On Probe-Response Attacks in Collaborative Intrusion Detection Systems

Emmanouil Vasilomanolakis, Michael Stahn,
Carlos Garcia Cordero, Max Mühlhäuser
Telecooperation Group,
Technische Universität Darmstadt,
Darmstadt, Germany
{vasilomano, garcia, max}@tk.tu-darmstadt.de, michael.stahn@stud.tu-darmstadt.de

***Abstract*—Cyber-attacks are steadily increasing in both their size and sophistication. To cope with this, Intrusion Detection Systems (IDSs) are considered mandatory for the protection of critical infrastructure. Furthermore, research is currently focusing on collaborative architectures for IDSs, creating a Collaborative IDS (CIDS). In such a system a number of IDS monitors work together towards creating a holistic picture of the monitored network. Nevertheless, a class of attacks exists, called *probe-response*, which can assist adversaries to detect the network position of CIDS monitors. This can significantly affect the advantages of a CIDS. In this paper, we introduce *PREPARE*, a framework for deploying probe-response attacks and also for studying methods for their mitigation. Moreover, we present significant improvements on both the effectiveness of probe-response attacks as well as on mitigation techniques for detecting them. We evaluate our approach via an extensive simulation and a real-world attack deployment that targets two CIDSs. Our results show that our framework can be practically utilized, that our proposals significantly improve probe-response attacks and, lastly, that the introduced detection and mitigation techniques are effective.**

I. Introduction

Nowadays, the number and sophistication of cyber-attacks is constantly increasing [1]. To cope with this, security solutions such as IDSs [2] are considered a mandatory line of defense for any critical network. However, isolated IDSs cannot cope with large networks in terms of scalability, and they are not able to provide high accuracy due to their inherited isolation. Collaborative IDSs (CIDSs) emerged from the need for such a scalable and holistic protection of large-scale networks. As the name implies, CIDSs work by collaboratively using a number of IDS monitors [3].

Over the last years, several CIDSs that adopt the role of a cyber-incident monitor have been proposed, e.g., DShield [4] and TraCINg [5]. A cyber-incident monitor provides valuable insights into adversarial activities by visualizing and correlating alert data from a large number of monitors. These systems are of high significance for a multitude of reasons; first, they are important for the scientific community for studying attacks, experimenting with real-world attack data, creating statistics, etc. Second, they can be utilized for the detection and containment of malware propagation. For instance, DShield aided in the early detection of the Code-Red worm [6].

For all CIDSs, it is essential that the network position of their monitors, i.e., their IP addresses, is not revealed [3]. This is important for several reasons. First, an adversary with such knowledge might attempt to take down monitors, e.g., via a Distributed Denial of Service (DDoS) attack. Furthermore, malware can utilize such knowledge to evade monitors and thus remain undetected for a longer period.

Probe-Response Attacks (PRAs) are a specialized class of attacks against CIDSs that aim at detecting the network position of collaborative monitors, i.e., their IP addresses. PRAs take advantage of the need for publicly accessible alert data generated by CIDSs. In particular, they make use of the output of a CIDS, as a feedback loop, to learn confidential information regarding the monitors of the CIDS.

In this paper, we propose several improvements to the PRAs as well as mitigation and detection strategies. In more details, we first introduce an open-source framework, called *Probe REsPonse Attack fRamEwork (PREPARE)* [7], that can be utilized for performing probe-response attacks and for studying mitigation techniques. Moreover, a number of novel mechanisms for improving PRAs and also for defending against them are proposed. We evaluate our framework and proposals in an extensive simulation environment and conduct real world experiments against two different CIDSs. Our results suggest that our proposed techniques significantly improve the efficiency of PRAs. In addition, the proposed detection and mitigation mechanisms can successfully prevent the improved attacks.

The remainder of this paper is structured as follows. In Section II, we discuss the related work of PRAs. Section III provides a detailed overview of our system as well as our contributions in the areas of improving and mitigating PRAs. Afterwards, in Section IV, we present and discuss our simulation and real-word evaluation results. Finally, Section V concludes this paper and provides insights into future work.

II. Related Work

As a whole, CIDSs can be classified, based on their network architecture, as centralized, hierarchical or distributed [3]. In this paper, we focus on CIDSs that publish their alert data publicly over the Internet. Even though most of the existing systems that lie in this category exhibit a centralized architecture, e.g., [4], [5], the applicability of the attacks discussed in this paper is agnostic to the architecture. The only requirement is to have access to the alerts generated by the CIDS.

PRAs were introduced by Lincoln et al. [8] and were further discussed by several researchers, e.g., [9], [10], [11], [12]. An example of such an attack is given in Figure 1. The attack usually involves several steps, which can be summarized as follows. The adversary begins a PRA by dividing the whole IPv4 address space into equally sized groups (for the sake of simplicity, Figure 1 initially assumes a total of six hosts divided into two groups). Each group is assigned a distinct specially crafted watermark, also known as *marker*. This implies that every host inside a group will be tagged with the same marker. A marker can take many forms; for instance, the adversary can use an uncommon source port to afterwards distinguish the marker from the responses received from the CIDS. The adversary subsequently probes each host with the respective marker. If a monitor is present among the probed hosts, it will classify the probe as an attack and notify the corresponding CIDS server. The CIDS will publish this incident in its report. By inspecting the published reports of the CIDS, the attacker can determine, by examining the marker, to which group the monitor belongs. At a glance, the driving idea behind such a *divide and conquer* attack is that the markers can be subsequently utilized for examining the output of the CIDS and determining whether it contains signs of the markers or not. In this context, and with respect to the received output from the CIDS, the attacker can reduce the probed IP space and repeat the probing steps until the monitors' addresses are revealed.

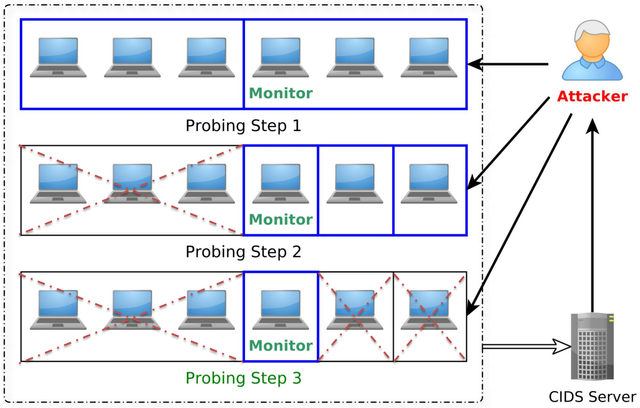


Fig. 1. Probe-Response Attack (PRA) example [13]

Bethencourt et al. presented a PRA that follows the aforementioned logic, along with algorithms for efficient probing [9]. In addition, the authors described a variety of adversarial models with regard to the capabilities of the attacker, e.g., the available bandwidth. The authors provided results of various simulations that demonstrate that their PRAs are feasible within a relatively short time-frame. The trade-off, however, is the bandwidth. One the one hand, with a network speed of 384Mbits/s, 3 days are required to conduct a complete PRA. On the other hand, with a network speed of 1.544Mbits/s, 34 days are required.

We provide significant improvements to the speed of the attacks by utilizing the state-of-the-art in Internet-wide probing (see below) and by improving the PRAs themselves. In particular and as it will be shown in the following, our framework can perform a PRA with significantly less time required and with even fewer bandwidth capabilities. In addition, to the best of our knowledge we are the first to evaluate the applicability of PRAs on two real-world CIDSs. Lastly, as we will describe in the following section, Bethencourt et al. do not effectively consider the effect of noise in their attacks. Noise refers to attacks that appear in a CIDS and are mistakenly interpreted as part of a PRA (see Section III-B).

For PRAs to be practically deployable, there is a need for efficient and rapid Internet-wide probing. The assumption behind such attacks is that a CIDS utilizes a large number of reachable monitors that are distributed all over the IPv4 address space. Over the last years research in this domain has made significant improvements, e.g., [14], [15]. In particular, Durumeric et al. [14] presented ZMap, a tool for performing Internet-wide network scanning. ZMap significantly reduces the required time for an Internet-wide probing, under certain assumptions, to one hour or less. As discussed in the next section, ZMap is also utilized inside our framework.

III. Attack and Mitigation Improvements

In this section, we first describe the structure of our framework and its properties. Afterwards, we provide insights regarding our implemented attack improvements as well as corresponding attack mitigation techniques.

A. *Probe REsPonse Attack fRamEwork (PREPARE)*

Figure 2, depicts an overview of our framework's architecture. PREPARE is written in Python and *C* and can be split into three main blocks, the *User Interface (UI)*, the *PRA logic*, and a *Wrapper* (that contains the scanner).

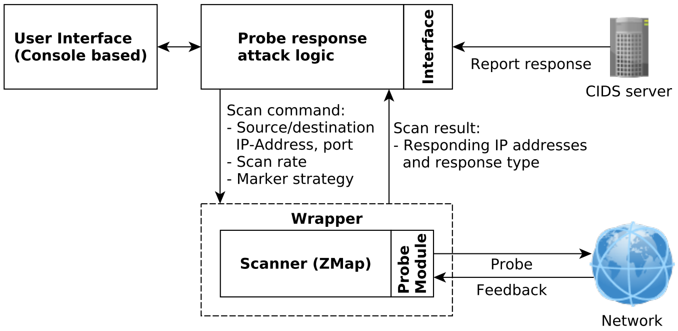


Fig. 2. A high-level overview of the PREPARE framework

The UI of PREPARE is a typical console-based interface that provides the user with all the basic commands for customizing the parameters of a PRA. The *PRA logic* implements our attack methodology based on a specific logic flow that is described in the following section (see Section III-B). The *Wrapper* contains a modified version of the core of the ZMap scanner [14]. In more detail, we extended ZMap by adding several new modules that are responsible for packet generation, response interpretation, and for handling the output. As one can observe from Figure 2, the *PRA logic* makes use of ZMap by first providing, as input, a configuration (e.g., the specific marker strategy, the scan rate, etc.) and afterwards receiving and analyzing the scan results. After the successful completion of an attack, the framework generates a CSV file that contains all information concerning identified monitors. More details about the framework can be found in [7].

B. Improving PRAs

The basic principle behind PRAs is to correlate specially marked attacks with the output information made public by a CIDS. To better understand how markers can be constructed, we provide an example that describes the utilization of destination ports as markers. Address encoding enables the attacker to map addresses of monitors to a certain port range. For instance, by encoding the first two bytes of a destination IP address range of $0.0.0.0$ to $255.255.0.0$ into a port range of 0 to 65535 ($0 = 0.0.0.0$, $1 = 0.1.0.0$, etc.), an attacker is able to decode the IP address later on. This in turn allows the reduction of the scanned address range. Based on the last example, if only the port value 1 is received from the attack report of the CIDS, further scans can be limited to the subnet 0.1.0.0/16. Assuming that the source and/or target ports are shown in the report, the attacker is able to read and decode the port information and apply this encoding methodology without additional effort.

The aforementioned encoding logic (introduced by Bethencourt et al. [9]) does not effectively take noise into account. *Noise* refers to ordinary attacks that appear in the output of the CIDS which can be falsely interpreted (by the adversary) as part of a PRA. Noise is important as it can introduce *false positives* and it can be seen as a two-dimensional problem. First, there is the case that a CIDS produces alerts in which the ports have high density (i.e., overall the detected attacks are scattered in the whole range of available ports). This degrades the effectiveness of a PRA as the number of noise-free ports (utilized as markers) is low and, hence, many false positives can be generated. Moreover, when the amount of alerts is very high and the total number of different observed ports is very low, the bandwidth requirement and re-probing amount increases. Diverging from previous related work, we propose a novel marker-encoding methodology that takes noise into consideration.

First, we argue that the utilized marker *type* is not limited to a specific field but can be rather dynamic with respect to the specifics of the targeted CIDS. For instance, in [13], we studied the distribution of possible probe markers in the DShield CIDS. Our analysis showed that from the set of all available ports, only a few are ever utilized, and that the source IP addresses can also provide enough space for a marker. Thus, introducing a combination of different probe types effectively multiplies the amount of available markers. In the following we describe our methodology for PRAs that combines the ability to utilize multiple markers along with the aforementioned need for handling noise.

1) Generic Marker Encoding Methodology (GMEM): Our approach, called GMEM, combines all available marker *values* (e.g., source/destination ports, source IP addresses, etc.) and introduces a *checksum* along with the encoded marker. The *checksum* offers a highly effective remedy against noise as all markers need to comply with a pre-computed checksum also found in the public output of a CIDS. Let $\mu = \{m_1, m_2, ..., m_N\}$ be the set of all N markers and $S(m_i)$ be the size (in bits) of marker m_i. The total amount of available marker bits M_{bits} can be calculated by multiplying the sizes of all markers as $M_{total} = \prod S(m_i)$ and deriving $M_{bits} = log_2(M_{total})$. For instance, the marker types of the source and destination ports[1] give a total marker size of $M_{total} = 65535 * 65535 = 4,294,836,225$ which results in $M_{bits} = 32$.

Figure 3, depicts the overall logic flow for a PRA that utilizes GMEM. The logic is split into four steps, namely: *Pre-selection*, *Encoding*, *Probing* and *Decoding*.

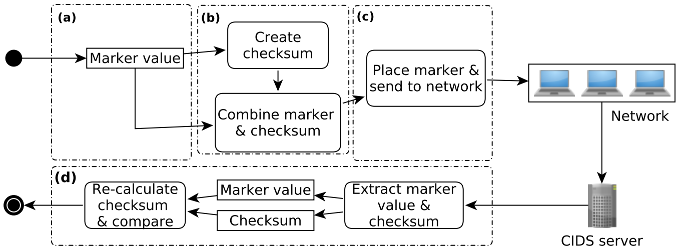

(a)
(b)
(c)
Marker value
Create checksum
Combine marker & checksum
Place marker & send to network
Network
(d)
Re-calculate checksum & compare
Marker value
Checksum
Extract marker value & checksum
CIDS server

Fig. 3. Generic Marker Encoding Methodology (GMEM) architecture and flow

a) Pre-Selection: In the first step, all available marker types are concatenated in a specific order. This creates a specific marker *pattern* that is afterwards used to generate the marker value and the checksum. As a non-exhaustive example the following marker types can be used:

- A: Destination port (16 bits)
- B: Source IP address (32 bits)
- C: Source port (16 bits)

The resulting marker pattern P can be presented as [AAAA][BBBBBBBB][CCCC], where every upper case letter represents four bits. Note, that intermixing individual bits is also allowed as long as the pattern maintains its structure throughout all steps of GMEM.

b) Encoding: In this phase, the actual marker value, e.g., the (candidate) IP address of the target monitor, is placed in the marker pattern. The IP address in this case can be represented as DDDDDDDD, and can be placed in the beginning of the pattern, transforming P to [DDDD][DDDDBBBB][CCCC]. After encoding the marker value, a marker checksum is calculated over the previously defined marker value. This checksum in turn gets placed at the end of the marker after setting all unused bits to 0. With respect to our encoding example, the marker pattern P would become [DDDD][DDDD0000][0000] before generating the checksum $C = checksum(P) = SSSSSSSS$ and appending it to the end of the marker value, which becomes marker $m = P \| C = [DDDD][DDDDSSSS][SSSS]$. Note that this simple concatenation can be exchanged with more sophisticated combinations of marker values and checksums as

[1] 65635 is the total number of available TCP and UDP ports.

long as the same procedure is reversely applied in the decoding part.

c) Probing: When the first two steps are completed, the *probing* phase can begin. Here, the generated marker m is placed in the network packets to be sent (see Figure 3).

d) Decoding: Lastly, by examining the feedback of the targeted CIDS, the decoding phase takes place. In this step, individual markers get extracted and sorted. The system calculates the checksum[2] of the marker value and compares it to the extracted checksum. In the case of a match the response is marked as accepted and can be further utilized to create subgroups so as to identify monitor nodes. Responses that fail the check are considered noise and therefore are ignored.

GMEM introduces a trade-off between noise avoidance (assigning more bits to the checksum value) and the amount of marker values used (assigning more bits to the markers themselves). Take the example of two marker types A and B, both providing four bits. With eight bits a total of 256 markers can be created. However, using all eight bits for marker encoding without any checksum would lead to a high number of false positives. Alternatively, by using six bits for address encoding (four bits from A and two bits from B) and two bits for a checksum (two bits from B), the total amount of "encodeable" markers would be reduced to 64.

From the perspective of a defender, noise can be utilized for the protection against PRAs. However, as one cannot know which bits will be taken for marker encoding (and optionally which checksum algorithm will be utilized), noise would have to be introduced for the whole target range of markers (e.g., for all the available ports). This reduces the probability of successful noise integration; that is, the probability that an introduced value matches a correct encoded attacker value. Note, however, that the filtering effectiveness increases with the amount of attack rounds because the introduced noise would have to match the probed value in every new iteration, until it introduces a false positive in the final probing. In Section IV-B1, we comprehensively study the aforementioned trade-off to better understand it and to derive the most effective parameters for a PRA.

C. PRA Detection and Mitigation

In this section, we discuss two mechanisms for defending against PRAs. The first one focuses on detecting such attacks, and the second one on reducing the effects of a PRA dynamically (upon detection).

1) PRA detection: The first step to defend against PRAs is to detect their presence in a CIDS. For this, we propose a statistical anomaly detection technique that is based on the following assumptions. First, in a generic CIDS scenario the adversary has no knowledge of either the IP addresses of the monitors nor their exact amount. Second, as a consequence of the first assumption, it can be expected that a large amount of monitors will be triggered during a PRA. Therefore, the following statistical properties are expected during PRAs:

- In a certain time-window the amount of unique monitors generating alerts is significantly increased.
- The number of unique destination (and/or source) ports will also increase (assuming probes are sent out using port-based markers).
- The number of unique source IP addresses will also increase (assuming the utilization of spoofed addresses by the adversary).

Bearing the above in mind, we propose a simple, yet effective, metric to detect such attacks by utilizing the *ratio* of generated alerts in relation to the number of actively reporting monitors. Let A be the set of all generated alerts, S be the set of all monitors, $S_t \subset S$ the set of reporting monitors within time-frame t, and $A_t \subset A$ the set of generated alerts within time-frame t. The ratio r_a is defined as:

$$r_a = \frac{|A_t|}{|S_t|}. \tag{1}$$

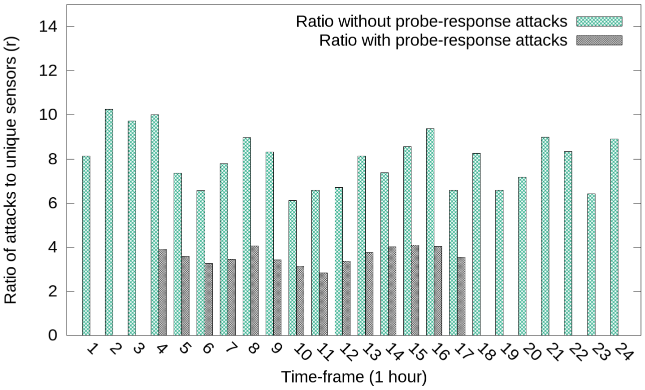


Fig. 4. Ratio r_a utilization example for DShield data [13]

Figure 4, depicts the distribution of r_a for data gathered by the DShield CIDS within a period of 24 hours. An attacker requires approximately 5 hours (with a 100Mbit/s network connection) to perform one probing step of the entire IPv4 range [14], probing approximately $90,000$ monitor addresses per hour of the total $500,000$ monitors[3]. With respect to our aforementioned assumption, in the presence of a PRA, the number of unique reporting monitors within a time-frame $|S_t|$ will increase significantly, while $|A_t|$ will only have a relatively small increase, therefore affecting r_a. In the presented period, we observe the number of monitors $|S| = 131,344$, the number of alerts $|A| = 10,934,768$, and an average unique monitor count (per hour) $\sum_t \frac{|S_t|}{24} = 55,000$.

We emulate a PRA by introducing alarms in the time-frames between 4 and 17 (which span enough time to enable three complete probing rounds) in a 24 hour period. By assuming that the maximum probing rate is $90,000$ IP addresses per second and that monitors might already be present, the PRA is conducted according to a uniform distribution between $80,000$ and $90,000$. As shown in Figure 4, it becomes evident that

[2] Note that in PREPARE is currently utilizing the hashing algorithm Fletcher32, for its efficiency [16], but this can be changed if needed by the user.

[3] This is the number of monitors that DShield is claiming to have [4].

during an attack the ratio r_a decreases significantly. Hence, the presence of a PRA can be detected.

Another technique for detecting the existence of PRAs is the studying of the *frequency* of unique destination ports within a specific time-window. In contrast to source ports (which are usually chosen randomly), the number of unique destination ports observed is expected to increase during a PRA. In this case, it is important to carefully choose the time window that will be used for studying the port frequency. Figure 5, depicts the distribution of the port frequency in DShield by setting a fixed start time and extending the window up to 24 hours. As can be observed, in the first half hour almost 93% of the ports are not utilized, while when the window is increased this percentage rapidly decreases. This suggests that large time-windows (e.g., more than two hours) are highly likely to introduce a large amount of false positives.

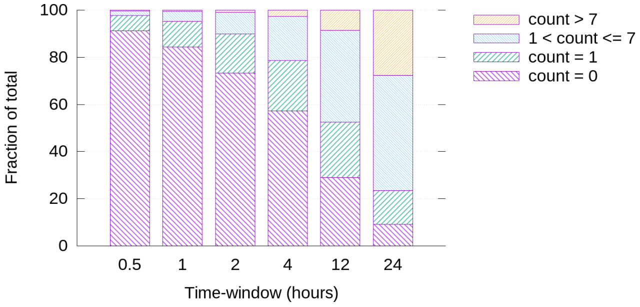


Fig. 5. Destination port frequency for different time-windows in DShield

Bearing this in mind, in Figure 6 (with a similar setup as Figure 4), we show how the frequency metric evolves under the presence and absence of an emulated PRA. It can be seen that the difference between attacked and non-attacked states can be utilized as a threshold for the detection of PRAs. In Section IV-B2, we further examine how the frequency of the non-utilized ports can be used for the detection of a PRA.

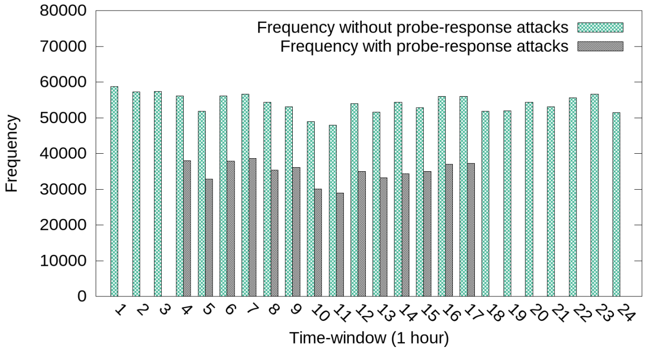


Fig. 6. Destination port frequency in DShield

2) Adaptive Reporting: Upon successfully detecting a PRA, the CIDS can perform a number of actions that aim at reducing the effects of the attack. The main goal for this is to reduce the number of identified monitors during a PRA as much as possible. That is, make it difficult for the adversary to gain enough information to derive whether an IP belongs to a monitor or not. We propose the concept of adaptive sampling, i.e., the CIDS will selectively publish a sample of the overall generated attacks whenever it detects the presence of a PRA.

Such mechanism can use the aforementioned ratio, the destination port frequency metric, or both to decide when the sampling should be activated. Furthermore, the intensity of the sampling can be proportional to the attack intensity, i.e., the more intense a PRA is, the less results are published by the CIDS. In Section IV-B2, we describe two practical variations of such an adaptive sampling approach. Furthermore, we study, in more detail, the efficiency of the two aforementioned detection techniques combined with the adaptive report sampling. As shown by our experiments, such an adaptive approach can efficiently reduce the effectiveness of a PRA. Nevertheless, this comes with the trade-off of publishing less attack results.

IV. Results

In this section, we discuss the results of our proposed attack and mitigation strategies in a simulated and a real-world environment. We begin by describing the setup and results of our simulations. Afterwards, we describe the results of testing our PRA methodology on two real-world cyber incident monitors.

A. Simulation Setup

In order to evaluate attacks and their proposed mitigation mechanisms, we have setup a simulation environment. Our simulations match the characteristics of DShield [4]. DShield is the largest and most well known cyber incident monitor, reporting thousands of potential attacks every day since ten years ago. Along with the DShield characteristics, we also take into consideration previous work in the area of Internet-wide scanning as our methodology relies on scanning the entire range of IP addresses exposed on the Internet.

All our simulations use the following parameters. We utilize a set of approximately 288.4 million responsive IPv4 addresses, as identified in [15], [17]. Within all these responsive addresses, we set randomly a total of 500 thousand monitors. This is the same number of monitors that DShield is utilizing [4]. In addition, as network traffic does not always reach its destination, we also take into account a 2% packet *drop rate*. This particular drop rate has been observed in related work [17], [14]. Lastly, we use a low *bandwidth*, i.e., 56Mbit/s, so as to mimic the bandwidth available to many users (in offices and households).

B. Simulation Results

Our simulation is split into two parts. First, we examine our proposed improvements for PRAs and, afterwards, our mitigation mechanisms. Note that our detection and mitigation techniques (see Section III-C) are generic and agnostic to the specifics of a PRA. Hence, they are able to be incorporated into any PRAs.

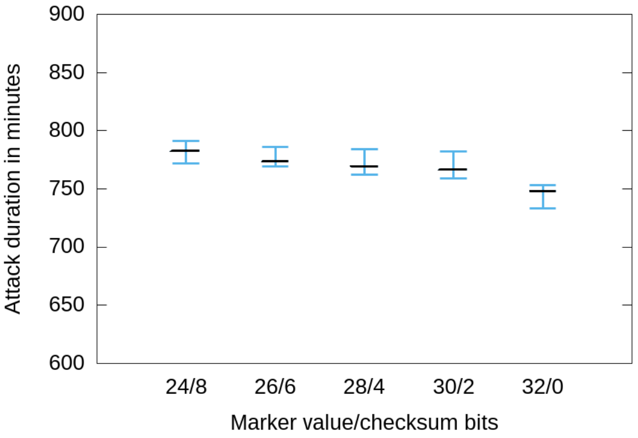


Fig. 7. Attack duration with respect to marker values and checksum bits

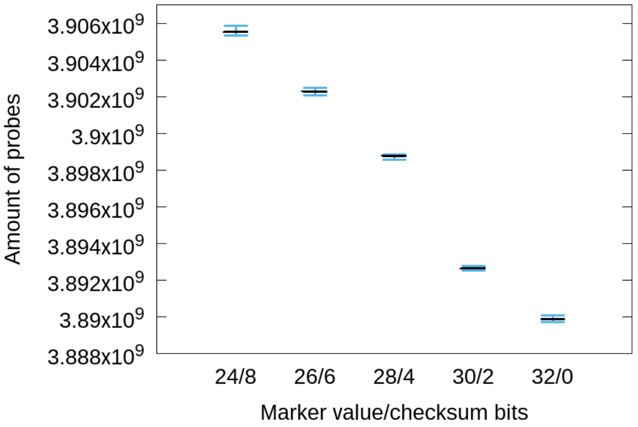


Fig. 8. Amount of required probes with respect to marker values and checksum bit

1) Improved PRA Results: We want to study the effectiveness of our generic marker encoding methodology and its efficacy in the presence of noise. We emulate noise by adding real-world data, taken from DShield, at a rate of 24 events per second.

Figures 7 and 8 present the attack duration and the amount of required probes to perform the PRA for different marker values and combination of checksum bits, respectively. The figures show that increasing the bits of the marker's value effectively decreases both the attack duration and the overall numbers of probes required. This seems to be due to the fact that the group size becomes smaller (with increased marker values). This, in turn, translates to a faster identification of empty or fully identified groups. As we show in the following, nevertheless, a trade-off exists between not utilizing checksum bits and the increase in false positives. This is the particular case where noise is taken into account.

We study the false positives introduced by noise and how our proposed checksum mechanism can assist in their reduction. Figure 9 depicts the false positives when utilizing various marker values and checksum combinations. As expected, the introduction of checksums decreases the amount of false positives at an almost exponential rate.

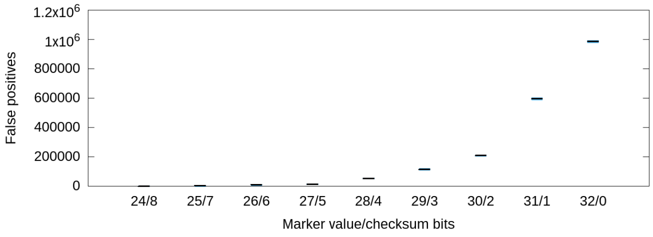


Fig. 9. False positives for different encoding configurations

Figure 10 compares our approach with the one proposed by Bethencourt et al. [9]. In particular, the figure compares the time required for enumerating all the monitors of a CIDS. This comparison is made between the PREPARE framework (using a $24/8$ marker value and checksum bits configuration) and the PRA presented by Bethencourt et al. [9]. The results show that a significantly improved performance is achieved for detecting the complete set of CIDS monitors. We also point out that PREPARE utilizes considerably less bandwidth (56Mbit/s) compared to the fastest case presented by Bethencourt et al. (384Mbit/s).

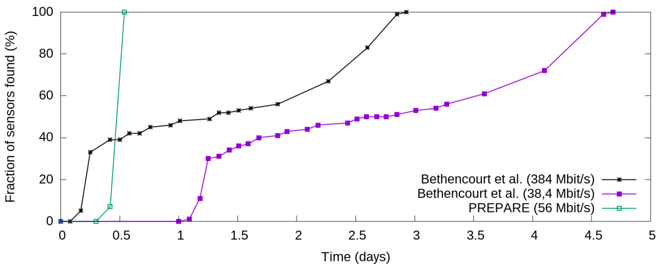


Fig. 10. PRA comparison: time required for the complete enumeration of monitors

2) Improving Mitigation: We want to study how well our mitigation strategy and detection mechanism perform. The main idea here is to utilize the ratio-based detection to first identify a PRA and afterwards perform sampling to reduce the effects of the attack. We expect a reduction in the detected monitors as well as a reduction in the total number of events published by the CIDS as a result of the sampling process.

We configure the simulation to perform PRA detection followed by adaptive sampling when the ratio drops below a certain threshold. Sampling refers, in this case, to the probability that an attack event is shown in the published results. The sampling is also adaptive in the sense that the lower the ratio is, the lower the sampling is; in other words, the sampling reacts to the intensity of the PRA. We extensively analyzed DShield output so as to choose a threshold ratio that, in the presence of PRAs, will not generate false positives. Our analysis showed that a threshold ratio in the interval $(3, 4]$ avoids triggering false negatives and false positives (with regard to the DShield CIDS). Hence, we utilize the threshold $R_t = 3$. With regard to the sampling, we utilize the sampling formula:

$$s_1 = \frac{R_m - 1}{R_t - 1} \quad (2)$$

where R_m is the measured ratio and R_t the threshold ratio. As the minimum value for the ratio is 1 (as every monitor must submit a minimum of one alert in order to be visible), subtracting 1 allows the theoretical sampling minimum value to be zero.

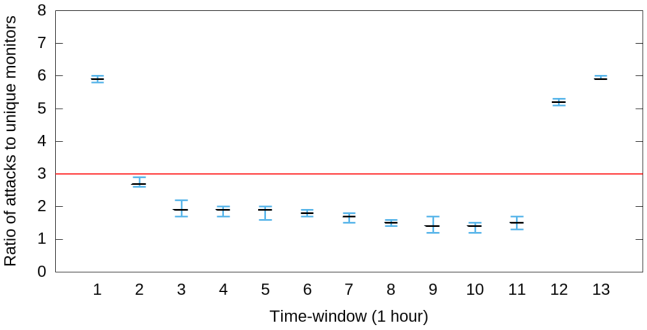


Fig. 11. Attacks/Monitors ratio (r_a) development during a PRA.

Figure 11 depicts the development of the ratio of attacks to unique monitors under the presence of a PRA. The ratio metric is checked every 60 seconds, i.e., one time-slot. The ratio (as already shown in Equation 1) is calculated by counting attacks and unique monitors for the whole time-window (one hour). The initial window state is set by loading one hour of DShield data without introducing any additional changes to the data. In total, the attack duration was 671 minutes with a total of $3,555,452,622$ probes sent. The attack started at time-window 2 and ends at time-window 11. As one can observe, the ratio drops from 2.7 to 1.5 by time-window 11. By issuing samples instead of the full range of events when the threshold ratio is exceeded, only 30.983% of the monitors are detected by the PRA. However, it should be noted that, as a result of the sampling, this technique also results in a reduction of 62% in the total number of events reported by the CIDS.

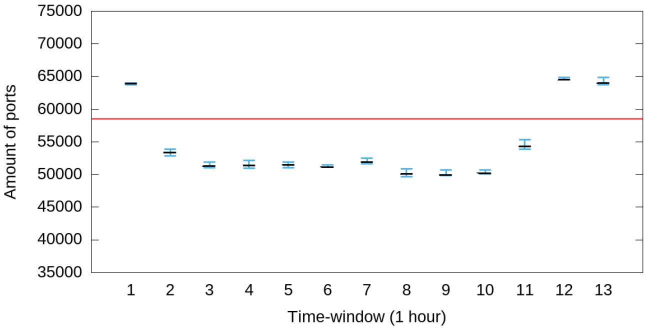
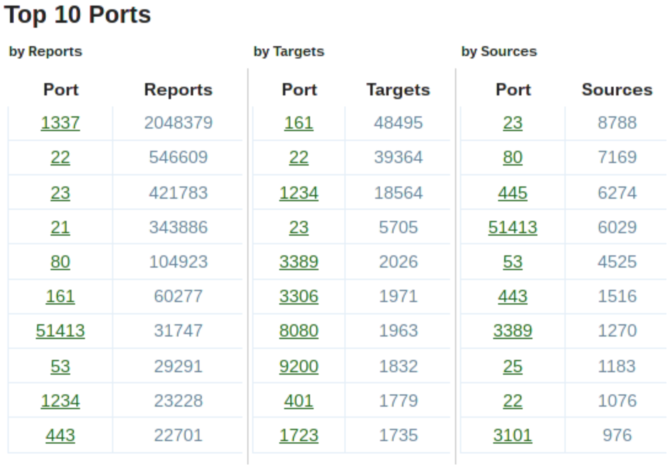


Fig. 12. Development of non-attacked destination ports during a PRA.

Figure 12 shows the development of non-utilized ports during a PRA. In this case, the sampling is done by utilizing the formula:

$$s_2 = \frac{P_t}{P_a} \quad (3)$$



Top 10 Ports

by Reports		by Targets		by Sources	
Port	Reports	Port	Targets	Port	Sources
1337	2048379	161	48495	23	8788
22	546609	22	39364	80	7169
23	421783	1234	18564	445	6274
21	343886	23	5705	51413	6029
80	104923	3389	2026	53	4525
161	60277	3306	1971	443	1516
51413	31747	8080	1963	3389	1270
53	29291	9200	1832	25	1183
1234	23228	401	1779	22	1076
443	22701	1723	1735	3101	976

Fig. 13. Top 10 Ports after the conducted PRA, as generated by the DShield CIDS

where P_t is the threshold ratio and P_a is the amount of attacked ports. With this mitigation mechanism, the PRA detected only 26.816% of the monitors but also resulted in a sampling reduction of 70% in the number of reported events.

C. Real-World Results

To further evaluate our proposals, we applied our PRA methodology against two CIDSs: TraCINg[4] and DShield.

TraCINg was tested with a bandwidth of 32Mbits/s, a marker value of 24 and a checksum of 8. The overall attack duration was $1,114$ minutes, sending a total of $3,621,468,528$ probes. Overall, 100% of the monitors were identified without introducing any false positives. The correctness of the results was confirmed both by manually re-probing the identified monitors and by examining our own ground truth knowledge (as this CIDS is deployed and maintained by our research institute).

We also applied our PRA methodology against DShield. For this, we utilized a marker value of 32 bits and no checksum. Checksums are used in the context where there is no obvious way to identify the source of the attacks. DShield, however, provides the source IP address of all attacks. As we know which IP addresses are used to perform a PRA, we do not require a checksum; we can compare the source IP address of every reported attack against our own IP addresses. Hence, we can use all available marker bits for probing. The utilized bandwidth in this case was $14,4$Mbit/s to minimize the probability of abuse complaints. The duration of the PRA was $2,071$ minutes and resulted in the identification of $1,932$ monitors, geographically distributed all over the world. Similarly to the case of *TraCINg*, we manually confirmed that the detected monitors did not include any false positives by manually re-probing the monitors and examining the CIDS's public reports.

We cannot evaluate the case of false negatives in DShield as we lack ground truth knowledge. Ten years ago, DShield

[4] http://www.tracingmonitor.org

claimed that they utilized around $500,000$ monitors. This number does not match, however, our findings. According to our analysis, we conclude that the reason for this is that many monitors are not publicly reachable (e.g., they are behind firewalls or inside LANs) and/or that the number of monitors has changed throughout these years. The aforesaid reasoning has also been confirmed as the main cause of not being able to detect all monitors from the people responsible for DShield.

Finally, Figure 13 illustrates some statistics about the ports observed in the DShield public reports after a PRA. As one can observe, the marker utilized in our PRA, i.e., port 1337, dominates the results and is considered the top attacked port. This also illustrates the easiness of not only performing a PRA but also of poisoning the results generated by a CIDS. For instance, a malicious entity could utilize such an attack to hide attacks manifested in certain protocols or ports.

V. Conclusion

Collaborative intrusion detection is an emerging field and a necessity for monitoring large networks. Probe-Response Attacks (PRAs) can considerably reduce the benefits of CIDSs and, in particular, of cyber-incident monitors that make their results publicly available. We present an open-source framework for the development and detection of PRAs and propose a number of novel techniques for improving such attacks and mechanisms for their detection. The simulation and real world results show the applicability of our framework as well as the efficiency of our proposed techniques.

In our future work, we plan to further examine possible improvements to PRAs and their detection. In addition, we intend to study methods for making such attacks stealthier as this has the potential of improving the applicability of PRAs. Lastly, with regards to our proposed sampling mitigation mechanism, further work is required to study the trade-off between hampering a PRA and withholding data from a CIDS.

Acknowledgment

This work has been co-funded by the DFG as part of projects "Scalable Trust Infrastructures" and "Long-Term Secure Archiving" within the CRC 1119 CROSSING. In addition, it has received funding from the European Union's Horizon 2020 research and innovation program, PROTECTIVE, under Grant Agreement No 700071.

References

[1] A. K. Sood and R. J. Enbody., "Targeted Cyber Attacks-A Superset of Advanced Persistent Threats," *IEEE Security & Privacy*, vol. 11, no. 1, pp. 54–61, 2013.

[2] R. Mitchell and I.-R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, p. 55, 2014.

[3] E. Vasilomanolakis, S. Karuppayah, M. Mühlhäuser, and M. Fischer, "Taxonomy and Survey of Collaborative Intrusion Detection," *ACM Computing Surveys*, vol. 47, no. 4, p. 33, 2015.

[4] J. Ullrich, "Dshield internet storm center," https://www.dshield.org/, 2000.

[5] E. Vasilomanolakis, S. Karuppayah, P. Kikiras, and M. Mühlhäuser, "A honeypot-driven cyber incident monitor: lessons learned and steps ahead," in *International Conference on Security of Information and Networks*. ACM, 2015, pp. 158–164.

[6] D. Moore, C. Shannon, and J. Brown, "Code-Red: A Case Study on the Spread and Victims of an Internet Worm," in *Second ACM SIGCOMM Workshop on Internet Measurment (IMW)*, 2002, pp. 273–284.

[7] E. Vasilomanolakis and S. Michael, "Prepare: Probe response attack framework," https://www.tk.informatik.tu-darmstadt.de/de/research/secure-smart-infrastructures/prepare, 2016.

[8] P. Lincoln, P. A. Porras, and V. Shmatikov, "Privacy-preserving sharing and correction of security alerts," in *13th USENIX Security Symposium*, 2004, pp. 239–254.

[9] J. Bethencourt, J. Franklin, and M. Vernon, "Mapping internet sensors with probe response attacks," in *USENIX Security Symposium*, 2005, pp. 193–208.

[10] Y. Shinoda, K. Ikai, and M. Itoh, "Vulnerabilities of passive internet threat monitors," in *USENIX Security Symposium*, 2005, pp. 209–224.

[11] V. Shmatikov and M.-H. Wang, "Security against probe-response attacks in collaborative intrusion detection," in *Workshop on Large scale attack defense - LSAD*. New York, USA: ACM, 2007, pp. 129–136.

[12] P. Barford, S. Jha, and V. Yegneswaran, "Fusion and filtering in distributed intrusion detection systems," in *Proc. Allerton Conference on Communication, Control and Computing*, 2004.

[13] E. Vasilomanolakis, M. Stahn, C. Garcia, and M. Muhlhauser, "Probe-response attacks on collaborative intrusion detection systems : effectiveness and countermeasures," in *Conference on Communications and Network Security (CNS)*. IEEE, 2015, pp. 699–700.

[14] Z. Durumeric, E. Wustrow, and J. A. Halderman, "ZMap: Fast Internet-wide Scanning and Its Security Applications," in *Proceedings of the 22nd USENIX Security Symposium*, 2013, pp. 605–619.

[15] D. Maan, J. J. Santanna, A. Sperotto, and P.-t. D. Boer, "Towards validation of the Internet Census 2012," in *20th EUNICE/IFIP EG 6.2, 6.6 International Workshop*. Springer, 2014, pp. 85–96.

[16] J. Stone, M. Greenwald, C. Partridge, S. Member, and J. Hughes, "Performance of Checksums and CRCs over Real Data," *October*, vol. 6, no. 5, pp. 529–543, 1998.

[17] J. Heidemann, Y. Pradkin, R. Govindan, C. Papadopoulos, G. Bartlett, and J. Bannister, "Census and Survey of the Visible Internet," in *Proceedings of the ACM Internet Measurement Conference*, 2008, pp. 169–182.