commercial systems designed to protect local networks. A typical representative of recently developed system is a SABER [18], which addresses not only threat detection, but attempts to actively protect the system by automatically generated patches. Technical perspective on many existing IDS systems, including SNORT [20] and other *signature matching* techniques that detect intrusions by detecting patterns specific to known attacks in network traffic, can be found in [13]. A good, even if slightly outdated review of classic research and systems in the domain is provided by [2].

3 Architecture

In our approach, we have decided not to develop a novel detection method, but rather to integrate several methods [6], [8], [9], [22] with an extended trust models of a specialized agent. This combination allows us to correlate the results of the used methods and to combine them to improve their effectiveness. Most anomaly detection methods today are not fit for commercial deployment due to the high ratio of misclassified traffic. While their current level of performance is a valid scientific achievement, the costs associated with supervision of such systems are prohibitive for most organizations. Therefore, our main research goal is to com-

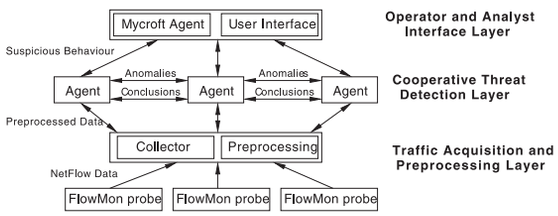


Fig. 1 System architecture overview.

bine the efficient low-level methods for traffic observation, with multi-agent detection process to detect the attacks with comparatively lower error rate, and to provide the operator with efficient incident analysis layer. This layer supports operator's decisions about detected anomalies by providing additional information from related data sources. The layer is also responsible for visualization of the anomalies and the detection layer status.

The architecture consists of several layers with varying requirements on on-line processing characteristics, level of reasoning and responsiveness. While the low-level layers need to be optimized to match the high wire-speed during the network traffic acquisition and preprocessing, the higher layers use the preprocessed data to infer the conclusions regarding the degree of anomaly and consecutively also the maliciousness of the particular flow or a group of flows. Therefore, while the computation in the higher layers must be still reasonably efficient, the preprocessing by the lower layers allows us to deploy more sophisticated algorithms. The system can be split into these layers, as shown in Fig. 1.

Traffic acquisition and preprocessing layer

The components in this layer acquire the data from the network using the hardware accelerated NetFlow probes [3] and perform their preprocessing. This approach provides the real-time overview of all active unidirectional connections on the observed link. In order to speed-up the analysis of the data, the preprocessing layer aggregates meaningful global and per-flow (or group of) characteristics and statistics.

Cooperative threat detection layer

This layer principally consists of specialized, heterogeneous agents that seek to identify the anomalies in the preprocessed traffic data by means of their extended trust models [16]. Their collective decision regarding the degree of maliciousness of a flow with certain characteristics use a reputation mechanism. The agents run inside the AGLOBE agent platform [19] and use its advanced features like agent migration and cloning to adapt the system to the traffic and relevant threats.

Operator and analyst interface layer

This layer is responsible for interaction with the network operator. The main component is an intelligent visualization agent that helps the operator to analyze the output of the detection layer, by putting the processed anomaly information in context of other relevant information. When the detection layer detects suspicious behavior on the network, it is reported to visualization. The visualization agent then opens the new case and retrieves relevant information from available data sources. The network operator can explore and evaluate the reported case subsequently. Another part of this layer is a set of lightweight, specialized visualization agents that allows the operator to follow only the selected characteristics of the system.

3.1 Traffic acquisition and preprocessing

The traffic acquisition and preprocessing layer is responsible for network traffic acquisition, data preprocessing and distribution to upper system layers. It only uses the flow characteristics based on information from packet's headers.

In general, each flow is a set of packets defined by common source and destination IP address, port numbers and protocol. Flows are thus unidirectional and all their packets travel in the same direction. For the flow monitoring we use NetFlow protocol developed by Cisco Systems [4].

The amount of traffic in nowadays high-speed networks increases continuously and traffic characteristics change heavily in time (network throughput fluctuation due to time of day, server backups, DoS attacks, scanning attacks, etc.). Performance of network probes must be independent of such states and behave reliably in all possible cases. The quality of provided data significantly effects the upper layers and chances to detect traffic anomalies.

Therefore we use hardware accelerated NetFlow probes called FlowMon. The FlowMon probe is a passive network monitoring device based on the COMBO hardware [3], which provides high performance and accuracy. The probe handles 1 Gb/s traffic at line rate in both directions and exports acquired NetFlow data to different collectors.

The collector servers store incoming packets with NetFlow data from FlowMon probes into its internal database. Each collector server provides interface to graphical and text representation of raw network traffic, simple flow filtration, aggregation and statistics evaluation, using source and destination IP addresses, ports and protocol.

To process acquired IP flows by upper system layers the preprocessing must be performed on several levels and in different manners. Packets can be sampled (ran-

dom, deterministic or adaptive sampling) on input and the sampling information is added to NetFlow data. On the collector side several statistics (average traffic values, entropy of flows) are computed. The collectors provide traffic statistics via *tasd* (Traffic Acquisition Server Interface Daemon) interface to the AGLOBE agent platform.

Even after probes deployment in monitored network, the probes can be reprogrammed to acquire new traffic characteristics. The system is fully reconfigurable and the probes can adapt their features and behavior to reflect the changes in the agent layer.

3.2 Cooperative threat detection

The goal of the cooperative threat detection layer is to provide the assessment of maliciousness of the individual flows in each flow set observed by the system. To achieve this goal, we use the trust modeling techniques, and extend them to cover the domain-specific needs.

Each detection agent contains one of the anomaly detection methods (discussed in Section 2 and detailed in [14]), coupled with an extended trust model defined in [16]:

- MINDS agent [6], which reasons about the number of flows from and towards the hosts in the network, and detects the discrepancies between the past and current traffic,
- Xu agent [22], which reasons about the traffic from individual hosts using the normalized entropies and rules,
- Lakhina Entropy [9] agent, which builds a model that predicts the entropy of traffic features from individual hosts and identifies anomalies as differences between predicted and real value, and
- Lakhina Volume agent [8], which applies the same method to traffic volumes.

All agents, regardless of their type, process the data received from the acquisition layer in three distinct stages (see Fig. 2):

- anomaly detection,
- trust update,
- collective trust conclusion.

Before the description of each these three stages, we will define the terms and techniques used in our approach. *Trust* of x in y is defined by Marsh as "*x expects that y will behave according to x best interests, and will not attempt to harm x*" [12]. In the network security domain, low trustfulness of the flow means that the flow is considered as a part of an attack. Trustfulness is determined in the $[0, 1]$ interval, where 0 corresponds to

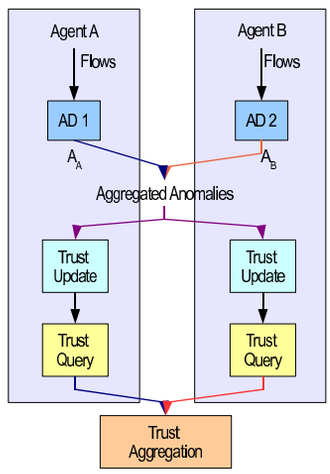


Fig. 2 Agent layer operation.

complete distrust and 1 to complete trust. The *identity* of each flow is defined by the features we can observe directly on the flow: *srcIP, dstIP, srcPrt, dstPrt, protocol*, number of *bytes* and *packets*. If two flows in a data set share the same values of these parameters, they are assumed to be identical. The *context* of each flow is defined by the features that are observed on the other flows in the same data set, such as the number of similar flows from the same *srcIP*, or entropy of the *dstPrt* of all requests from the same host as the evaluated flow. While the agents in our system use the same representation of the identity, the context is defined by the features used by their respective anomaly detection methods to draw the conclusions regarding the anomaly of the flow. Identity and context are used to define the *feature space*, a metric space on which the trust model of each agent operates [16]. The metrics of the space describes the similarity between the identities and contexts of the flows, and is specific to each agent.

Anomaly detection

During the anomaly detection stage, the agents use the embedded anomaly detection method to determine the anomaly of each flow as a value in the $[0, 1]$ interval, where 1 represents the maximal anomaly, and zero no anomaly at all. The anomaly values are shared with other detection agents, and used as an input in the second phase of the processing.

Trust update

During the trust update, the agents integrate the anomaly values determined for individual flows during the first phase into their trust models. As the reasoning about the trustfulness of each individual flow is both computationally infeasible and unpractical (the flows

are single shot events by definition), the model holds the trustfulness of significant flow samples (e.g. centroids of (fuzzy) clusters) in the identity-context space, and the anomaly of each flow is used to update the trustfulness of centroids in its vicinity. The weight used for the update of the centroid's trustfulness with the anomaly values provided for the flow decreases with distance from the centroid. Therefore, as each agent uses a distinct distance function, each agent has a different insight into the problem — the flows are clustered according to the different criteria, and the cross correlation implemented by sharing of the anomaly values used to update the trustfulness helps to eliminate random anomalies.

Collective trust estimation

In the last stage of processing, each agent determines the *trustfulness* of each flow (with an optional normalization step), all agents provide their trustfulness assessment (conceptually a reputation opinion) to the aggregation agents and the visualization agents, and the aggregated values can then be used for traffic filtering.

In order to be successful, the trustfulness aggregated by the system should be as close as possible to the maliciousness of the flow. When we reason about the malicious and untrusted flows as sets (they are actually fuzzy sets), we wish them to overlap as much as possible. We can define the common misclassifications errors using the trustfulness and maliciousness of the flow. The flows that are malicious and trusted are denoted as *false negatives*, and the flows that are untrusted but legitimate are denoted *false positives*. Typically, when we tune the system to reduce one of these sets, the size of the other increases. Intuitively, it may seem that we may be ready to ignore higher rate of false positives, rather than false negatives. However, this is rarely the case in the IDS systems deployed for operational use, as the legitimate traffic vastly outnumbers the attacks and even a low rate of false positives makes the system unusable [1].

Performance of an isolated detection agent would be similar to the performance of the anomaly detection method it is based on. The application of trust modeling mechanism in the layer above the anomaly detection allows the agents to eliminate probable false positives identified as malicious only by a single method. We shall note that each agent represents the flows in its feature space, and that the context subspace definition depends on the anomaly detection algorithm applied by the agent. The context of each flow depends on the (*i*) flow identity, (*ii*) characteristics of other (similar) flows in the current observed flow set, and (*iii*) the current state of the agent's anomaly detection model, which is typically based on past flow sets. This implies two defining features of our methodology, *method integration* and *history integration*.

Method integration

Proximity in the identity-context space of one of the agents doesn't imply the proximity in the identity-context space of another agent — as all agents use different contexts to compare the traffic, the anomalies signaled by a single agent will likely be noticed if they are consistent with the other anomalies (provided by the same agent or different agents) of *similar* flows, that share the centroids. The key aspect is that flow similarity is assessed by each trusting agent using its own criteria — when one of the agents signals an anomaly in the flow that falls into the centroid with mostly trusted traffic, this anomaly will be dispersed and will not manifest itself in the trustfulness evaluation. On the other hand, when the anomaly falls into/near the cluster with existing lower trustfulness (or a new cluster), and is consistently reported by most anomaly providers, the agent will return a low trustfulness for this flow. The Fig. 3, illustrates the concept — all attack flows are concentrated next to a single centroid in agent's trust model, and the situation is similar in the trust models of other agents. The legitimate traffic is dispersed among other clusters.

History integration

Trust model provides an efficient method for the integration of the past observations by aggregating the trustfulness values from the history of anomalies of

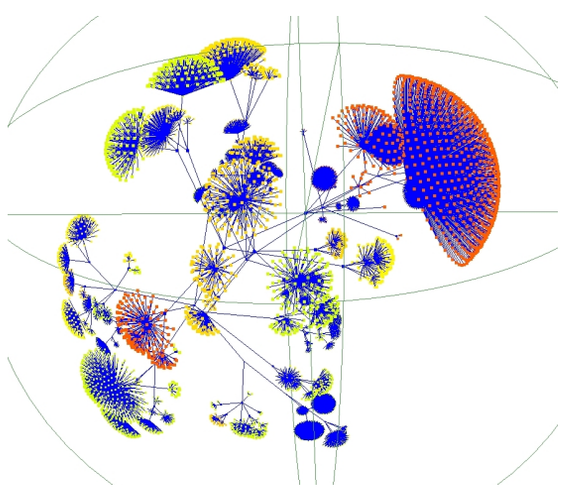


Fig. 3 A peek into the trust model of a detection agent. Attack flows displayed as tree extremities attached to the closest centroid of the trust model, shown as a chrysanthemum-like structure on the right side of image. In the background, we can see other centroids, with the rest of the more trusted traffic attached to them.

similar flows acquired in the previous data sets. This aggregation also helps to eliminate individual anomaly peaks specific to one flow set that are not consistent with global assessment.

The overall impact of aggregations over history and over several anomaly detection algorithms aims to eliminate singular anomalies identified only occasionally by single anomaly detection method, and thus reduce the number of false positives. On the other hand, some small-scale attacks may be missed by the system. We consider this as an acceptable trade-off, because the system is designed to provide warnings in the case of large-scale events, and not detailed protection of individual hosts.

It shall also be noted that our design helps to integrate various anomaly detection models. However, as the computational performance of the system is one of the main goals, each agent only uses the data that is relevant to its anomaly detection (and consecutively flow representation features) method. This makes the direct translation between the trust models of different types of agents largely unfeasible — trustfulness is attached to centroids in their feature spaces, and there is no guarantee whatsoever regarding the compatibility of the metrics. Furthermore, direct translation is also difficult for the agents sharing the same anomaly detection method, but working with different data (acquired by distinct probes). As the identity-context space *metrics often depends on the data* observed in the past, direct merging of the models does not provide any guarantees of consistence.

To compensate this problem, our architecture puts the interactions at the processing stages where the agents share the same language, i.e. the identity of specific network flows. When sharing the anomalies and trustfulness (which may be normalized into degree of trust [14]) values, these values are relevant to individual network flows, and each agent is able to position them in its feature space.

In the last phase of evaluation, each agent shares the trustfulness of flows with other agents. Agents then use a simple reputation mechanism to reach a collective conclusion regarding the trustfulness of observed flows, allowing to filter the traffic by trustfulness for analysis. Collectively accepted flows are then sent to analyst interface layer for further filtering based on user preferences, and possibly assisted analysis by analyst.

The decision whether a given flow is trusted or untrusted depends on the typical degree of anomaly in the observed network traffic. This parameter varies widely with network type — the number of anomalies is typically low on corporate or government networks, but is significantly higher on public Internet backbone links, or in the university settings. To avoid the problems with manual tuning of the system, we use a fuzzy-inference process integrated with the trust model to decide whether the given flow is malicious, by computing its inference with Low and High trust values. These values are determined similarly to the trustfulness of individual flow representations, but are represented as two fuzzy intervals [15].

When we look at the problem from the machine learning perspective [21], it is difficult to decide whether the trust learning as implemented in our system is supervised or unsupervised. It is clear that there is no dedicated supervisor entity, but each agent provides the inputs (i.e. anomalies) to the others, and their adaptation is mutual. On the other hand, these inputs are provided by the anomaly detection methods of individual agents, and these can be in some cases considered as supervisors for the trustfulness learning.

4 System evaluation and performance

In order to evaluate the effectiveness of our system, we have deployed one probe at the main network gateway of one of the university campuses, and we have used the standard tools to attack the hosts inside the observed network, on the background of regular network traffic. We present a selection of results that we have obtained.

In the results, we want to validate several properties of the system. First and foremost, the system shall be able to identify the introduced attack as malicious. This requires the agents to assign low trustfulness to the attack traffic, while assigning higher trustfulness to normal traffic.

While this requirement is important, there are also other properties that we want to validate. As the trust management system described in [14] and Section 3.2 effectively performs fuzzy classification of flows to the more or less trusted classes defined by the centroids in the identity-context space, we want the traffic relative to single attack to fall into the same class (or few classes). The trustfulness of these classes shall be low to comply with the previous requirement, but their low number and proximity in the identity-context space shall allow classification of attacks and signifies that the features used to define the space, the metrics using these features and the algorithm used for centroid creation are well adjusted to detect this type of attack in the current environment.

In the experimental results provided below, we illustrate the capabilities of the system by showing the data from the actual attack we have launched against our experimental platform. The attacks were performed using the `nmap` [11] tool, which is used by network administrators and attackers alike to map the network and to identify the services running on individual hosts. We

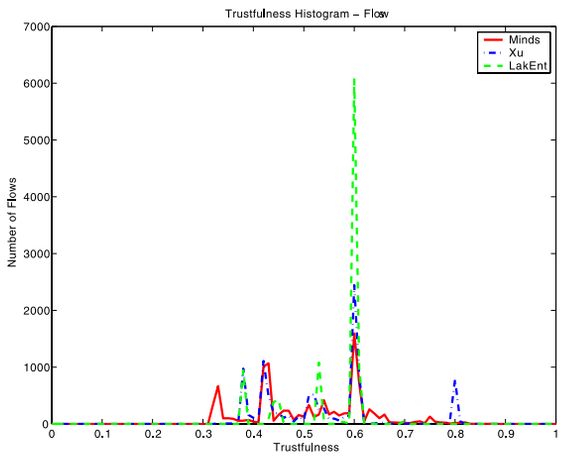


Fig. 4 Trustfulness of all flows as determined by the agents.

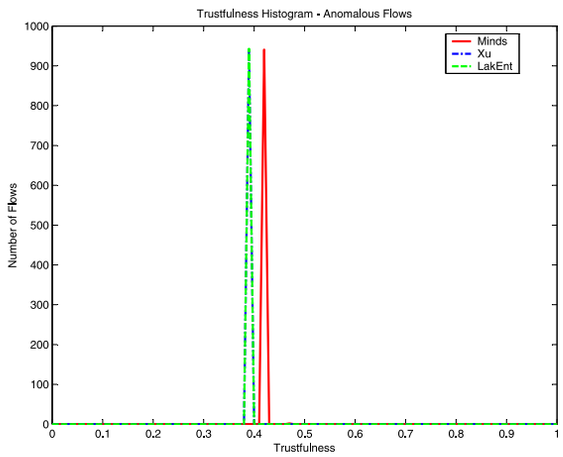


Fig. 5 Trustfulness of SYN scan attack flows as determined by the agents. Note the different scale from Fig. 4.

have performed several vertical scans of a single host, probing different ports for listening services using the SYN scan.

In Fig. 4, we can see the histogram of trustfulness of the flows for 1024 ports SYN scan as determined by detection agents [14] based on MINDS, Xu. et al. and the agent based on entropy prediction as introduced by Lakhina et al. (Lakhina Entropy). The trustfulness is displayed on $[0, 1]$ scale, 0 denotes complete distrust, while 1 corresponds to complete trust. Each of the agents maintains its trust model using the processes described in Section 3.2, and we present the distribution of the traffic on the trustfulness scale as assessed by each of them. We can note that the trustfulness of most traffic falls between 0.3 and 0.75 for the MINDS method, 0.4 and 0.6 for the Lakhina Entropy method, and 0.4 and 0.8 for the Xu method. We can clearly observe the ability of all agents to distinguish the untrusted traffic on the left side of the distribution from the trusted traffic on the right side, but we can also see that each agent provides a different shape of the histogram.

In Fig. 5, we only present the traffic corresponding to our attack. We can distinguish the peaks observed on the first graph, and we can assert that the attack we have generated was clearly identified by the method as an attack. For both Lakhina Entropy and Xu distributions, it even constitutes the lower boundary of the trustfulness of the observed flows. When we analyze the traffic contributing to the 0.3 peak on the left side of the attack peak in the MINDS distribution, we can mostly distinguish peer-to-peer networks, heavy web use (e.g. http proxies), and several clear examples of false positives, mostly due to the irregularities in the activity of observed servers. When we combine the results of the agents in the AND mode, the attack remains the only event (i.e. group of flows) which is classified with trust lower than 0.5 by all three agents. This is easily observed in Fig. 6, where we show the distribution of flows over the aggregated trustfulness. In this observation, we can observe the attack flows forming an isolated peak on the left side of the distribution, around 0.4, and that the rest of the traffic falls into the main part of the distribution. We can note the peak on the left side of the main distribution — most of the traffic in this peak corresponds to the response to scan, and the associated lower trustfulness is thus justified.

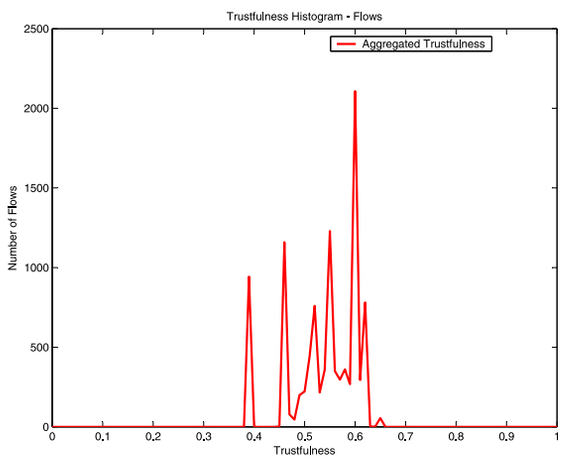


Fig. 6 Aggregated trustfulness of all flows as averaged for each flow from the values provided by the agents. Note the isolated peak on the left, which contains attack flows.

We also need to address the other criteria of quality. In the results shown above, the agents assign the flows contributing to the attack to the neighborhood of very few centroids, showing that the trust models are set to

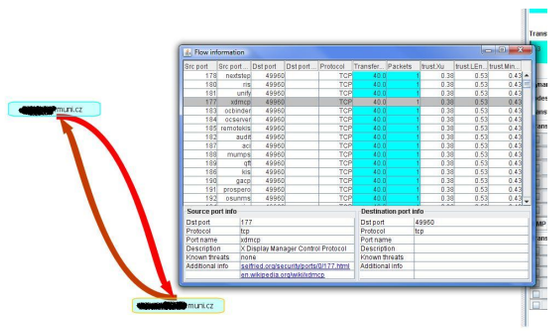


Fig. 7 Operator's view of the attack through the GUI with applied filtering.

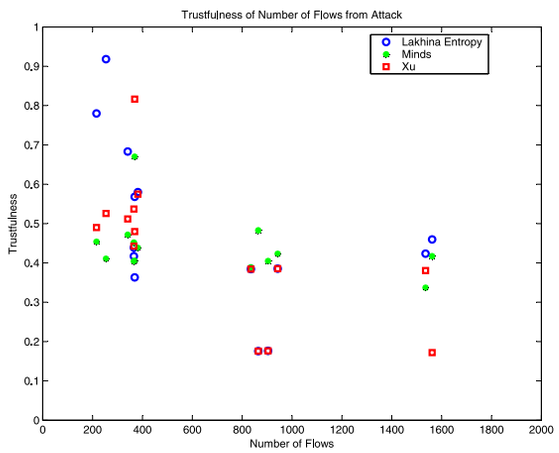


Fig. 8 Trustfulness of various types of scanning attacks as estimated by the agents as a function of the number of flows in the attack.

an appropriate level of granularity [16] which makes a trade-off between precision and efficiency. In Fig. 3, we can look into the trust model of the Lakhina Entropy agent, and we can see the attack flows attached to the single centroid (with low trustfulness) on the right front side of the image, while the rest of the traffic is attached to other centroids. The results for other agents are similar, and this consistency shows that the granularity of the trust model is well adapted to current network status.

While Fig. 3 provides an interesting insight into the reasoning process of the trusting agents, it is of limited use to network administrators that need more practical representation of data. Therefore, the system contains a dedicated visualization agent which allows the administrators to filter out the trusted traffic, as assessed by the agents, and to concentrate on the analysis of untrusted traffic. In case of the attack presented above, the result is shown in Fig. 7.

While the single attack attempt can illustrate the function of the system, it offers only a limited insight into its effectiveness. Therefore, we present a graph which illustrates the effectiveness of the system in detecting the vertical scans of with different parameters, techniques (TCP CONNECT/SYN scan, UDP scan), speed and ordering. We will show the trustfulness attributed to the events by individual agents (including the aggregation) in function of the number of flows during the acquisition period. The Fig. 8 shows that the attacks with more than several hundreds flows (typically corresponding to number of ports scanned on the target machine) are consistently discovered by all agents. The slower attacks, using lower number of flows (300 and less) are more tricky. While the aggregated trustfulness values are typically low, the higher average trustfulness attributed to these attacks means that they can blend into the normal traffic.

The results we have presented show that the use of trust models for the integration of several anomaly detection methods improves the results of network behavior analysis, and that this method can be efficiently integrated with real traffic using efficient inline NetFlow probes. While we have only presented the results relative to identification of vertical scanning, the other results show that our method is effective against other relevant threats, such as worms and other malware propagation, horizontal scanning, peer-to-peer networks and other irregularities. The evaluation of effectiveness of our method against these threats is an important topic of our current and future work, together with improvements of the method itself.

5 Conclusions and future work

The research presented in this paper uses highly abstract methods from the field of multi-agent trust modeling, and extends them to efficiently fulfill the needs of network operators: identification of significant malicious traffic events in the supervised network, without being bothered by comparatively high rate of false positives.

To achieve this goal, we can not rely on agent technologies alone. The data must be acquired from the network, preprocessed in the collector servers and transferred to the detection layer for filtering. The detection process is based on the combination of agent-specific anomaly detection techniques with extended trust modeling, that each agent uses to integrate the anomaly values from past observations, as well as anomaly assessments provided by other agents. This step reduces significantly the number of false positives, as the non-systematic anomaly peaks are diluted between normal traffic and anomaly values provided by other agents.

The results of the detection process performed inside

each agent are integrated by dedicated agents, using the reputation-like mechanism, and all trustfulness values are used during the traffic analysis by the operator. This analysis is performed using the Mycroft agent, that not only uses the trustfulness for shown traffic filtering, but also facilitates the analysis by getting additional information about the traffic.

In our future work, we have several principal goals. First, the primary goal is to further increase the system effectiveness, to improve the ratio between false positives and negatives. The performance of the system is also crucial, and while we can still improve it by rigorous analysis and software engineering, we also seek alternative ways. The most promising is the autonomous adaptation — as the detection agents are self-contained, and the system is open, we will introduce the distributed adaptation mechanism that will select the best agents, well adapted to the current traffic status.

Acknowledgement

This material is based upon work supported by the European Research Office of the US Army under Contract No. N62558-07-C-0001. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the European Research Office of the US Army. Additional support provided by Czech Ministry of Education grants 1M0567 and 6840770038. Martin Rehak would also like to acknowledge the support of the National Institute of Informatics in Tokyo during his visit.

References

[1] S. Axelsson, "The base-rate fallacy and the difficulty of intrusion detection," *ACM Trans. Inf. Syst. Secur.*, vol. 3, no. 3, pp.186–205, 2000.

[2] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," *Technical Report* 99-15, Chalmers Univ., March 2000.

[3] CESNET, z. s. p. o. Family of COMBO Cards. http://www.liberouter.org/hardware.php, 2007.

[4] Cisco Systems. Cisco IOS NetFlow. http://www.cisco.com/go/netflow, 2007.

[5] Evan Cooke, Farnam Jahanian, and Danny Mcpherson, "The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets." *Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI)*, pp.39–44, June 2005.

[6] L. Ertoz, E. Eilertson, A. Lazarevic, P.-N. Tan, V. Kumar, J. Srivastava, and P. Dokas, "MINDS - Minnesota Intrusion Detection System." *Next Generation Data Mining*, MIT Press, 2004.

[7] A. Lakhina, M. Crovella, and C. Diot, "Characterization of Network-Wide Anomalies in Traffic Flows." *ACM SIGCOMM conference on Internet measurement IMC '04*, pp.201–206, New York, NY, USA, ACM Press, 2004.

[8] A. Lakhina, M. Crovella, and C. Diot, "Diagnosis Network-Wide Traffic Anomalies." *ACM SIGCOMM '04*, pp.219–230, New York, NY, USA, ACM Press, 2004.

[9] A. Lakhina, M. Crovella, and C. Diot, "Mining Anomalies using Traffic Feature Distributions." *ACM SIGCOMM, Philadelphia, PA, August 2005*, pp.217–228, New York, NY, USA, ACM Press, 2005.

[10] M. Lesk, "The new front line: Estonia under cyberassault." *IEEE Security and Privacy*, vol. 5, no. 4, pp.76–79, 2007.

[11] Gordon Lyon, Nmap. http://insecure.org/nmap/.

[12] S. Marsh, Formalising trust as a computational concept, 1994.

[13] S. Northcutt and J. Novak, *Network Intrusion Detection: An Analyst's Handbook*. Thousand Oaks, CA, USA, New Riders Publishing, 2002.

[14] M. Rehak, M. Pechoucek, K. Bartos, M. Grill, and P. Celeda, "Network intrusion detection by means of community of trusting agents." *IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2007 Main Conference Proceedings) (IAT'07)*, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[15] M. Rehák, L. Foltýn, M. Pěchouček, and P. Benda. "Trust Model for Open Ubiquitous Agent Systems." *Intelligent Agent Technology, 2005 IEEE/WIC/ACM International Conference*, number PR2416 in IEEE, 2005.

[16] M. Rehak and M. Pechoucek. "Trust modeling with context representation and generalized identities." *Cooperative Information Agents XI*, number 4676 in LNAI/LNCS. Springer-Verlag, 2007.

[17] K. Scarfone and P. Mell, "Guide to intrusion detection and prevention systems (idps)." *Technical Report 800-94*, NIST, US Dept. of Commerce, 2007.

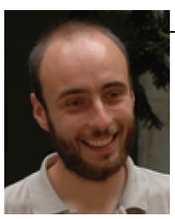
[18] S. Sidiroglou and A. D. Keromytis. "Countering network worms through automatic patch generation." *IEEE Security & Privacy*, vol. 3, no. 6, pp.41–49, November/December 2005.

[19] D. Šišlák, M. Rehák, M. Pěchouček, M. Rollo, and D. Pavlíček, "A-globe: Agent development platform with inaccessibility and mobility support." *Software Agent-Based Applications, Platforms and Development Kits*, pp.21–46, Berlin, Birkhauser Verlag, 2005.

[20] Sourcefire, Inc. SNORT – Intrusion Prevention System. http://www.snort.org/, 2007.

[21] J. Tožička, M. Rovatsos, and M. Pěchouček, "A framework for agent-based distributed machine learning and data mining." *Autonomous Agents and Multi-Agent Systems (AAMAS 2007)*, pp.666–673, New York, NY, ACM Press, 2007.

[22] K. Xu, Z.-L. Zhang, and S. Bhattacharrya, "Reducing Unwanted Traffic in a Backbone Network." *USENIX Workshop on Steps to Reduce Unwanted Traffic in the Internet (SRUTI)*, Boston, MA, July 2005.



Martin REHÁK

Martin REHÁK works as a researcher at the Agent Technology Group, Gerstner Laboratory, Czech Technical University. He holds engineering degree from Ecole Centrale Paris. His current research interests are centered on trust, distributed system security, intrusion detection and multi-agent task allocation.

Dr. Michal PĚCHOUČEK

Dr. Michal PĚCHOUČEK works as a reader in Artificial Intelligence and the Head of Agent Technology Group at the Department of Cybernetics, CTU. He holds degrees from FEE-CTU and University of Edinburgh and completed his Ph.D. in AI at CTU in Prague. His research focuses on multi-agent systems, especially topics related to social knowledge, security, communication inaccessibility, coalition formation, agent reflection and multi-agent planning.

Pavel ČELEDA

Pavel ČELEDA works as a researcher at the Institute of Computer Science at the Masaryk University, Brno. He has a engineering degree in electrical engineering from the Military Academy Brno and Ph.D. in informatics from University of Defence. His current research areas include monitoring of high-speed networks and development of network probes.

Vojtěch KRMÍČEK

Vojtěch KRMÍČEK is a Ph.D. student at the Faculty of Informatics, Masaryk University, Brno, where he got his master degree. His current research activities are focused on high-speed networks and network security. He is a member of the software group in Liberouter project, developing high-speed network monitoring devices.