International Journal of Cryptology Research 2(1): 89-99 (2010)

Detecting Attacks in Encrypted Networks using Secret-sharing Schemes

[1]Vik Tor Goh, [2]Jacob Zimmermann and [3]Mark Looi
Information Security Institute
Queensland University of Technology
Queensland, Australia
Email: [1]v.goh@qut.edu.au, [2] j.zimmerm@isi.qut.edu.au and [3]m.looi@qut.edu.au

ABSTRACT

Secret-sharing schemes describe methods to securely share a secret among a group of participants. A properly constructed secret-sharing scheme guarantees that the share belonging to one participant does not reveal anything about the shares of others or even the secret itself. Besides being used to distribute a secret, secret-sharing schemes have also been used in secure multi-party computations and redundant residue number systems for error correction codes. In this paper, we propose that the secret-sharing scheme be used as a primitive in a Network-based Intrusion Detection System (NIDS) to detect attacks in encrypted networks. Encrypted networks such as Virtual Private Networks (VPNs) fully encrypt network traffic which can include both malicious and non-malicious traffic. Traditional NIDS cannot monitor such encrypted traffic. We therefore describe how our work uses a combination of Shamir's secret-sharing scheme and randomised network proxies to enable a traditional NIDS to function normally in a VPN environment.

INTRODUCTION

Secret-sharing schemes such as those initially proposed by Shamir [1] and Blakely [2] are used to securely distribute a secret among a group of participants. A (k, n) secret-sharing scheme is one where a single secret is split into n shares and any k shares is sufficient to recover the original secret where $k < n$. Secret-sharing has been used as the primitive of multi-party computations [3] and error correction codes [4]. In our work, we apply secret-sharing in a novel way to detect attacks initiated over encrypted networks.

Intrusion detection systems (IDSs) are based on the notion that computer misuse can be detected by analysing audit data in computer system or network. Audit data can be in the form of log files, file statuses and even network traffic. A host-based IDS (HIDS) operates within a host, monitoring

log files and system calls for signs of potential abuse while a network-based IDS (NIDS) monitors network traffic for malicious activities.

However, if the audit data cannot be accessed due to data corruption or encryption, the IDS cannot function properly. Our work focuses on the effects of encrypted networks on NIDS operations. We are motivated by the growing use of encrypted networks that obfuscate network traffic between any two hosts in the network. An encrypted network such as a virtual private network (VPN) encrypts the entire network, thus completely masking traffic – malicious or otherwise. This can negate the protection offered by a NIDS because it cannot analyse encrypted traffic.

In this paper, we propose a novel method of using secret-sharing to allow a NIDS to function properly in a VPN. It does this without compromising on the confidentiality and integrity afforded by the VPN infrastructure.

RELATED WORK

The most common approach of enabling intrusion detection in VPNs is to co-locate a NIDS on the VPN gateway. In this setup, the VPN traffic from an external host is terminated at the VPN gateway located at the network perimeter. At this point, network traffic is decrypted by the VPN gateway and subsequently given to the co-located NIDS for analysis. If the NIDS detects benign traffic, it allows the traffic to pass to the internal host in cleartext form. If the traffic is malicious, an alert will be triggered and the malicious traffic is dropped. Wagner [5] proposed such an approach. Despite that, this approach does not support an end-to-end VPN tunnel, established from the external host right up to the internal host. The end-to-end tunnel offers the greatest amount of confidentiality.

A variant of the co-located NIDS approach is the man-in-the-middle approach that supports end-to-end VPN tunnels. Yamada et al. [6] suggested that a VPN gateway with a co-located NIDS makes two separate VPN tunnels – one with a host on the left and another with the host on the right. With the gateway sitting in between the two communicating hosts, it is able to relay network traffic between them with each host believing that it is communicating directly with the other. It relays network traffic while analysing the network traffic that has been decrypted at the gateway before encrypting it again and sending it to the other host. This approach requires the private keys of both hosts and implies the use of a complicated Public Key Infrastructure (PKI).

To the best of our knowledge, there are three other approaches that attempt to address this problem of detecting attacks over VPNs. The first approach uses statistical traffic analysis techniques to detect intrusions. Encrypted network traffic is extracted using network sniffers and then analysed to infer information from frequently observed patterns. Although network traffic is encrypted, there are still some outwardly observable features such as frequency and size of packets. These features can reveal certain trends and patterns that uniquely characterise certain attacks. Yamada *et al.* [6], Piccitto *et al.* [7] and Foroushani *et al.* [8] use statistical analysis to infer the presence of malicious activities in SSL and VPN traffic. Statistical analysis is nevertheless limited in scope due to the few traffic patterns that can be practically deduced solely from observing the network traffic. Moreover, a large volume of network traffic has to be collected and analysed before any obvious trends begin to emerge.

The second approach detects misuse of network encryption protocols such as Secure Sockets Layer (SSL) and IP Security (IPsec). A misuse is defined as a case where the internal states of these protocols do not transition from one state into another in an expected and predictable manner. Md. Fadlullah *et al.* [9] and Joglekar and Tate [10] proposed that whenever there is an unexpected transition, their intrusion detection systems assume that the encryption protocol is being misused and exploited for ulterior purposes. For the intrusion detection system to correctly identify such misuse, it must first have an accurate specification of a legitimately working encryption protocol. Any deviation from this specification is considered as an attack. However, the task of defining the specification is not trivial because every possible legitimate state must be modeled correctly.

Both these approaches are suitable for detecting attacks in encrypted networks because they do not directly analyse the payload contained within the encrypted packet. Instead, they monitor external features for specific trends that are symptomatic of attacks. Despite that, an attack can evade detection if it complies exactly with regular traffic. This is typical of application-level attacks such as SQL injection and buffer overflow attacks where the malicious codes are contained within the payload portion of the encrypted packet. Only a few network packets are needed to corrupt the target and as such is statistically insignificant.

The third approach addresses the inability to detect application-level attack by using a framework that retrieves decrypted network traffic for deep packet inspection. A NIDS that performs deep packet inspections examines the payload portion of the packet. Abimbola *et al.* [11] and Wang *et al.* [12]

proposed a framework where customised programs are installed into every network host. The program is able to extract already decrypted network traffic from the host and feed that into a NIDS for analysis. Although this approach recovers the decrypted payload for deep packet inspection, it does not consider the fact that the program residing in the targeted host can be defeated if the host is compromised.

Although a HIDS could have been used to detect attacks propagating through VPNs, it has the added overhead of being intrinsically complex. A HIDS has to continuously monitor many distinct aspects of a host such as system calls and application logs to function effectively. Moreover, a HIDS has a local view of malicious activities and does not correlate with other HIDS to gauge the severity of the attack at the network level.

We thus propose a detection framework that allows the use of deep packet inspection to detect attacks in VPN. The framework combines the use of two well-established technologies together, namely NIDS and VPN. This framework does not compromise on confidentiality and allows the establishment of end-to-end VPN tunnels. It also does not require a complicated PKI.

OVERVIEW OF INTRUSION DETECTION SYSTEM

A traditional NIDS extracts network traffic for analysis using network sniffers. However, if the network is encrypted using a VPN, the NIDS is rendered useless. As such, our approach uses a host-based approach where a custom network interface driver is installed on every host. The network interface driver is part of the detection framework that relies on a protocol that replicates network traffic from a sender and forwards that replicated traffic to both the NIDS and receiver. In other words, a copy of all network traffic is explicitly forwarded to the NIDS. This is achieved while preserving the privacy and integrity of the communication exchange.

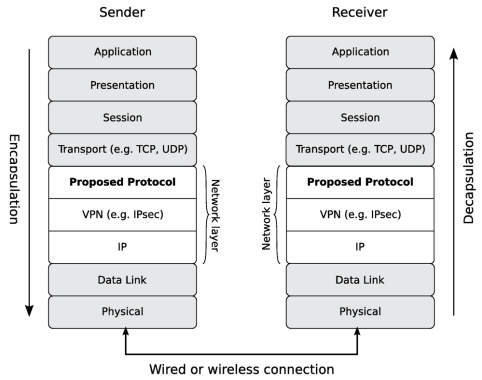

Figure 1: OSI network model and the proposed detection protocol.

The privacy and integrity of the network traffic is maintained because the proposed protocol functions on top of an underlying VPN infrastructure. In a VPN-enabled network, every link between network peers is encrypted by the VPN's cryptographic algorithms. In Figure **1**, we see that the proposed protocol is the topmost protocol in the network layer of the OSI model. Since it resides above VPN protocols such as IPsec, all network traffic that has been processed by the protocol is subsequently encapsulated by IPsec's confidentiality and integrity algorithms. This architecture prevents unwanted eavesdropping or tampering by unauthorised third-parties while the payload is en route to its destination. More importantly, this architecture does not require that the VPN framework be modified in any way to accommodate the proposed protocol.

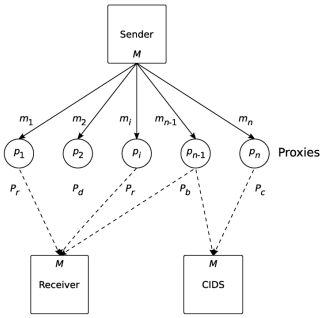

Figure 2: Interactions between components in proposed detection protocol.

In our work, we propose that a Central IDS (CIDS) be used together with forwarding proxies as shown in Figure **2**: Interactions between components in proposed detection protocol.. The CIDS is a traditional NIDS that performs traffic analysis and intrusion detection. The CIDS operates as a separate and dedicated host in the network. If a sender wishes to send network packets to a receiver, it will do so via the forwarding proxies. This traffic routing is achieved using the custom network interface driver that has been installed in the sender. The forwarding proxies in turn ensure that the network traffic is forwarded to the receiver and CIDS. Our protocol is summarised by the following principle:

All traffic sent to a receiver by a sender must be replicated and forwarded also to the NIDS, without the possibility of the sender withholding traffic from the NIDS or forging fake traffic, and while maintaining the confidentiality and integrity of the encrypted network.

Although the VPN infrastructure prevents unauthorised network sniffing, the forwarding proxies are able to access the network packets relayed through them. This exposes the network packets to unwanted scrutiny and possible tampering while transiting through the proxies. To ensure confidentiality with respect to the proxies, we use secret-sharing. Each original packet is split into its corresponding shares and it is these shares that are relayed through the proxies. The use of secret-sharing thus ensures that the packets remain confidential without the need for a PKI.

Secret-sharing Scheme

Our work uses Shamir's secret-sharing scheme [1]. Without loss of generality, a (k, n) secret-sharing scheme represents a secret S as a set of n shares where S = $\{s_1, s_i, ..., s_n\}$ and $k < n$. Knowledge of any k or more s_i recovers S while knowledge of any k - 1 or less reveals nothing. If some of the shares are incorrect, the original secret cannot be recovered; that is $\{s_1', s_i', ..., s_n\} \neq$ S. Confidentiality is thus an inherent feature of a secret-sharing scheme if and only if the unauthorised party or colluding parties do not have k or more shares.

We use Shamir's secret-sharing scheme because it is straightforward to implement and is not computationally too intensive. The secret-splitting process involves evaluating the polynomial at n different points. Using the Horner algorithm, the polynomial can be efficiently evaluated as a series of linear operations. This reduces the splitting process to a complexity of $O(n)$ [13]. For the secret-recovery process, the Lagrange form of the interpolation

polynomial is known to be quadratic with a complexity of $O(n^2)$ [14]. Secret-sharing also has the property where the size of each share does not exceed the size of the original secret. This is an important property for our protocol because it ensures that each share is bounded in size and suitable for transmission over the network.

Protocol Description

If a sender wishes to send a message (i.e. network packet) to the receiver, the sender first splits the message into its corresponding shares using the secret-sharing scheme. So, if the message to be sent is M, its corresponding shares are $M = \{m_1, m_i, \ldots, m_n\}$. The sender then sends all n shares to n proxies. After receiving its share, each proxy randomly chooses where to forward the share to. There are four possible destinations:

- Forward to the intended receiver with probability P_r;
- Forward to the CIDS with probability P_c;
- Forward to both the CIDS and receiver with probability P_b; or
- Drop the share with probability P_d.

The receiver and CIDS can recover the message M as long as they each receive at least k distinct shares. The algorithm that implements the protocol is shown in

TABLE **1**.

TABLE 1: Algorithm for sending and receiving network packets

Algorithm
1. Sender splits packet M into corresponding shares of $\{m_1, m_i, \ldots, m_n\}$ using a (k, n) secret-sharing scheme; 2. Sender selects n proxies from a pool of N proxies where $n < N$; 3. Each m_i is sent to one of the n forwarding proxies, p_i; 4. Proxy p_i does one of the four predefined actions; • Forward to receiver only with probability P_r; • Forward to CIDS only with probability P_c; • Forward to both CIDS and receiver with probability P_b; or • Drop the share with probability P_d. 5. If the receiver receives k or more m_i's, it recovers the packet M; and 6. If the CIDS receives k or more m_i's, it recovers packet M for analysis.

We now consider the case of a sophisticated attacker who is able to manipulate the protocol. An attacker could evade detection by

simultaneously sending two different types of messages; M to the target while M' to the CIDS. This is done with the intention of misleading the CIDS with forged but harmless traffic M' while the receiver receives the malicious traffic M.

For the sake of clarity, let M be the malicious traffic and M' be the forged but harmless traffic. They are represented by $M = \{m_1, m_i, ..., m_\alpha\}$ and $M' = \{m_1', m_j', ..., m_\beta'\}$ where $\alpha + \beta \leq n$ and $\alpha, \beta \geq k$. The attacker sends both M and M' to the proxies in the hopes that the proxies will forward m_i only to the receiver and the CIDS only receives m_j'.

Since the actions of the proxies are *a-priori* unpredictable the attacker will not know beforehand which of the n proxies will forward to whom (CIDS, receiver or both). It is therefore impossible for the attacker to reliably select which proxies should receive shares of M and which should receive M' in the hopes of obtaining a favourable outcome for the attacker. With this uncertainty, the CIDS may receive k or more shares that resolve to one of the following cases with the use of Shamir's secret-sharing:

1. Receives only m_i shares and recovers M;
2. Receives only m_i' shares and recovers M'; or
3. Receives a mixture of both m_i and m_j' shares. This case results in a corrupted message, C.

Because of the randomly behaving proxies, it is with high probability that the shares will arrive mixed up, resulting in the corrupted message C. When C is received, the CIDS knows that there has been an attempt to forge fake message with the intention of evading detection. Consequently, we envision two types of attacks that can be detected by our proposed approach. They are:

- **Application-level attack.** Our approach is essentially a tunneling protocol. Network traffic, malicious or otherwise, that is received will always match the original content. Therefore, attacks sent through the tunnel can be detected by a standard NIDS. In any case, these attacks pose a traditional intrusion detection problem.
- **Evasion attack.** These attacks attempt to evade CIDS detection in our context, by playing a game of chance with the random proxies. This can be detected if a message is corrupted upon delivery.

DISCUSSIONS

As shown in Figure 1, the protocol functions at the network layer. Just like other network layer protocols, our proposed protocol only provides best-effort packet delivery. If reliable message delivery is required, a suitable transport layer protocol such as TCP can be used to encapsulate the message before handing it off to our protocol. TCP has built-in mechanisms that handle packet losses through retransmissions.

An attacker could possibly bypass the forwarding proxies entirely and send its payload directly to the receiver. In an initially "clean" network, it is reasonable for a receiver to expect for the sender's payload to arrive through the proxies. If the receiver receives the payload directly from the sender, it will ignore this non-compliant flow. Source address spoofing to fool the receiver into believing that it is receiving the payload from the proxies is also not possible. Peer authentication mechanism of the underlying VPN protocol ensures that the host is who it claims to be – sender or proxies [15]. As such, a malicious sender will have to comply with the protocol if it wants its payload to be properly delivered.

CONCLUSIONS

In this work, we introduce a detection framework that allows a traditional NIDS to perform deep packet inspection in an encrypted network traffic. The framework relies on a protocol that ensures all network traffic bound for the receiver is also sent to the NIDS for inspection. The VPN's security protocols are leveraged upon to preserve confidentiality from unauthorized third-parties while secret-sharing is used to maintain privacy from the forwarding proxies.

We foresee a growth in attacks propagating through encrypted networks in the future. Traditional NIDS will no longer be feasible to address this problem and must be adapted accordingly. We believe that our approach is one step in that direction.

ACKNOWLEDGEMENTS

This work is supported in part by funds from International Information Systems Security Certification Consortium, Inc. $(ISC)^2$.

REFERENCES

[1] Shamir, A. 1979. How to share a secretr. *Communications of the ACM*, **22**(11), 612–613.

[2] Blakley, G. R. 1979. Safeguarding cryptographic keys. In *AFIPS National Computer Conference*: 313–317.

[3] Yao, A. C. 1982. Protocols for secure computations. In *23rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 1982)*: 160–164.

[4] Yang, L. and Hanzo, L. 2001. Redundant residue number system based error correction codes. In *54th IEEE Vehicular Technology Conference (VTC 2001)*:1472–1476.

[5] Wagner , A., Dubendorfer T., Hiestand, R., Goldi, C. and Plattner, B. 2006. A fast worm scan detection tool for VPN congestion avoidance. In *3rd International Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2006)*:181–194.

[6] Yamada, A., Miyake, Y., Takemori, K., Struder, A. and Perrig, A. 2007. Intrusion detection for encrypted web access. In *21st International Conference on Advanced Information Networking and Applications Workshops (AINAW 2007)*: 569–576.

[7] Piccitto, D., Burschka, S., and Urvoy-Keller, G. 2007. *Traffic mining in IP tunnels*. Master's thesis, Eurecom Institute.

[8] Foroushani, V. A., Adibnia, F., and Hojati, E. 2008. Intrusion detection in encrypted access with SSH protocol to network public servers. In *International Conference on Computer and Communication Engineering (ICCCE 2008)*: 314–318.

[9] Md. Fadlullah, Z., Taleb, T., Ansari, N., Hashimoto, K., Miyake, Y., Nemotoi, Y. and Kato, N. 2007. Combating against attacks on encrypted protocols. In *IEEE International Conference on Communications (ICC 2007)*:1211–1216.

[10] Joglekar, S. P. and Tate, S. R. 2004. Protomon: Embedded monitors for cryptographic protocol intrusion detection and prevention, In

International Conference on Information Technology: Coding and Computing (ITCC 2004): 81–88.

[11] Abimbola A., Munoz, J. M., and Buchanan, W. J. 2006. Nethost-sensor:Investigating the capture of end-to-end encrypted intrusive data. *Computers & Security*, **25**(6): 445–451.

[12] Wang, Z., Jiang, X., Cui, W., Wang, X. and Grace, M. 2009. ReFormat: Automatic reverse engineering of encrypted messages. In *14th European Symposium on Research in Computer Security (ESORICS 2009)*: 200–215.

[13] Maurer, S. B., and Ralston, A. 2004. *Discrete algorithmic mathematics*, 3rd ed., Massachusetts, USA: A. K. Peters Ltd.

[14] Cheney, E. W. and Kincaid, D. R. 2003. *Numerical mathematics and computing*, 5th ed., California, USA: Brooks Cole.

[15] Maughan, D., Schneider, M., Schertler, M. and Turner, J. 1998. RFC 2408: Internet security association and key management protocol (ISAKMP).