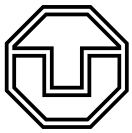

TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science Institute of Systems Architecture, Professorship of Systems Engineering

Short version of the dissertation to achieve the academic degree Doktor Ingenieur

COMMUNITY-BASED INTRUSION DETECTION

Stefan Weigert

4th March 2015

Supervisor
Prof. Christof Fetzer, Ph.D.
Technische Universität Dresden, Germany

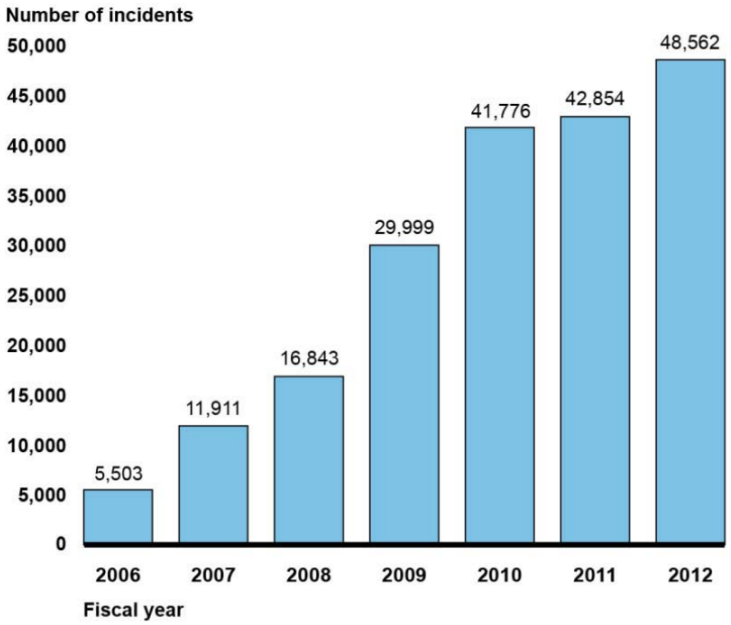

Figure 1: Number of incidents reported to US-CERT between 2006 and 1012 [Wil13]

The Internet has become a major driving force for economies around the world. Between 2006 and 2011 alone the Internet was responsible for 21% of the gross domestic product (GDP) growth[1], creating 2.6 jobs for every single job lost, as reported by McKinsey [Rau+11]. The study also finds that 75% of Internet impact on economies arises from traditional industries and that businesses making heavy use of web technologies grow and export twice as much as others. Given these observations, it comes to no surprise that today virtually every company world-wide is connected to the Internet. Unfortunately, this connectivity also exposes the companies to a vast array of online security threats. Even more so because running their daily business on the web requires their business data to be accessible from the Internet. This creates a financial incentive to attack these companies — customer credit card information, for example, can be sold on the black market or used for blackmailing. It also creates an economical incentive — industrial espionage, while not new, is now expanding into cyberspace. Finally, it creates a political incentive — terrorists and even states use cyberattacks to enforce their political agenda.

A report published by the United States Government Accountability Office [Wil13], offers a glimpse of how severe this problem has become. The authors counted the number of incidents reported by US-based federal agencies. From 2006 to 2012 alone this number has increased by 782%, as shown in Figure 1. While 37% of the incidents were still under investigation at the time the report was published, all but 7% of the remaining incidents were successful attacks.

Now the question is: How does such an attack work? In order to gain access to a system, attackers must find a vulnerable part in the IT infrastructure which they can exploit. Such vulnerabilities may be found in the software-stack as programs contain bugs or have been misconfigured, for example. Vulnerable are, however, also the humans who operate the software-stack for their daily business: For instance, email attachments are opened without authenticating the sender or prior virus scans. In fact, targeting people's carelessness has a long and often successful tradition in human history and mythology. For example, after the ten year siege of Troy during the Trojan War the Greeks built a large wooden horse with a hidden force of soldiers inside. They withdrew their remaining forces, seemingly terminating

[1]The study selected 13 countries which account for more than 70% of the global GDP.

the siege. But they left the large wooden horse — the perfect war trophy in the eyes of the triumphant Trojans. The Trojans pulled the horse inside the city walls of Troy and with it their inevitable nemesis. The very same tactic is used by attackers in cyberspace today: Since it is hard to break into a computer system from the outside, they try to make insiders (e.g., employees of a company) their involuntary allies — for example, by tricking them into executing malicious email attachments. Such a malicious attachment is accurately related to as a "Trojan" or "Trojan Horse".

Since neither software nor humans can be guaranteed to be invulnerable, IT infrastructures are secured by a multitude of defense mechanisms, such as firewalls to restrict the number of open ports or credential management to restrict the accessibility of valuable assets to a small number of users. However, all defense mechanisms are ultimately part of the software stack and operated by humans and, therefore, suffer from the same fundamental problems. It is hence desirable to at least detect when an attacker has compromised a part of the infrastructure in a timely manner in order to deploy adequate countermeasures. The research area that deals with detecting that a part of a system has been compromised is called *intrusion detection*.

1 SECURITY THREATS

Wuesst and Candid [Wue14] monitored targeted attacks worldwide in a period from July 2012 to June 2013. The study, published by Symantec, observed an average of 74 of such attacks per day with the most common targets being the government/public sector, directly followed by the energy sector. From their analysis, the authors conclude that cyberespionage and sabotage attacks are becoming increasingly common. However, even more worrisome is their statement that these attacks also become increasingly more sophisticated and that individual campaigns vary significantly in exploited vulnerabilities and malware in use.

Such targeted and sophisticated attacks are classified as *Advanced Persistent Threat (APT)* attacks [Bej10]. Juels and Yen summarize APT attacks in a single statement: "An APT isn't a playbook. It's a campaign" [JY12]. This refers to the fact that APT attacks are not automated and, therefore, need a command-and-control facility in order to complete their objective. Many of the recent high-impact attacks classify as APT attacks. Their commonalities will guide the design of our system.

1.1 SECURITY CHALLENGES

It seems that commonly applied intrusion detection techniques have failed at protecting many systems. Virvilis and Gritzalis [VG13] came to a similar conclusion. They analyzed several recent APT attacks that were targeted against vital industries. All analyzed attacks evaded firewalls by using commonly open ports, such as ports 22, 80, and 443. All analyzed attacks evaded intrusion detection (usually based on packet inspection) by obfuscating or encrypting their traffic. As a consequence, all analyzed attacks were active for several years and managed to exfiltrate high-profile data. Given what we learned so far, we derive four essential security challenges that will guide our IDS design:

Security Challenge 1: Polymorphism APT attacks vary significantly in how they are executed. Although there is a somewhat consistent story-line in every APT attack campaign as detailed in this section, the actual implementation varies in the malware used, the exploited vulnerabilities, and traffic patterns. This means that an intrusion detection system must be able to dynamically add new rules in order to look for additional patterns.

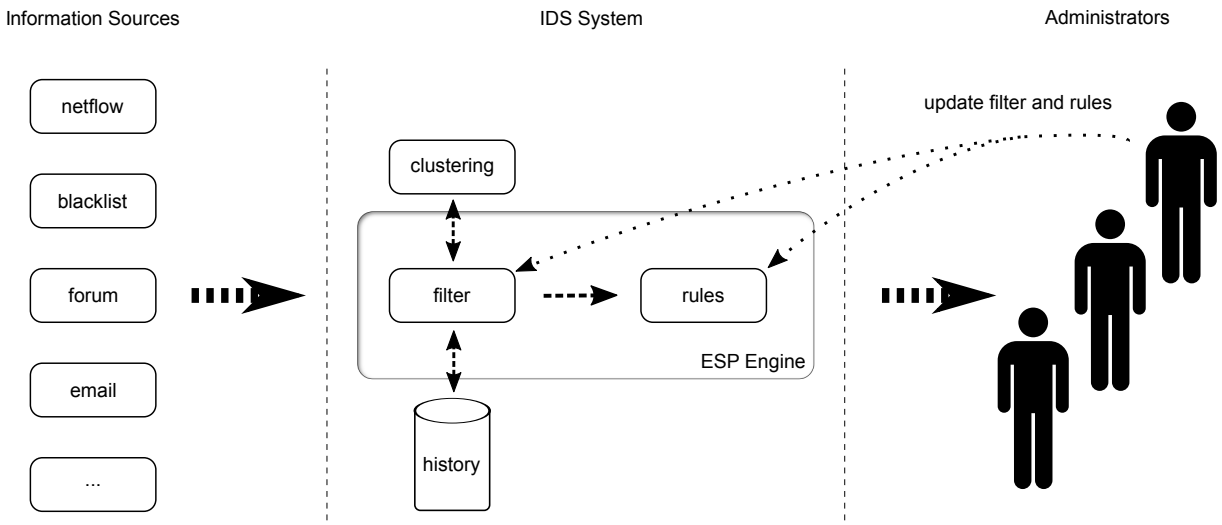

Figure 2: Architecture of a community-based intrusion detection system.

Security Challenge 2: Encrypted, obfuscated, or compressed traffic with command-and-control servers The in- and out-bound traffic used to control the malware and download data employs encryption, obfuscation, compression, or any combination of these techniques to conceal the attack. Thus, the intrusion detection system must not rely on packet inspection.

Security Challenge 3: Industry-oriented attacks APT attacks, by their very meaning, are targeted. However, as our analysis and the analyses of others [VG13; Wue14] have shown, the actual target may be as broad as a complete industry. This means that an effective defense against such threats needs to operate at the industry-level as well.

Security Challenge 4: Stealthy attacks Since APT attacks act with a specific goal in mind they remain stealthy throughout the complete attack until they have completed their agenda. Such "below the radar" attacks are extremely hard to discriminate from the background-noise which is inherent to large-scale communication.

This thesis therefore tries to envision and implement a system which is capable of monitoring an entire industry for suspicious activity.

2 AN ARCHITECTURE FOR A COMMUNITY-BASED IDS

This section is concerned with a system design capable of handling the aforementioned security challenges. We will discuss design decisions step-by-step, starting with the choice of input datasets and then proceeding to how these data-sources are processed. This section will be closed by discussing the contributions, presented within this thesis.

2.1 INPUT DATA

Figure 2 is an abstract depiction of an IDS, as envisioned in this thesis. Our primary input-data is IP flow information provided by an *Internet Service Provider (ISP)*. This choice is guided by Security Challenge 2: Encrypted, obfuscated, or compressed traffic with command-and-control servers, which makes packet inspection impossible, and Security Challenge 3: Industry-oriented attacks, which calls for a cross-domain dataset. IP flow information is usually gathered at the edge routers of ISPs in a format called *netflow*. Roughly speaking, it contains

information of every route established from a source IP to a destination IP, including duration, amount of transferred bytes, used protocol, and many more properties. Another important input-dataset are online blacklists, such as *DShield*[2]. At least in some of the aforementioned attacks, previously hacked systems have either been used as command-and-control server, for the lateral movement, or for data exfiltration. Blacklists do not only offer means to detect communication from such IP addresses early, but they have proven to be reliable predictors of future attacks [ZPU08]. Finally, they can also be used to warn possible future victims of detected threats. A similar argument can be made for scanning forums (both security and hacker forums) for activity. Especially campaigns driven by hacktivists have reportedly been deliberately initiated from forums [Den01]. Another interesting source of information could be email traffic. For example, Wuesst and Candid [Wue14] have successfully monitored targeted attacks worlwide by scanning email traffic for spear-phishing campaigns. Our conclusion from these considerations is that an effective intrusion detection system must provide the ability to monitor and cross-correlate any number of sources. This observation raises the question: How does a processing architecture look like which can handle such a diverse set of input? Or, more specifically, what are the challenges, imposed by the input data-sets?

Data Challenge 1: Discrepancy in throughput We expect a discrepancy in the throughput of the different information sources. For example, individual IP flows (like netflow) are small in size, but an ISP collects millions of entries per second. In contrast, information from distributed blacklists is updated less frequently, but individual updates are larger than individual flows. Still, a large attack may result in a sudden burst of updates as reports are coming in. Finally, information from forums or mail servers are similarly subject to bursts. We also assume that updates from such sources are significantly larger in size than updates from other sources.

Data Challenge 2: Diversity of datatypes The input data may be structured such as IP flow and blacklists, or semi-structured such as emails and forum posts. The structure of the input data has an impact on how such information is represented and processed — for example, IP flows can be naturally represented in graphs which offer several well-defined algorithms for analysis. While forum and email databases can also be represented as graphs (e.g., with emails as vertices and edges as actions, such as reply-to, forward, and so on), the actual content of the email is unstructured and needs to be processed differently, by using text-mining algorithms, for example.

Data Challenge 3: Hidden community structure As APT attacks are targeted against complete industries (such as the complete energy industry), intrusion detection should be a community concern as well. However, a prerequisite for cross-domain event correlation to work is the ability to assign data-items to their respective communities. Unfortunately, this assignment is not explicit in any of the datasets we wish to interpret. For example, we do not know all IP addresses which belong to the energy industry. Even a manual inspection of DNS entries would not yield a correct result: Infrastructure-as-a-service has eliminated the need to host every service within the company's network (e.g., web-servers are often hosted in the cloud). But even more importantly, such an industry does not have a sharp boundary. Rather, specialized law-firms or supply companies should be counted as industry members as they are often less well protected and present the perfect entrance card into an otherwise hard to intrude business community. We expect, however, that these characteristics are reflected in how the industry members interact with one another. Hence, we assume to be able to detect an industry in our input data.

[2] `https://www.dshield.org/`

2.2 DATA PROCESSING

The processing architecture can be split into five main components where the data-stream flows from the left to the right and may or may not generate alarms as depicted in Figure 2.

The *filter* component discards any information that is not of any use for the downstream components. For example, it drops any IP flows where neither source- nor destination IP address belong to a protected industry. In order to cope with Security Challenge 1: Polymorphism, the filter needs to be adaptable, such that filtering rules can be altered, added, and removed on the fly. To accommodate for Data Challenge 1: Discrepancy in throughput, the filter needs to be scalable, such that any amount of incoming data, including spikes, can be handled. Finally, the filter must support a rich filtering language that is not tied to a specific kind of input data in order to tackle Data Challenge 2: Diversity of datatypes.

The *history* component stores all incoming traffic for a given amount of time. This may be necessary to understand exactly what actions an intruder has performed and supports the administrators in deploying effective counter-measures.

The input is also forwarded to the *clustering* component, which is responsible for maintaining and up-to-date list of IP addresses belonging to the protected industry. Protecting the industry at large allows for statistical correlation between individual companies, which makes it much harder for a targeted attack to pass below the radar [ZLK10] and, therefore, tackles Security Challenge 3: Industry-oriented attacks and Security Challenge 4: Stealthy attacks. The clustering component must, hence, be able to infer community structure from that data as noted in Data Challenge 3: Hidden community structure.

The *rules* component applies user-defined rules to the data-streams. It supports rules that operate on meta-data, such as graphs representing IP or email communication. With that, this component tackles Security Challenge 2: Encrypted, obfuscated, or compressed traffic with command-and-control servers. However, rules that operate on content are not at all obsolete. Forum posts or blacklist entries are not encrypted or obfuscated and mining such sources is of great help in understanding and predicting attacks. Therefore, this component requires a flexible (see Data Challenge 2: Diversity of datatypes) and scalable (see Data Challenge 1: Discrepancy in throughput) event stream processing architecture. The possibility for users to change, add, and remove rules accommodates for Security Challenge 1: Polymorphism and is necessary to facilitate a feedback-loop, which connects the system and the administrators. With that, an administrator is able to mark alarms as false positive or true positive, helping to improve the performance of the employed algorithms.

3 CONTRIBUTIONS

This section presents a short overview of the major contributions of this thesis. They are organized into sub-aspects of the architecture for a community-based IDS, described above.

3.1 FILTERING

This work presents a highly scalable implementation of the filter component. However, our implementation does not support dynamic addition and removal of filters during runtime and is restricted to the filtering of IP flows. Therefore, the implementation is generalized, based on the popular Publish/Subscribe paradigm. This implementation allows to filter any kind of information. Should new insights on the maliciousness of an IP address require to re-evaluate input from the past, the *history* component can be queried for such information. Its implementation is out of the scope of this thesis, but related work on providing a history within Publish/Subscribe Systems exists [Cil+03; Li+07] and could be added to our system.

3.2 CLUSTERING

This thesis contains a novel algorithm for finding industries in IP flows. IP flows are a natural candidate for detecting industries, since there is already a large body of related work that is concerned with clustering entities, given how they communicate. Moreover, we decided to base our industry detection on IP addresses because IP addresses are not only present in IP flow data (albeit most prominently), but also in the other datasets. Distributed blacklists refer to suspected or convicted IP addresses. Spear-phishing emails are sent to company mail servers, hence, allowing to assign emails to a specific company. And finally, forums also often contain references to IP addresses, either those of possible targets or attackers.

3.3 RULES

This work presents a generic, highly scalable graph-based rule-engine. This graph-based rule engine is the basis for a prototypical community-based IDS, that uses IP flow information to protect vital industries (i.e., energy, financial, and defense industries). Following that, a general processing engine, which allows to apply any kind of rule-engine to its inputs is presented. Both engines are implemented on StreamMine, an event stream processing (ESP) framework, and can be integrated seamlessly. StreamMine itself is presented in detail in separate publications [Mar+11; Bri+11].

4 CONCLUSION

This work presents a novel community detection algorithm to detect industries in IP flow data. This is necessary to understand which entities of our input data belong to one of the business communities we want to protect. In contrast to the networks which are commonly studied for community detection, Internet flow data exhibits some distinguishing characteristics. First, its edges are ambiguous, for example, a communication between two IP addresses might mean that a user accesses another machine intentionally but it may also mean that an advertisement has been downloaded or that a click-analysis site was accessed. It is hard to differentiate between such unintended connections and actual user-intended connections. Second, the business communities do not manifest themselves as tightly knit communities, as assumed by conventional community detection. Rather, they are loosely-coupled with more connections to the rest of the Internet than to other members of the industry. Consequently, we found that traditional community detection algorithms were unable to detect our business communities reliably. Therefore, we propose a new algorithm, which instead of finding a closely knit community around a seed-set, tries to expand a given set of seeds by including those neighbors with a very strong connection to the community. And, most important, without preferring those neighbors, which are otherwise barely connected to the rest of the network. Our algorithm is able to detect three sample industries — energy, financial, and defense — reliably in IP flow data while maintaining a low number of false positives.

This is followed by the design and implementation of a distributed scalable graph mining engine. This enables us to store and process the vast amounts of input data, such as netflow data from distributed ISP routers and properly represent its inter-dependencies. It supports in-memory storage and updates to the graph with high throughput and below 10ms latency. In fact, those updates can be arbitrarily complex. For example, we show how to construct a COI graph on the fly by performing window-based aggregation of graph-updates. Furthermore, it is possible to query the graph — concurrent to other queries and concurrent to the graph updates. Using these queries, we implement several important graph algorithms like the local clustering coefficient or community detection algorithms. Finally, we use our engine to implement fraud detection in telephony networks and intrusion detection in IP flow data. Our

engine shows excellent scalability, throughput, and latency for both applications.

Thereafter, a deeper evaluation of the intrusion detection application is given. In order to tune the intrusion detection for different aspects of the input data, we added different views, all running concurrently. In the case of netflow, the thesis presents three views: A COI graph whose weight is based on the amount of transferred bytes, a COI graph whose weight is based on the vulnerability of the ports used, and a COI graph whose weight is based on the number of attempted connections. We ran our engine for several months and found that we were able to detect suspicious IP addresses which maintained an otherwise very low traffic profile per company. Throughout this period, the daily alarm rate remained below 20 for most of the days. We consider this to be a manageable amount of reports for a community of security administrators. The thesis presents several example cases of possible intrusions. Post-processing alarms is exemplified by using distributed blacklists and DNS lookup services. This post-processing has been successfully applied for ranking the issued alarms.

The next step was to design a system on top of which we cannot only implement our graph-based intrusion detection technique but which would also allow for adding, removing, and adapting any arbitrarily complex detection algorithm. Furthermore, the goal was to put administrators in control of which alerts they want to receive and which not. In theory, content-based Publish/Subscribe permits exactly that: Subscribers (the administrators) subscribe to a given kind of event-content using a subscription language (e.g., a conjunction of predicates). Subscribers can add, remove, or change their subscriptions at any time. Publishers (our data sources) may send publications at different throughput and new publishers can be added during runtime. While, so far, it seemed to be the perfect solution, the state-of-the-art content-based Publish/Subscribe systems support only a fixed kind of filter (e.g., stateless conjunction of predicates) and were not suited for high-throughput and low-latency processing. Hence, we designed and implemented a new architecture for Publish/Subscribe, called StreamHub, allowing to run any number of arbitrarily complex filters within the same system. Our key contribution is that we implement the different stages, involved in filtering publications, as individual operators which we can scale separately to the current workload. Moreover, departing from the traditional peer-to-peer overlay used for Publish/Subscribe, allows us to support any kind of filter - even stateful ones, such as our graph-based intrusion detection. Our evaluation shows that StreamHub maintains low predictable latency at high throughput and out-performs the state-of-the-art by an order of magnitude.

BIBLIOGRAPHY

[Bej10] R. Bejtlich. "CIRT-level response to advanced persistent threat". SANS Forensic Incident Response Summit. 2010.

[Bri+11] Andrey Brito, André Martin, Thomas Knauth, Stephan Creutz, Diogo Becker, Stefan Weigert, and Christof Fetzer. "Scalable and Low-Latency Data Processing with StreamMapReduce". In: *3rd IEEE International Conference on Cloud Computing Technology and Science*. Athens, Greece: IEEE Computer Society, Nov. 2011, pp. 48–58. DOI: 10.1109/CloudCom.2011.17.

[Cil+03] M. Cilia, L. Fiege, C. Haul, A. Zeidler, and A. P. Buchmann. "Looking into the Past: Enhancing Mobile Publish/Subscribe Middleware". In: *Proceedings of the 2Nd International Workshop on Distributed Event-based Systems*. DEBS '03. San Diego, California: ACM, 2003, pp. 1–8. ISBN: 1-58113-843-1. DOI: 10.1145/966618.966631. URL: http://doi.acm.org/10.1145/966618.966631.

[Den01] Dorothy E Denning. "Activism, hacktivism, and cyberterrorism: the Internet as a tool for influencing foreign policy". In: *Networks and netwars: The future of terror, crime, and militancy* 239 (2001), p. 288.

[JY12] Ari Juels and Ting-Fang Yen. "Sherlock Holmes and the Case of the Advanced Persistent Threat". In: *Proceedings of the 5th USENIX Conference on Large-Scale Exploits and Emergent Threats*. LEET'12. San Jose, CA: USENIX Association, 2012, pp. 2–2. URL: http://dl.acm.org/citation.cfm?id=2228340.2228343.

[Li+07] G. Li, A. Cheung, Sh. Hou, S. Hu, V. Muthusamy, R. Sherafat, A. Wun, H.-A. Jacobsen, and S. Manovski. "Historic Data Access in Publish/Subscribe". In: *Proceedings of the 2007 Inaugural International Conference on Distributed Event-based Systems*. DEBS '07. Toronto, Ontario, Canada: ACM, 2007, pp. 80–84. ISBN: 978-1-59593-665-3. DOI: 10.1145/1266894.1266908. URL: http://doi.acm.org/10.1145/1266894.1266908.

[Mar+11] André Martin, Thomas Knauth, Stephan Creutz, Diogo Becker, Stefan Weigert, Andrey Brito, and Christof Fetzer. "Low-overhead fault tolerance for high-throughput data processing systems". In: *ICDCS '11: Proceedings of the 2011 31st IEEE International Conference on Distributed Computing Systems*. Minneapolis, Minnesota, USA: IEEE Computer Society, June 2011, pp. 689–699. DOI: 10.1109/ICDCS.2011.29.

[Rau+11] Matthieu Pélissié du Rausas, James Manyika, Eric Hazan, Jacques Bughin, Michael Chui, and Rémi Said. *Internet matters: The Net's sweeping impact on growth, jobs, and prosperity*. McKinsey & Company, 2011.

[VG13] N. Virvilis and D. Gritzalis. "The Big Four - What We Did Wrong in Advanced Persistent Threat Detection?" In: *Availability, Reliability and Security (ARES), 2013 Eighth International Conference on*. Sept. 2013, pp. 248–254. DOI: 10.1109/ARES.2013.32.

[Wil13] Gregory C. Wilshusen. *A Better Defined and Implemented National Strategy Is Needed to Address Persistent Challenges*. U.S. Government Accountability Office, GAO-13-462T, Mar. 2013. URL: http://www.gao.gov/assets/660/652817.pdf.

[Wue14] Candid Wueest. *Targeted Attacks Against the Energy Sector*. www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/targeted_attacks_against_the_energy_sector.pdf. Jan. 2014.

[ZLK10] Chenfeng Vincent Zhou, Christopher Leckie, and Shanika Karunasekera. "A survey of coordinated attacks and collaborative intrusion detection". In: *Computers & Security* 29.1 (2010), pp. 124–140. ISSN: 0167-4048. DOI: http://dx.doi.org/10.1016/j.cose.2009.06.008. URL: http://www.sciencedirect.com/science/article/pii/S016740480900073X.

[ZPU08] Jian Zhang, Phillip Porras, and Johannes Ullrich. "Highly Predictive Blacklisting". In: *Proceedings of the 17th Conference on Security Symposium*. SS'08. San Jose, CA: USENIX Association, 2008, pp. 107–122. URL: http://dl.acm.org/citation.cfm?id=1496711.1496719.