Combating against Hacks on Encrypted Procedures by using DTRAB through Strategically Distribution Monitoring Stub

C.Mani, MCA., M.Phil., M.E, V.Santhoshkumar,
Associate Professor, Final year,
Department of Computer Applications, Nandha Engineering College/Anna University, Erode, India.

Abstract—*The unbridled development of the Internet and the network-based applications has contributed to enormous security leaks. Even the cryptographic procedures, which are used to provide secure communication, are often targeted by diverse hacks. Intrusion detection systems (IDSs) are often employed to monitor network traffic and host activities that may lead to unauthorized accesses and hacks against vulnerable services. Most of the conventional misuse-based and anomaly-based IDSs are ineffective against hacks targeted at encrypted procedures since they heavily rely on inspecting the payload contents. To combat against hacks on encrypted procedures, we propose an anomaly-based detection system by using strategically distributed monitoring stubs (MSs). We have categorized various hacks against cryptographic protocols. The MSs, by sniffing the encrypted traffic, extract features for detecting these hacks and construct normal usage behaviour profiles. Upon detecting suspicious activities due to the deviations from these normal profiles, the MSs notify the victim servers, which may then take necessary actions. In addition to detecting hacks, the MSs can also trace back the originating network of the attack. We call our unique approach DTRAB since it focuses on both Detection and TRACE Back in the MS level. The effectiveness of the proposed detection and trace back methods are verified through extensive simulations and Internet datasets.*

Keywords —*Computer security, encrypted procedure (crypto-graphic procedure), intrusion detection system (IDS)*.

I. INTRODUCTION

Cryptographic procedures rely upon encryption to provide secure communication between involved parties. A wide range of cryptographic procedures are employed by popular applications and services to ensure data confidentiality, integrity, and authentication. For demo, Secure Socket Layer (SSL) and its successor Transport Layer Security (TLS)

Cryptographic procedures have also been developed for the network level such as IPSec, which is extensively used in virtual private networks (VPNs). The purpose of all these encrypted procedures is to resist malicious intrusions and eavesdropping.

It is, however, ironic that the network services and applications become vulnerable once the underlying encrypted procedures get compromised.

The number of hacks against encrypted procedures has in-creased significantly in recent times. With the evolution of high-speed Internet and processing power, it is only natural to assume that more sophisticated hacks will emerge and pose serious threats to encrypted procedures. This raised a huge concern among the networking community since the versions of SSL procedure used by thousands of Web servers were then vulnerable to this type of attack. More innovative hacks like the Bleichenbacher hacks took advantage of flaws within the PKCS #1 function to gradually reveal the content of a RSA encrypted message. Although these hacks involved the transmission of several million trial-and-error-based cipher-texts to the encrypted Web servers, they practically implied that a SSL session key could be exposed in a reasonable amount of time, perhaps a day or less. Later on, the timing attack devised by Boneh and Brumley extracted private keys from an Open SSL-based Web server in less than 6 h, leading to a fascinating breakthrough in the field of network security. Similar hacks against SSH also came into use, such as Portable Open SSH PAM timing hacks in which an attacker could determine the existence of a given login by comparing the time the remote SSHD-daemon took to refuse an invalid password for a nonexistent login to that for a valid login. As it is evident that these hacks do exist in practice, it is imperative that these threats be detected as early as possible in order to thwart them. Our topic of interest is a distributed detection mechanism that is able to detect the anomalous events as early as possible, especially before significant damage is inflicted on the victim by the attacker. The coordination of distinct agents monitoring the network flows at different points requires an appropriated architecture that must be developed. We address these issues in our paper efficiently and attempt to design ad-equate solutions to these problems. We propose the DTRAB scheme, which is not limited to constructing a defensive mechanism to discover hacks; we devise an aggressive countermeasure that not only detects a potential threat, but also investigates the root of the threat by attempting to trace back the attacker's original network or sub network.

The remainder of this paper is organized as follows. Section II surveys some related work on intrusion detection and trace back systems while exploring the shortcomings of these contemporary approaches to deal with cryptographic hacks. Section III at first presents the scope of hacks that may be detectable by DTRAB. In this section, the network topology that may be considered is then provided, which is based upon the monitoring stubs (MSs). The functionality of the MSs in order to detect and trace back the hacks against cryptographic procedures is then illustrated in four modes, namely learning, detection, alert, and trace back phases. The performance of DTRAB is evaluated in Section IV with the aid of simulations. Due to difficulties in obtaining real encrypted traces that are rather sensitive in this section also demonstrates the application of DTRAB for detecting non encrypted hacks in Internet datasets. Finally, Section V concludes the paper.

II. RELATED WORK

A. Previous Work on Detecting Hacks Against Encrypted Procedures

Intrusion detection has been an active field of research for over two decades and most conventional IDSs operate by inspecting the contents of the networking packets. Once encrypted, the packet contents are garbled and the intrusion detection systems (IDSs) fail to recognize whether the payloads are normal or potentially malicious. As a result, despite substantial research and commercial investments, traditional IDSs still re-main ineffective when they encounter encrypted traffic.

Intrusion detection systems can be broadly categorized in two ways, namely signature-based and anomaly-based detection techniques. A signature-based (also known as rule-based or misuse-based) IDS uses previously stored attack descriptions to compare if a portion of the monitored network packets is malicious.

The attack descriptions or attack signatures may be simply stored as patterns or may use complex approaches based on state machines or neural networks to map multiple features to abstract the overall attack manifestation. Since a given signature is associated with a known attack abstraction, a signature-based detector can usually assign names (such as Smurf or Ping-of-Death to hacks with ease. If a similar attack manifestation is found, signature-based IDSs can identify previously unseen hacks, which are equivalent to known patterns. Having said this, the signature-based detection schemes are inherently incapable of detecting truly novel hacks and suffer from high rates of false alarms when attack signatures match both intrusive and nonintrusive patterns. On the other hand, the primary strength of an anomaly-based detection scheme is its ability to recognize novel hacks.

At first, statistical models and artificial intelligence (AI) tools such as neural networks are employed to characterize a normal profile. At the advent of a malicious event, an anomaly-based technique senses a significant deviation from the normal profile. The drawbacks of the anomaly-based IDS include higher false alarm rates and the difficulty in classifying or naming hacks.

Recent research efforts have been devoted toward detecting various hacks against encrypted procedures such as SSL/TLS and SSH. For instance, an attacker launching the infamous re-mote timing attack against an Open SSL server could extract the private key stored in the server within 6 h. All the attacker had to do was to measure the time the Open SSL service took to respond to decryption queries on a trial-and-error basis. Cancel *et al.* illustrated password hacks against Internet Message Access Procedure (IMAP) servers using SSL-tunnel with its peer users. This compromised the security of mail transactions even under encrypted sessions. Version 3.0 of SSL was found to be vulnerable against the Version Rollback attack in which an attacker tricked the server to downgrade its version of SSL to 2.0. By doing so, the attacker could then take advantage of the vulnerabilities associated with the lower version of SSL. Additionally, buffer-overflow hacks against open SSL servers led to denial of service (DOS)-like phenomena. Propagation of the Slapper worm is a notable demo that posed a serious DENIAL ATTACK threat to Apache Web servers that use the mod-SSL library. McClure *et al.* identified encryption to be the biggest inhibitor to the development of network-based IDSs. By encrypting traffic over SSL, the commercial Web servers practically blind the network IDS sensors from detecting hacks. The only defence against this shortcoming of the network IDS is on-the-fly SSL decryption technology such as SSL Dump However, such approaches require a copy of the SSL server's private certificates and present an additional security hazard. Yamada *et al.* illustrate an encrypted traffic analysis to reinforce the detection of encrypted Web intrusion. Their method focused on analysing the contents of the encrypted tr Affric by using only data size and timing without having to resort to decryption. Access information in terms of data size and timing for every Web customer was extracted from the encrypted Web traffic by reconstructing the TCP sessions and the headers of the encrypted sessions. Based on the low access frequency of malicious activities, the encrypted traffic analysis statistically detected rare events as anomalies and reported the same as suspicious hacks.

An overlay-based architecture called Web SOS comprising access points, beacons, and servlets, has been conceived to enable a Web server to function even under a DENIAL ATTACK. The end-to-end communication between a customer and the server is secured by SSL sessions. When an access-point is attacked, Web SOS chooses another access point so that traffic from legitimate customers can still enter the overlay.

On the other hand, if a node is under attack, the overlay topology is modified by computing new paths to other nodes in the overlay. In order to thwart the Man-in-the-Middle (MITM) attack against SSL and TLS-based customer/server communication, "SSL/TLS session-aware user authentication" has introduced a new approach. In this method, a customer first authenticates and provides a credential to a legitimate cryptographic server, which then generates a corresponding user authentication code (UAC) and an initial SSL/TLS session. In the event that a MITM attacker steals the UAC, he cannot modify the UAC contents, which are encrypted. To pretend as a valid customer, the attacker then requires to send the UAC using his own SSL/TLS session, which is not the same as the server-generated session. This session awareness prevents the retransmission of any intercepted UAC. Proto Mon an anomaly- based IDS for both cryptographic and application-level procedures, includes the use of lightweight procedure monitors to detect a deviation from a previously constructed normal behaviour profile. Proto Mon functions in three modes, namely Learn, Detect, and Prevent modes. First in the Learn mode, a monitoring stub per server constructs normal usage patterns for the monitored procedures. In the Detect mode, Proto Mon constantly compares the online observations with the acceptable threshold of normal profiles. Once the system detects an anomaly, it switches to the third and final mode, in which the monitor stub slows down the procedure response so that the anomaly may not go beyond the threshold level.

The delay is re-moved when no more anomaly is detected. Despite the unique features introduced in Proto Mon, there are several significant shortcomings. The use of a simple arbitrary threshold to deter-mine the anomaly is inadequate due to the dynamic changes in the network behaviour. Integrating procedure monitors with the procedure library will also affect the system once the procedure libraries require to be updated. The delay imposed in the Pre-vent mode serves as a mere damage control mechanism. Furthermore, a monitoring agent for each server serves the purpose, but perhaps not as efficiently as compared to a distributed set of monitoring stubs exchanging information. The current work (DTRAB) presents a solution to these problems by using a dynamic there scolding scheme to detect anomaly and by distributing unique monitoring agents over the network topology.

B. Previous Work on Tracing Back Attackers Against Encrypted Procedures

In addition to detecting hacks, the issue of tracing back the attackers has remained a challenging problem over the years. Many researchers have focused on integrating trace back with detection schemes . Some traditional trace back techniques use a variation of the Time-Efficient Stream Loss-tolerant Authentication (TESLA) procedure to generate a code, based on the IP addresses of the routing devices, that sequentially handles packets. By employing a map of the IP addresses of all up-stream routers, the victim of an attack can efficiently reconstruct the route of a packet up to 32 devices. These works focus on identifying the route within the network packets without in-creasing the packet size, which has challenged trace back re-searchers over the recent years. Other trace back approaches require the routers to generate additional packets for each packet that passes through the routers.

The victim host receives both the original packets and these extra packets, which provide identification of the originating routing devices. The obvious disadvantage of this approach is an increase in the network traffic. In order to deal with this, proposes an extra "trace-packet" to be generated on a probabilistic basis, for instance approximately one trace-packet for every 20,000 packets. This approach works for hacks involving a large number of attack packets (e.g., TCP SYN-flood) lasting for a reasonable length of time.

However, an attack causing a lower volume of attack packets can evade this system since enough trace-packets are not generated for successful reconstruction of a path back to the attack-host. Trace back techniques such as probabilistic packet marking (PPM) and its enhanced variant and other packet marking schemes including deterministic packet marking (DPM), ICMP trace back (TRACE), and logging techniques such as Source path Isolation Engine (SPIE) require the IP header information. This requirement poses difficulty in tracing back an attacker that sends encrypted packets since the trace-back modules need to decrypt the headers of the attack packets. However, decrypting packets at intermediate monitoring agents will not be effective since this will contribute to significant overheads and violation of privacy.

One of the most common techniques to evade detection is the use of "stepping stones", where an attacker often masks his identity by launching hacks from intermediary hosts that were previously compromised. This enables the attacker to use a chain of interactive connections using procedures such as

SSH to dispatch malicious commands over the "stepping stone" chain to gain access to the victim machine. It is, indeed, difficult to trace back the trail of the attacker owing to the sheer volume as well as the chaotic nature of the traffic on the Internet. The final victim can, at best, see the traffic from the last hop of the chain of the stepping stone. In quest of tracing a stream of attack packets through a number of "stepping stones," content-based stream-matching approaches came into use. One notable ex-ample of such an approach is "Thumb-printing", which shows good performance in tracing back stepping-stone hacks involving non encrypted procedures only. Alternate approaches include correlation methods based on inter packet-delay (IPD) for tracing back hacks against encrypted connections.

IPD remains as a distinctive feature in normal interactive connections that employ encrypted procedures such as SSH. By correlating IPD of different connections across the network, this approach identifies whether the inspected tokens belong to the same connections. Because there are different correlation points in the experimental setup, connections that are highly correlated can be tracked in a reverse way.

Blum proposed an algorithm based on the distinctive characteristics such as packet size and timing information of the interactive traffic rather than the packet contents. Using the algorithm, it was possible to find stepping stones even when the traffic was encrypted. The timing-based algorithm performed more efficiently compared to the traditional context-based techniques. Blum. investigated not only the detection of interactive stepping stones, but also made attempts to determine an algorithmic bound over the detection approach. The step-ping-stone detection problem sheds some light on the difficult ordeal of tracing back hacks against encrypted procedures.

III. PROPOSED DETECTION AND TRACEBACK SCHEMES—DTRAB

The previous sections revealed that more attention needs to be paid in detecting and tracing back hacks against encrypted procedures, as contemporary techniques fail to adequately combat with these threats. In this section, we propose DTRAB, which hinges on its ability to detect anomalies in the procedure behaviour that serve as indications of hacks.

A. Envisioned Hacks

In this section, we attempt to clarify the different attack classes that DTRAB may address. For instance, the Open SSL implementation of SSL is particularly vulnerable to specific remote timing, MITM, buffer overflow, and version rollback hacks. In the remote timing attack, SSL renegotiation attack, and password attack (also known as dictionary attack or brute force attack) against SSH, there is a high interaction between the attacker and the cryptographic procedure server.

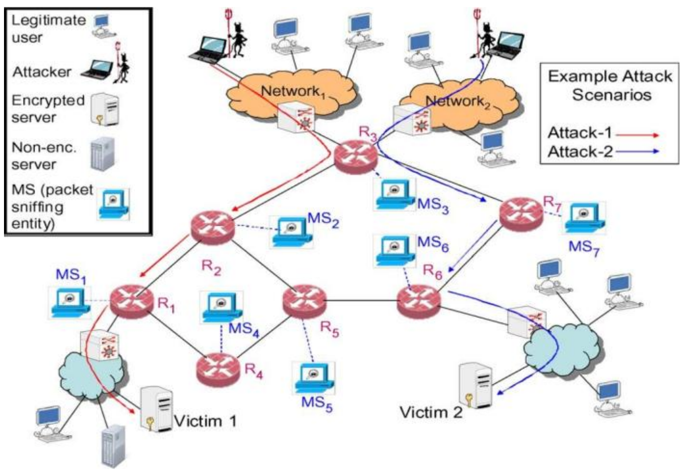
Scanning hacks that examine the existence and configuration of a generic Web server or a proxy server (or even an encrypted HTTPS-enabled server) at an IP address also contribute to a high volume of attack features. Directory-traversal hacks also exhibit similar characteristics. We broadly call these hacks the "highly interactive hacks." On the contrary, few messages are exchanged between the customer and the server in a buffer overflow attack, which can execute arbitrary codes on the victim server by overwriting stack or heap memory of the process. We term these hacks as "low interactive hacks." Cross-scripting languages that lead to at-tack vulnerabilities may also be categorized as "low

interactive hacks." Since a lot of applications (e.g., patches, antivirus, malware detection tools) are used to counter these low interactive buffer-overflow and cross-scripting hacks at the hosts, we mainly provide demos on detecting highly interactive hacks in the rest of the paper.

B. Expected Network Topology

We intend to implement the IDS and trace back entities a side
network elements, such as edge or border gateway routers. The obvious reason behind this choice is to avoid the additional computational load and memory overhead occurring at the server if the detection and trace back modules are integrated with the servers. Consequently, this paper contributes by en-visioning uniquely designed IDSs, which we call monitoring stubs, or simply MSS. In contrast with the monitoring agents dedicated to each server as proposed in [5], the proposed MSs are distributed (e.g., aside gateways, edge routers, and some of the selected core routers) over the entire network topology. An demo scenario of the envisioned network topology, which consists of a number of servers running services based on both encrypted and application-level procedures. Users from an unfrosted network or from the Internet may connect to any one of these servers. Seven MSs are placed aside the network elements. The MSs, by sniffing, monitor the traffic headers but do not inspect the payloads. When an attack is launched by a host (in Network-1), say from the untreated network to victim server 1, MS_3, MS_2 , and MS_1 consequently observe an influx in abnormal procedure operations interpreted as an attack feature. In the remainder of this section, we shall de-scribe how the MSs effectively detect hacks against encrypted procedures and try to trace back the attacker. Furthermore, by specifying the normal operation modes and request-for-comments (RFCs)

specifications of different procedures in the MS' databases, this approach may also



be extended to detect hacks For ease of under-standing, we include two demos demonstrating the deviations from normal operations of two encrypted procedures (SSH and HTTPS,

respectively) under attack that may be considered as possible attack features. Since a MS is only a packet-sniffing entity located aside a router, it does not slow down the network traffic. In case of application level procedures, it is a trivial task to sniff both the net-work packet headers and the payload contents and to inspect and analyse the information afterward. For encrypted procedures, a MS needs to adopt a different approach. A MS utilizes the TCPDUMP tool to monitor the TCP headers that are not encrypted. For demo, in order to detect a failed SSH session due to a password-based attack against SSH-based services on port 22, a MS requires to know how the SSH procedure works in the transport layer level. At first, a customer attempts to establish a connection to the server by sending a SYN packet. The server acknowledges this by sending an ACK and a SYN packet of its own.

If the customer manages to successfully log onto the server and wants to quit, the customer will initiate the FIN packet first. This is a normal mode of operation in SSH. On the contrary, if the server initiates the FIN packet first, it indicates that the server is shutting down the connection because of either an invalid attempt to access the service or a timeout.

IV. PERFORMANCE EVALUATION

First, we evaluate the performance of DTRAB with small-scale computer network-based experiments and simulations. The simulations were conducted five times, and the average values are used as results. To verify the performance of DTRAB in the large-scale networks, we also apply the DTRAB detection scheme on different unencrypted Internet traces obtained from CAIDA datasets.

A. Performance of the Detection Scheme

1) Highly Interactive Attack Detection:

a) ***Experimental setup****:* Since each MS deployed in the considered architecture sniffs and monitors encrypted traffic only, this mechanism is inherently same for all the MSs, and therefore, we do not need to redundantly provide detection performance for every single MS. We considered MSs one hop away from the victim cryptographic server for evaluating the proposed detection method. To evaluate the attack detection by an individual MS, a SSH server was targeted, which ran on Open SUSE Linux 10.1 with 3.2 GHz processor speed and 1 GB memory. A customized SSH traffic generator was designed at the customer end, which ran on Windows XP, and the procedure version used was SSH-2. In normal scenario, the SSH connection arrivals follow a Gamma distribution, shape and rate parameters of which are set to 0.2784 and 0.2260, respectively [38]. The sniffer running on a virtual machine configured on the server acted as the monitoring stub, MS_a in this experiment.

a) Results and analysis: The system achieves that no one can't miss behave with the network access points. i.e if any one want to make interrupt with another system using any kind of hacks then system will recognizes that it belongs to unauthorized access. After that automatically computer will be shut down for preventing from unauthorized access.

b) ***DTRAB Detection Phase:*** The detection approach adopted in DTRAB involves detecting anomalies. This relies on detecting the point of change in the encrypted procedure behaviour as quickly as possible under an attack. For this purpose, we employ the nonparametric Cusum algorithm, which is a statistical tool. The impact of the statistical application of the nonparametric Cusum algorithm in the analysis of attack features extracted from the packet headers in a unique manner is our contribution. Our ingenuity lies in how we treat the problem of cryptographic attack detection and apply the statistical tool. We realize that it is expensive to employ the classical version of the Cusum algorithm and other change-point detection algorithms due to the manner in which they demand to learn about statistical probabilities of hypotheses of the normal and abnormal events *a priori*. Such hypotheses are referred to as parameters.

Furthermore, Internet traffic cannot be modelled appropriately based on such hypotheses. This is why we choose to adopt the nonparametric version of

Cusum, which is a lightweight algorithm applicable to the traffic in the Internet, including the scope of encrypted traffic. It is to be noted that the term "nonparametric" implies that the scheme may be adopted without having any knowledge of the traffic distribution beforehand. We employ the nonparametric Cusum algorithm at the MSs to detect points of changes in the network behaviour at the advent of an anomaly. The ability of this scheme to detect minute changes in the network profile, and the ease with which it can be deployed at the MS-level encourages us to select the non parametric Cusum algorithm as the core detection tool.

c) DTRAB Alert Phase: When the nonparametric Cusum algorithm detects an anomaly, the sequence begins to increase. Once exceeds, the MS generates an alert to the server and the neighboring MSs. The MS can also request the server to slowdown the procedure response in an attempt to thwart intrusions such as remote timing hacks. Finally, the MS switches to the trace back mode to identify the attacker, which will be described in the next subsection.

d) DTRAB Trace back Phase: The proposed trace back mechanism relies on the collaboration of the MSs to correlate monitored abnormal operations of the procedures (e.g., failed session rates) over time. Every MS, in its database, stores the information of failed sessions that it observes for both incoming and outgoing traffic. This database also contains a list of collaborating MSs, with which the MS can contact in order to reconstruct the attack path. Once an attack is detected, the MS closest to the victim encrypted server goes to the "Trace-back mode."

V. CONCLUSION

In this paper, we addressed the online detection of hacks against application-level procedures, which are encapsulated inside encrypted sessions. Experiments carried out in the real data network have provided evidence that implementation of the proposed DTRAB in the monitoring stub (MS) is feasible. DTRAB is autonomous at the MS that carries out the detection, i.e., the detection method does not need information from other MSs. Furthermore, the MS builds the database portraying the normal procedure behaviour profile, which is not dependent on the traffic volume. As a result of this design, the proposed detection scheme manages to avoid false alarms during flash crowd. The conducted simulations demonstrate the effectiveness of the detection technique. Our investigations have considered the attack detection delay and the "failed session detection error rate." We have also addressed the problem of tracing back attackers against encrypted procedures based on the correlated attack features at neighboring monitoring stubs.

As an approach of responding to the detected hacks, this work may be extended to selectively slow down the procedure response as long as the Cusum sequence exhibits anomalous behaviour. Admittedly, when IPSEC procedure is employed by end-hosts through a secure tunnel, the transport layer headers may be encrypted and not visible to the MSs. Our future ex-tensions to this work will consider how DTRAB may overcome such issues. The further extensions of our work may also facilitate combating against hacks on encrypted procedures in the wireless network environment.

REFERENCES

[1] C. E. Landwehr and D. M. Goldschlag, "Security issues in networks with internet access," *Proc. IEEE*, vol. 85, no. 12, pp. 2034–2051, Dec. 1997.

[2] D. Bleichenbacher, "Chosen Ciphertext hacks against procedures based on the RSA encryption standard PKCS #1," in *Proc. 18th Annu. Int. Cryptol. Conf.*, Santa Barbara, CA, Aug. 1998, pp. 1–12.

[3] D. Brumley and D. Boneh, "Remote timing hacks are practical," in *Proc. 12th USENIX Security Symp.*, Washington, DC, Aug. 2003, p1.

[4] "Open SSH PAM timing hacks," 2006 [Online]. Available: http://se-curityvulns.com/news2789.html

[5] S. P. Joglekar and S. R. Tate, "ProtoMon: Embedded monitors for cryp-tographic procedure intrusion detection and prevention," *J. Universal Comput. Sci.*, vol. 11, no. 1, pp. 83–103, Jan. 2005.

[6] Z. M. Fadlullah, T. Taleb, N. Ansari, K. Hashimoto, Y. Miyake, Y. Nemoto, and N. Kato, "Combating against hacks on encrypted pro-tocols," in *Proc. IEEE ICC*, Glasgow, Scotland, Jun. 24–28, 2007, pp. 1211–1216.

[7] H. Wang, D. Zhang, and G. Shin, "Change-point monitoring for the detection of Denial attack hacks," *IEEE Trans. Depend. Secure Comput.*, vol. 1, no. 4, pp. 193–208, Oct.–Dec. 2004.

[8] J. P. Anderson, *Computer Security Threat Monitoring and Surveil-lance*. Fort Washington, PA: Anderson, 1980.

[9] A. Bivens, C. Palagiri, R. Smith, B. Szymanski, and M. Embrechts, "Network-based intrusion detection using neural networks," in *Proc. ANNIE*, St. Louis, MO, Nov. 2002, pp.

[10] "Smurf IP denial-of-service hacks," CERT Advisory CA-1998-01, 1998 [Online]. Available: http://www.cert.org/advisories/CA-1998-01. html

[11] "Pingofdeath,"1997[Online]. Available: http://insecure.org/sploits/ ping-o-death.html

[12] B. Canvel, A. Hiltgen, S. Vaudenay, and M. Vuagnoux, "Password interception in a SSL/TLS channel," in *Proc. Crypto 2003*, Santa Barbara, CA, Feb. 2003, vol. 2729, LNCS, pp. 583–599.