Adaptive Multiagent System for Network Traffic Monitoring

Martin Rehák, Michal Pěchouček, Martin Grill, and Jan Stiborek,
Czech Technical University

Karel Bartoš, *Czech Education and Scientific Network (CESNet)*

Pavel Čeleda, *Masaryk University*

Individual anomaly-detection methods for monitoring computer network traffic have relatively high error rates. An agent-based trust-modeling system fuses anomaly data and progressively improves classification to achieve acceptable error rates.

Network behavior analysis (NBA) is an intrusion-detection technique that uses the patterns in network-traffic structures and properties to identify possible attacks and technical problems with minimal impact on user data privacy. The analysis is not based on content of the transferred information but on network traffic statistics in the NetFlow/IPFIX (IP Flow Information Export) format. The statistical data comes from either network routers or dedicated inline devices.

NetFlow/IPFIX structures data in flow records. Each record describes one *flow*—that is, a unidirectional component of a TCP connection (or its UDP or ICMP equivalent) that contains all packets with the same source-IP address, destination-IP address, source and destination ports, and transport protocol (TCP/UDP/ICMP). A flow record contains this basic information, together with the number of packets/bytes transferred, the flow duration, and the TCP flags encountered in the flow packets. NetFlow probes aggregate the flow records over a predefined observation period (typically 5 minutes). When the observation period elapses, the data is read out for analysis, and a new observation period begins.

Detecting attacks in flow data sets is typically based on the anomaly detection (AD) paradigm (see the sidebar, "Intrusion Detection Techniques"). AD methods build a traffic model and compare the model's predicted results with the actual traffic observed. The differences are assumed to result from an attack or technical problems, and they're reported to system administrators for further analysis. The principal parameter of any AD method is the classification error. *False negatives* denote malicious flows that the system wrongly labels as legitimate; *false positives* denote legitimate flows labeled as malicious. Current NBA methods suffer from a high rate of false positives and, consequently, a high volume of false alarms. This restricts possibilities for using classical statistical, machine learning, and AI methods in NBA systems.

Furthermore, scalable deployment of a robust NBA system requires fully distributed operation and localized data analysis. Remote components must communicate in a way that lets the system as a whole efficiently analyze global network status without fully centralized data analysis processes. Finally, because the network environment is a complex dynamic system with properties

Intrusion Detection Techniques

Network behavior analysis (NBA) is only one of the principal approaches to intrusion detection. Intrusion detection systems are classified by their location (on a host, a wired network, or a wireless network), detection methodology (signature matching, anomaly detection, or stateful protocol analysis), or capability (simple detection or active attack prevention).[1] The most relevant techniques with respect to NBA are those based on signature matching and stateful protocol analysis.

Signature matching techniques try to identify attacks by comparing the content of network packets with a set of rules that describe known attack signatures. However, these techniques are increasingly unreliable because of the growing proportion of ciphered traffic and the use of self-modifying malware or other evasion techniques.

Stateful protocol analysis matches each connection with an established baseline profile for a given protocol and reports any deviations from this profile. This technique is effective against certain classes of attacks, such as horizontal network scanning or host behavior profiling. However, it's ineffective against attacks that conform with the normal protocol behavior.

In general, neither technique is sufficient on its own. Most well-managed networks feature a combined signature matching/stateful protocol analysis appliance for intrusion detection and integrate it with a network firewall. They rely on NBA techniques to detect malicious behavior within the protected network.

References

1. K. Scarfone and P. Mell, *Guide to Intrusion Detection and Prevention Systems* (IDPS), tech. report 800-94, Nat'l Inst. Standards and Technology , US Dept. of Commerce, 2007.

that change frequently, manual NBA system configuration is impractical. Therefore, self-configuration and self-adaptation capabilities are also a crucial requirement.

Agent-based data mining is a suitable approach for addressing these issues.[1] In this article, we introduce Camnep, an agent-based framework that improves the classification precision of NBA techniques and supports their use for intrusion detection in high-speed backbone networks. The Czech Technical University's Agent Technology Center developed the Camnep framework in collaboration with Masaryk University; the Czech Education and Scientific Research Network (CESNet), operator of the Czech Republic's National Research and Education Network and the US Army Communications-Electronics Research, Development, and Engineering Center (CERDEC).

System Architecture and Requirements

The Camnep system architecture is layered to cope with the vast amounts of data in its lower, highly efficient layers, while making progressively more sophisticated decisions in the upper layers.

The high data volumes produced by individual network probes impose the need to decentralize the data processing. Therefore, as the system operational flow in Figure 1 shows, local agents process the data from each probe and escalate only the identified cases of malicious or dubious behavior. This structure avoids the computational and communication cost associated with the centralized processing of the complete traffic data.

FlowMon probes (www.liberouter.org) handle the traffic-acquisition layer of the system architecture. These dedicated FPGA-based inline probes gather NetFlow/IPFIX-formatted information about the passing traffic. This information is also available from general-purpose network equipment, such as routers, but the data quality tends to suffer under high traffic loads or attack, when these devices give priority **to** their primary function rather than to monitoring and aggregating the NetFlow data set. The Camnep system handles considerable data volume, and it can process millions of flow records in each 5-minute observation period on gigabit network loads. The probe progressively builds this data from observations of the individual packets passing through the link and periodically sends this data into a *collector*. This single-purpose database uses a modified version of the existing Nfdump (nfdump.sourceforge.net) and

NfSen (nfsen.sourceforge.net) open source tools to aggregate the flow data and provide it to the local agent *container*, where a set of agents performs the attack detection.

Camnep implements the AD layer functionality as a multistage, distributed process based on interactions between local high-performance detection agents and aggregation agents, supported by the incident agents.

- *Detection agents* process all flows from the local probe and use their anomaly and trust models to assign trustworthiness, represented as a value in the [0,1] interval, to all flows in the current set. This value is obtained as an output of the agent's trust model, which aggregates and generalizes the outputs of the AD methods of all the container's detection agents. The set of detection agents can vary between the containers, to reflect the performance of the individual methods in various environments.
- *Aggregation agents* integrate the opinions of all local detection agents regarding a flow's trustworthiness—that is, each agent's estimate of the flow's legitimacy. Each aggregation agent embodies one or more averaging functions for building a joint trustworthiness value,
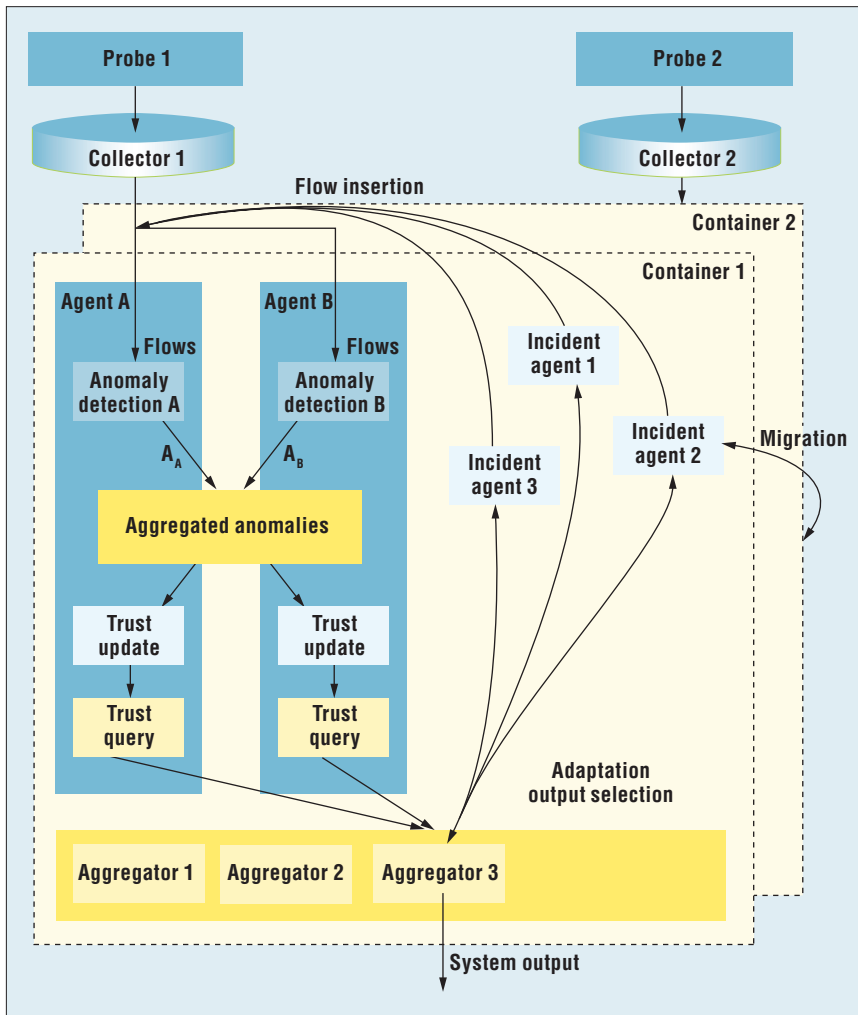
Figure 1. The Camnep system operational flow. Details inside a single container show detection agents A and B determining the anomalies and trustworthiness of individual flows. An aggregation agent provides the final result, which is selected from the pool of aggregation agents by incident agents that probe the system's response to known problematic attacks and known false positive cases.

and they compete with each other to provide the best overall result—that is, the best combination of the individual agents' opinions.

- *Incident agents* reflect a comprehensive representation of the system's knowledge about one specific incident. These agents have a twofold role: as an input for a local adaptation process of individual detection containers and, because of their mobility, as a means of knowledge transfer between the containers.

The system distributes the agents in container sets that represent an abstrac-

tion of runtime locations for agents. The containers make the physical-logical distribution transparent for the agents by providing a unified system structure and interface for agents running within the same process, same host, or physically distinct hosts. All detection and aggregation agents in each container process the identical set of data, using the real-time attack detection algorithm we describe in the next section. The incident agents are self-contained, and they can autonomously migrate between containers. The architecture doesn't include the operator interface, visualization and

alert fusion, or management components. The IDS research community has already addressed these matters, and we have designed the Camnep system to be a functional component of a larger network-monitoring solution.

Real-Time Attack Detection Algorithm

The main Camnep system functionality is autonomous traffic classification. To perform the classification, detection agents combine the results of AD techniques by using trust modeling.[2] Each detection agent is based on one existing AD technique. These techniques extract a subset of method-specific features from the traffic set and compare the observed feature values with the values predicted from past traffic. The system then designates as anomalous any observed traffic that's inconsistent with the prediction. The detection agents integrate the anomalies provided by their own AD method and the AD methods of other detection agents for the current set of flows with past AD results already represented by the trust model state.

These operations constitute the first two stages of agent operations: anomaly detection and trust update. The third stage is trust aggregation, when the agents build a collective opinion about the maliciousness of the flows observed by the local platform.

Before we describe the AD methods integrated in the system and the detection process in general, we define the terms and techniques used in our approach.

We adopt Stephen Marsh's description of x's *trust* in y as "x expects that y will behave according to x's best interests and will not attempt to harm x."[3] In the network security domain, trustworthiness estimates the system's belief that a given flow is benign from the destination host's point of view. This belief is based on the output from AD methods that are designed principally to detect anomalies correspond-

ing to known broad classes of suspicious behavior, while not losing their generalization capabilities. The system determines the trustworthiness in the [0,1] interval, where 0 corresponds to complete distrust and 1 to complete trust.

The features observable directly on each flow define its *identity*: source IP address, destination IP address, source port, destination port, transport protocol, number of bytes, and number of packets. The features observed on the other flows in the same data set define each flow's *context*—for example, the number of similar flows from the same source IP or the entropy of the destination IPs of all requests from the same host as the evaluated flow. All Camnep detection agents use the same flow identity representation, but the context is defined by the features that the agents' respective AD methods use to draw conclusions regarding the flow anomaly. The trust modeling algorithm uses the identity and context to define the *feature space*, a metric space on which each agent's trust model operates.[2] A metric space describes the similarity between the flow identities and contexts, and it is specific to each agent.

Anomaly Detection Methods
Our system's AD methods are based on existing intrusion-detection approaches. We've integrated these approaches in the two stages described earlier:

- extracting meaningful features associated with each flow (or group of flows), and
- using these feature values to determine whether the flow is anomalous or not.

The features fall entirely in the context category—that is, they are observed on other flows in the current data set, rather than on the flow itself. This allows the methods embedded in the

system agents to detect attacks with a structure similar to the real traffic, such as denial-of-service attacks with high volumes of correctly formatted requests. The system's baseline configuration, which we use in the evaluation reported later in the section "Algorithm Characteristics," contains five detection agents, each based on a distinct detection algorithm:

- The Minnesota Intrusion Detection System (Minds) builds the con-

> The trust modeling algorithm uses a flow's identity and context to define a feature space that's specific to each anomaly detection agent.

text information for each flow using the number of flows from the same source IP as the evaluated flow, number of flows toward the same destination host, number of flows toward the same destination from the same source port, and number of flows from the same source toward the same destination port.[4]
- The algorithm developed by Kuai Xu and colleagues actually classifies the traffic sources, which imposes the same context for all flows from the same source IP.[5] For each source, we determine the normalized entropy of the set of source ports, destination ports, and destination IPs of all the flows from this source. Anomalies are determined by applying static classification rules.
- The volume prediction algorithm uses Principal Components Analy-

sis (PCA) to build the traffic volume model from individual sources.[6] We model three values for each source IP with a nonnegligible volume of originating traffic: number of flows, number of bytes, and number of packets from the source. We use the PCA method to identify the relationships between the traffic from distinct sources.
- The entropy prediction algorithm is based on the similar PCA-based traffic model, but uses different features from those for volume prediction.[6] It aggregates the traffic from the individual source IPs but, instead of traffic volumes, predicts the entropies of destination IPs, destination ports, and source ports over the set of flows from each source.
- The Time Access Pattern Scheme (TAPS) method differs from the preceding approaches by targeting a specific class of attacks: the horizontal and vertical scans.[7] The TAPS algorithm classifies a subset of suspicious traffic sources, characterized by three features: number of destination IP addresses in the set of flows from the source, number of ports in the same set, and entropy of the flow size measured in number of packets. The anomaly of the source is based on the ratios between the values.

Local Detection Algorithm
These five AD methods are integrated into the detection agents and used in our algorithm in three phases:

- *Anomaly detection.* The detection agents use the embedded AD method to determine each flow's anomaly as a value in the [0,1] interval, where 1 represents the maximal anomaly and 0 represents no anomaly at all. The anomaly values are shared with the other detection agents within the container, averaged, and used as an input in the second processing phase. All the detection agents in the
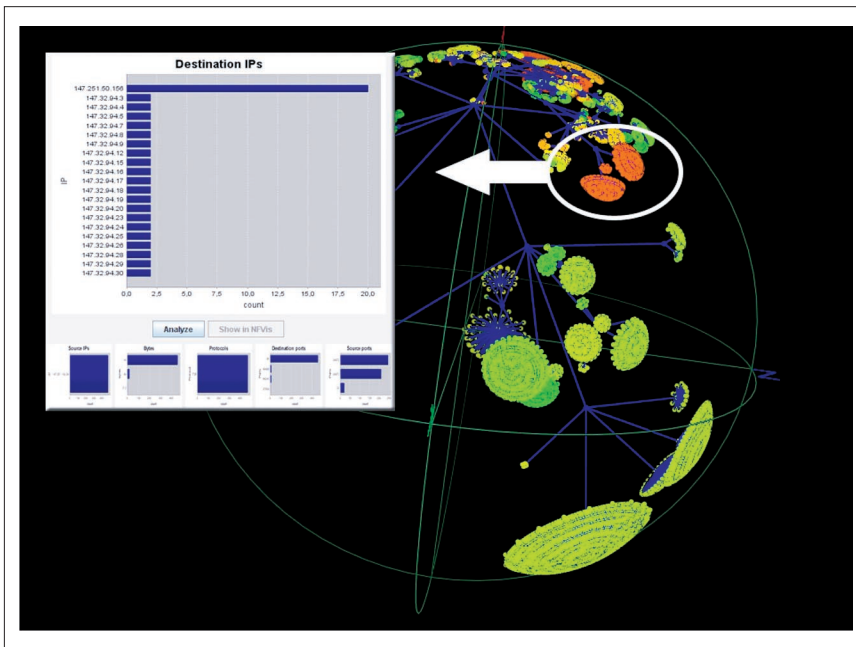
Figure 2. Flows corresponding to one attack event (TCP horizontal scan) highlighted in the trust model of one of the agents, projected into 3D space. Each flow is displayed as a colored dot, attached to the nearest centroid. The color corresponds to the centroid's trustworthiness.

container use the same anomaly values as an input for trust modeling.

- *Trust update.* The detection agents integrate the anomaly values of individual flows into their trust models. Reasoning about each individual flow's trustworthiness is both computationally infeasible and impractical (the flows are single-shot events by definition), so the trust model holds the trustworthiness of significant flow samples (for example, centroids of fuzzy clusters) in the identity-context feature space and uses each flow's anomaly to update the trustworthiness of centroids in its vicinity. The weight used for updating the centroid's trustworthiness with the flow's anomaly values decreases with the distance from the centroid. Therefore, as each agent uses a distinct distance function, each agent has a different insight into the problem, as shown in the example in Figure 2. The flows are clustered according to the different criteria, and the cross-correlation implemented by sharing the anomaly values used to update the trustworthiness helps

eliminate random anomalies.

- *Trust aggregation.* Each detection agent determines the trustworthiness of each flow (with an optional normalization step), and all agents provide their trustworthiness assessment (conceptually, a reputation opinion) to the aggregation agents. Each aggregation agent then builds one possible system output by combining the detection agent's trustworthiness opinions using its own aggregation function, which is based on identity-weighted or order-weighted averaging (OWA). Identity-weighted averaging is fairly straightforward. It assigns weights to the trustworthiness provided by the system's individual agents and averages the trustworthiness values according to these weights. OWA is different because the weights aren't assigned to the individual agents. They are assigned instead to the order of trustworthiness values provided by the individual detection agents. The mechanism can thus emphasize the lowest or highest trustworthiness values or use

only the middle values from the distribution.

The main problem of the trust aggregation stage is identifying the optimal trustworthiness aggregation function for building a consensual trustworthiness opinion from the partial trustworthiness opinions of the individual agents. Finding the optimal aggregation function is difficult. We typically don't have any information regarding the quality of the individual agents' conclusions, and their performance can change dramatically with variations in traffic and attack characteristics. This fact—together with the lack of representative, network-specific training samples—prevents the use of more traditional ensemble methods. The performance implications of a suboptimal choice can be important.

Figure 3 shows histograms of the same data set aggregated by two different operators. A simple average of trustworthiness values is clearly not an optimal choice because it assigns trustworthiness values that place incidents quite close to the legitimate (but unstable) traffic sources in the trustworthiness histogram (Figure 3a). The use of a more appropriate OWA function achieves a better separation (Figure 3b).

While the trustworthiness assignment to the individual flows provides a traffic ordering, the trustworthiness by itself does not perfectly separate the normal and malicious traffic. The thresholds applied on the aggregated trustworthiness values split the traffic that the individual nodes process into three classes, as shown in Figure 4:

- Normal traffic consists of flows with high trustworthiness, which the system considers to be legitimate. This class should contain most of the traffic volume and is not included in later processing stages.
- Malicious traffic consists of flows

Figure 3. Flows aggregated by trustworthiness with two different aggregation order-weighted-averaging operators. (a) arithmetic average and (b) average of the two highest values (out of four). The buffer overflow attempt (red) is separated from the rest of the traffic by both methods, but the permanently active threats of P2P botnets (orange and yellow) are better separated from the first legitimate active host (green) in (b).

(a)

(b)

with low trustworthiness. The system reports these flows (grouped into incidents) to the user and propagates the selected incidents to other parts of the system for adaptation, as we describe in the next subsection.

- Unclassified traffic consists of flows with trustworthiness in the middle of the spectrum, between the two thresholds. Correct classification of the incidents from this class, in which the trustworthiness by itself cannot separate the legitimate and malicious traffic, is a key factor in achieving a low error rate of the system. The sources in this category are therefore typically subject to further processing.

We can now describe the procedure that selects the best aggregation agent and automatically identifies the optimal threshold values.
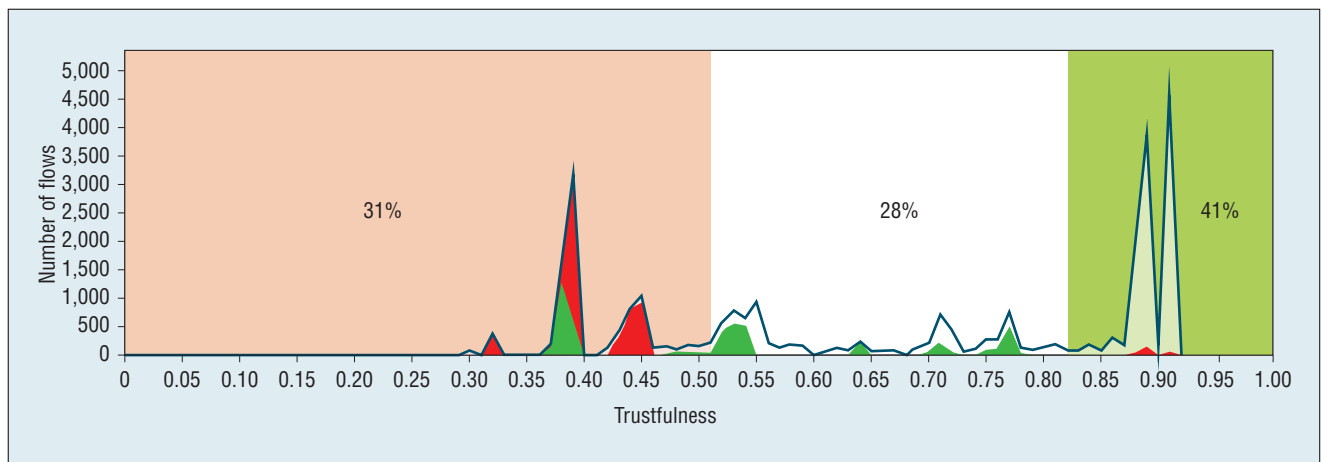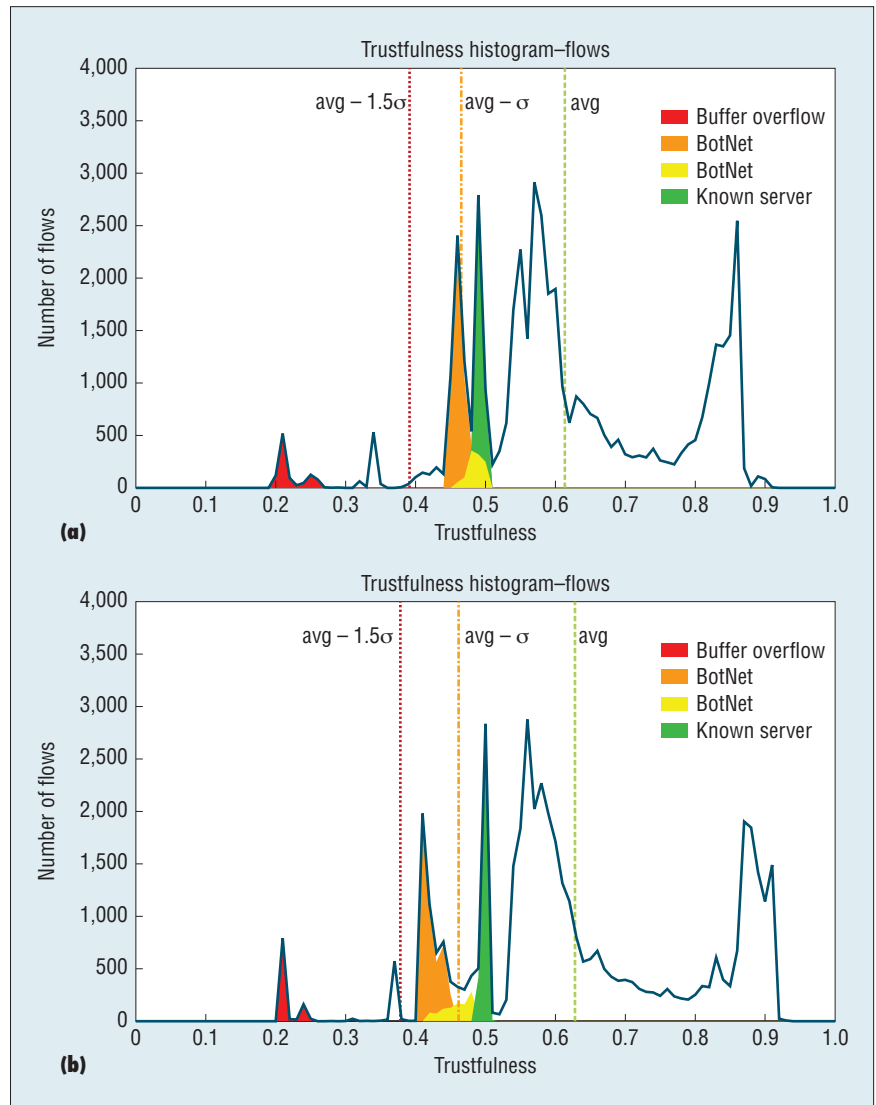

Figure 4. Traffic histogram with the traffic divided into three classes: malicious (light orange background, left), unclassified (white background, middle) and normal (light green background, right). The solid colors over the selected histogram portions reflect the ground truth as identified manually.

Local System Adaptation

We based the Camnep system architecture on sophisticated, dynamically self-configured aggregation of hypotheses provided by the individual detection agents. Although the adaptation mechanism does not directly influence the individual detection agents, it identifies the optimal aggregation function to reflect the actual network traffic environment.

The adaptation aims principally to select the optimal aggregation agent, whose opinion the system will then use as an output. The selection is guided by the incident agents, which represent system knowledge about specific known attacks or legitimate traffic types. Each incident agent contains a description of a particular incident, extracted characteristics of the event, the incident classification (legitimate or malicious traffic), and the full record (or a representative sample) of the flows that constituted the incident.

A container's incident agents inject the flow records they contain into the traffic captured by the system before the AD stage. The injected traffic is then processed together with the observed traffic, and the incident agents receive the final trust estimates as provided by the container's aggregation agents. Each incident agent then ranks the aggregation agents according to the trustworthiness that the aggregation agent attributed to the flows inserted by the individual incident agent. The incident agents seek to maximize the trustworthiness of the false positives (legitimate traffic) or minimize the trustworthiness of the true positives. They vote to determine the best aggregation agents, and the system selects these results as an output.

Besides selecting the best aggregation function, the same mechanism determines the threshold values that separate malicious and legitimate traffic. Each aggregation agent uses the feedback from the incident agents to update the threshold positions. Upon processing of each data set, the aggregation agent determines two values:

- the lowest trustworthiness assigned to the traffic inserted by the incident agent representing a false positive, and
- the highest trustworthiness assigned to the traffic inserted by the incident agent representing a true positive.

These values are averaged with the corresponding values obtained on the pre-

> Selection of the optimal aggregation agent is guided by incident agents, which represent system knowledge about specific known attacks.

vious data sets. The results define the boundaries between malicious and unclassified traffic and between unclassified and normal traffic, respectively.

This adaptation technique addresses the major problem of autonomic IDS adaptation—that is, the lack of training data relevant to the current network traffic. It's based on the assumption that while the traffic is highly dynamic, the typical attack or false positive patterns as embedded in the incident agents are relatively transferable between networks with various traffic characteristics. However, the training data set (defined by the incident agent's traffic samples) remains extremely sparse when compared to typical traffic volumes. Furthermore, it ignores the network traffic patterns typical for the network where the probe is deployed. We argue that our

approach remains effective even with incomplete and small training data sets because the system learns only how to merge the results from the ensemble of detection agents. Each detection agent embodies a large amount of domain knowledge in its AD and trust modeling algorithms. This reduces the learning problem's scope to efficient algorithm combinations in the given environment, which can be successfully performed with a small amount of training data inserted by incident agents into the unclassified background traffic.

Distributed Adaptation and Knowledge Transfer

The primary role of Camnep incident agents is to represent knowledge about specific incidents. The description of one incident includes the complete flow record (which can be sampled in cases of extremely large incidents); extracted properties of the incident and the trustworthiness values attributed to it, including the identification of the detection and aggregation agents providing these values; and the background data set. The Camnep system uses incident agents to accomplish two principal tasks: local system self-adaptation, as described earlier, and knowledge transfer between the containers attached to different probes, so that the system behavior is consistent across the distributed monitoring infrastructure.

The incident agents can migrate between containers and thus make the adaptation process distributed. The individual containers dynamically create new incident agents from recent important events, and these agents then transfer the information about the event to other system containers. This transition helps the whole system adapt to new threat types as soon as the agents in one of the containers recognize the threat. On the downside, a knowledgeable attacker might exploit this approach to overemphasize the system response to a single attack
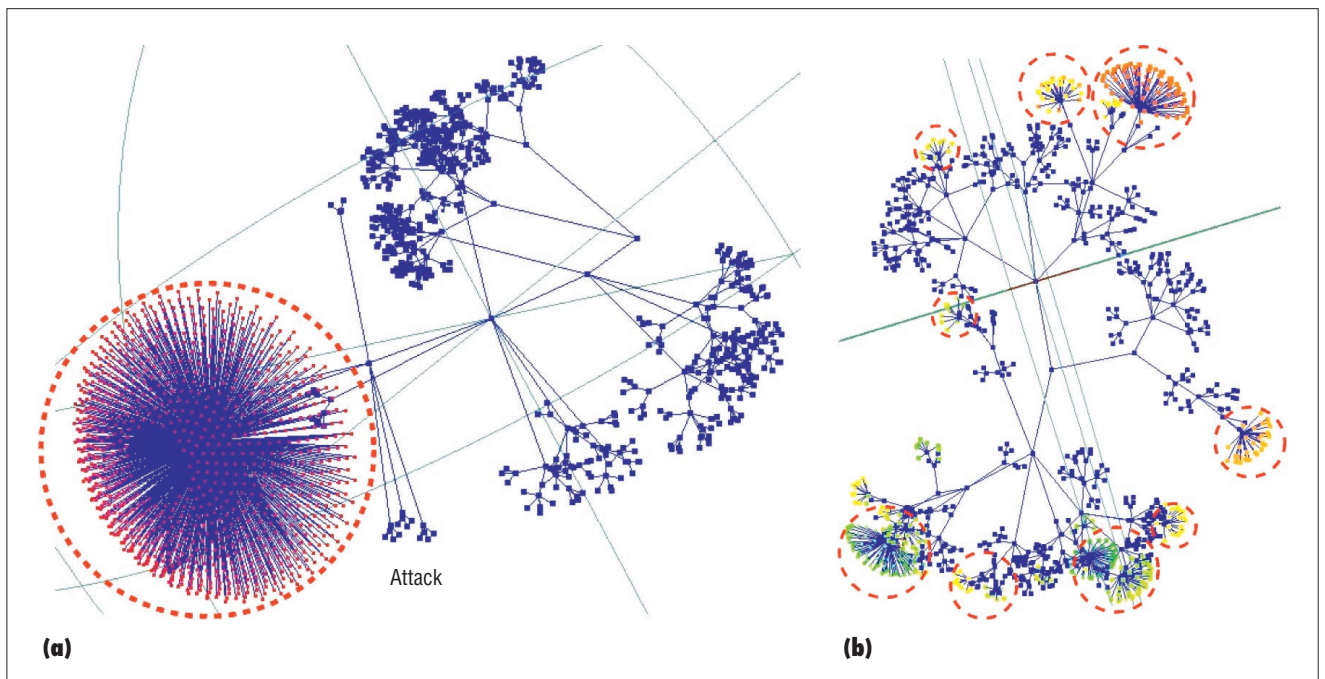
Figure 5. A visualization of a detection agent's trust model. The flows are displayed as tree extremities attached to the closest centroid of the trust model. (a) The attack flows are concentrated next to a single centroid, while (b) the normal traffic from legitimate sources is spread over the whole model.

technique, thus making other techniques less detectable. We address this threat by imposing diversity requirements on the incident agents, ensuring that all major attack techniques and corresponding false positives are represented.

The distributed adaptation mechanism based on incident agents is conceptually similar to the massively parallel approaches from the artificial immune systems (AIS) field. However, compared to AIS approaches, we employ the deliberative incident agents that are designed to optimize their operations using information about the environment provided by the platform and other agents.

Algorithm Characteristics

The attack detection algorithm we've presented aims to reduce the rate of false positives without increasing the rate of false negatives within the performance constraints imposed by the system's real-time nature. The filtering process is multistage, with each stage improving the results' quality.

Our experiments suggest that a sim-ple *average of anomaly values*, which is a baseline technique known from the ensemble classification field, reduces the error rate considerably.[8] The improvement factor depends on the traffic and classifier characteristics, but the false positive rate can decrease by a factor of 3–5 when compared with a typical AD method performance.[9] In the second stage, the use of trust modeling improves results by another factor of 3–4, bringing the total to roughly the same number of false positives and true positives in the result set.

The trust-based algorithm improves the classification quality by the parallel trust modeling that the detection agents perform and the adaptive integration of their outputs into the final trustworthiness assessment. The trust modeling inputs are identical for all agents, and the principal improvement comes from the different traffic features that each agent uses to define its feature space and metrics. Because of these differences, each trust model uses its own criteria to perform the data aggregation. The features of each

model are derived from the features that separate the malicious traffic from the legitimate traffic (because they are defined by an existing AD method). Therefore, the attack flows identified by any individual agent are typically concentrated in a rather limited part of the feature space, covered by few centroids. If a group of flows (such as the flows from the single source) falls into the untrusted region in a feature space of all agents, the trustworthiness of the centroids in this region will be consistently low because it was accumulated from highly anomalous traffic (Figure 5a). On the other hand, if only one agent or a few identify the traffic as malicious, the trust modeling algorithm will likely disperse the flows over many centroids in the trust models of other agents, and the dispersed flows will have higher trustworthiness (Figure 5b). This hypothesis, which explains how the trust modeling stage improves the false positives rate, was verified empirically during our experiments. When an agent identifies a particular event as a false positive, the flows that constitute the event are

Table 1. Error rates at various system layers with true positives (a complement of false negatives) emphasizing the ratio of false and valid alarms.

Layer	False Positives	True Positives
Individual anomaly detection algorithm	300	2
Average of anomalies	58	2
Arithmetic average of trustworthiness	15	2
Best ordered-weighted-average aggregation of trustworthiness	5	2

spread over a relatively high number (more than 20) centroid-defined clusters, while the attack flows are normally associated with fewer than five centroids.

The adaptive selection of the optimal aggregation agent reflects a task's dynamic characteristics and enables the system to significantly improve the classification quality. For example, most AD methods returned more than 300 events (counted as distinct traffic sources), of which two were actual attacks and the rest false positives (see Table 1). The average of anomaly values (for example, as in the work of Giorgio Giacinto and his colleagues[9]) reduced the number of events to about 60, with two attacks still present. The use of trust modeling further improved the results, reducing the number of incidents to 17, with 15 false positives and 2 true positives returned by the simple average of trustworthiness values provided by the detection agents. With the use of automatically identified OWA operators, the system managed to reduce the number of events to seven, while still correctly identifying both attacks. These results were obtained on the Internet connection of one of the university campuses, and the attacks in question were suspected botnet coordination nodes. The results were obtained over a longer time period and recalculated to the scale of one 5-minute interval.

We obtained these results in settings where we prohibited the use of the unclassified category in the results—the agents had to label each flow as either normal or malicious. This is a typical setting for isolated deployments. However, distributed systems can further improve results by using the unclassified category and delegating its event classifications to more computationally intensive techniques, such as signature detection. This technique achieves more reliable results in the normal and malicious categories. When we set the system to return about 1 percent false negatives in the normal traffic category (in terms of raw flow counts, not aggregated events or sources), we typically obtain about 40 percent false positives in the malicious category. We can attribute the false positives mostly to secondary effects of the attacks or to a low number of well-known hosts (we manually identified the remaining 60 percent of traffic as probable attacks) and 70 percent of false positives in the unclassified data, which also contain about 5 percent of attack traffic, with the rest of the unclassified traffic resisting reliable categorization.

W e deployed the Camnep system as part of an experimental campuswide IDS. We also tested its performance on major Internet backbones. The main practical deployment concern is related to the sheer volume of data the system must process. When operating on a 10-gigabit network link loaded to 10 percent capacity, the system must process roughly one million flows during each 5-minute observation period (3,800 flows per second), aggregated from about 125,000 packets per second. Given that university network traffic contains about 10 percent malicious traffic, the system detects hundreds of events during each

5-minute interval, about half of them being true positives. The ratio of 50 percent of false positives is commonly accepted as the minimal required performance for a deployed IDS solution. The detection layer, represented by one container with 6 detection agents and 30 aggregation agents, can process this data volume when running on a single quad-core PC.

We achieved these results by carefully merging applicable agent techniques, such as trust modeling and migration, with appropriate approaches from intrusion detection and ensemble classification. The results suggest that agent approaches can successfully complement more traditional data discovery and mining approaches. In our case, agents provide the integration infrastructure (Aglobe platform, http://agents.felk.cvut.cz/aglobe), data fusion (trust modeling), adaptation, and knowledge transfer. ◼

References

1. S. Zhong, T.M. Khoshgoftaar, and N. Seliya, "Analyzing Software Measurement Data with Clustering Techniques," *IEEE Intelligent Systems*, vol. 19, no. 2, 2004, pp. 20–27.

2. M. Rehák and M. Pěchouček, "Trust Modeling with Context Representation and Generalized Identities," *Cooperative Information Agents XI*, LNCS 4676, Springer, 2007, pp. 298–312.

3. S. Marsh, *Formalising Trust as a Computational Concept*, doctoral dissertation, Dept. of Mathematics and Computer Science, Univ. of Stirling, 1994.

4. L. Ertoz et al., "MINDS—Minnesota Intrusion Detection System," *Next Generation Data Mining*, MIT Press, 2004.

5. K. Xu, Z.L. Zhang, and S. Bhattacharrya, "Reducing Unwanted Traffic in a Backbone Network," *Proc. Usenix Workshop on Steps to Reduce Unwanted Traffic in the Internet* (SRUTI 05), Usenix Assn., 2005.

6. A. Lakhina, M. Crovella, and C. Diot, "Mining Anomalies Using Traffic Feature Distributions," *Proc. ACM SIGCOMM*, ACM Press, 2005, pp. 217–228.

7. A. Sridharan and T. Ye, "Tracking Port Scanners on the IP Backbone," *Proc. 2007 Workshop on Large Scale Attack Defense* (LSAD 07), ACM Press, 2007, pp. 137–144.

8. M. Rehák et al., "Trust-Based Classifier Combination for Network Anomaly Detection," *Cooperative Information Agents XII*, LNCS 5180, Springer, 2008, pp. 116–130.

9. G. Giacinto et al., "Intrusion Detection in Computer Networks by a Modular Ensemble of One-Class Classifiers," *Information Fusion*, vol. 9, no. 1, 2008, pp. 69–82.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.

THE AUTHORS

Martin Rehák is a senior researcher with the Czech Technical University Department of Cybernetics' Agent Technology Center, where he leads the network security activities. His research interests include network intrusion detection, trust modeling, and task allocation in adversarial environments. Rehák has a PhD in AI and biocybernetics from the Czech Technical University in Prague. He is a cofounder of Cognitive Security, a spinoff company introducing the Camnep system on the network monitoring market. Contact him at rehak@agents.felk.cvut.cz.

Michal Pěchouček Martin Pěchouček is head of the Agent Technology Center and vice-chair of the Department of Cybernetics, both at the Czech Technical University. His research focuses on problems related to multiagent systems, especially social knowledge, machine learning, communication, and planning. Pěchouček has a PhD in AI and biocybernetics from the Czech Technical University in Prague. He is a cofounder of Cognitive Security. Contact him at pechoucek@agents.felk.cvut.cz.

Martin Grill is a member of the Czech Technical University Department of Cybernetics' Agent Technology Center, a researcher at CESNet, and a master's degree student in software development with the Czech Technical University's Faculty of Nuclear Sciences and Physical Engineering. His current research focuses on network security and collective adaptation in multiagent systems. Grill has a bachelor's degree in computer science from the Czech Technical University Faculty of Nuclear Sciences and Physical Engineering. Contact him at grill@agents.felk.cvut.cz.

Jan Stiborek is a researcher with the Czech Technical University Department of Cybernetics' Agent Technology Center and a master's degree student in engineering informatics with the Czech Technical University's Faculty of Nuclear Sciences and Physical Engineering. His current research interests center on network security and opponent and threat modeling. Stiborek has a bachelor's degree in computer science from the Czech Technical University Faculty of Nuclear Sciences and Physical Engineering. Contact him at stiborek@agents.felk.cvut.cz.

Karel Bartoš is a researcher at CESNet, a researcher with the Czech Technical University Department of Cybernetics' Agent Technology Center, and a master's degree student in engineering informatics with the Czech Technical University's Faculty of Nuclear Sciences and Physical Engineering. His professional interests include network security, trust in multiagent systems with a focus on intrusion detection, and data fusion. Bartoš has a bachelor's degree in computer science from the Czech Technical University Faculty of Nuclear Sciences and Physical Engineering. Contact him at bartos@agents.felk.cvut.cz.

Pavel Čeleda is a researcher and systems analyst at the Institute of Computer Science, Masaryk University. His research areas include network security, monitoring of high-speed networks, and development of network monitoring devices. Čeleda has a PhD in informatics from University of Defence, Brno. Contact him at celeda@ics.muni.cz.