ISSN: 2321-7782 (Online)
Volume 2, Issue 7, July 2014
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Procuring Vague Sessions by Exploiting Intrusion Prevention System over Web Traffic

Prajakta Suhasrao Kale[1]
Department of CSE
Shreeyash College of Engineering
Aurangabad
Maharashtra – India

Prof. S. M. Tidke[2]
Department of CSE
Shreeyash College of Engineering
Aurangabad
Maharashtra – India

Abstract: ***To a huge degree traffic travels on the network every minute a day. As the traffic enlarges a lot of signatures get definite. Several traffic cataracts into category of 'evident' and another into 'vague'. Obvious is what network waits for and healthily transmits but vague will be causing harm to network. When it comes to distinguishing between evident and vague traffic, a group of packets has to pass through an Intrusion Prevention System and on the basis of that whatever packets moves towards right purpose moving intended information and the packets causing damage to the network can be eminent. So wise False Positives and False Negatives occurs to every IPSs. None of the scheme could judge better all the time. The key purpose here is to plan Vague Session Fetching (VSF) System to obtain vague sessions as whole and ideal as possible which gives IPS developers resources for further investigation. First, the VSF captures real traffic and reruns captured traffic traces to various IPSec then create to fetch vague traffic from replayed traffic traces and then prove and authenticate it. IPS developers can more examine the fetched traffic traces and verify that a few are FNs or FPs. To entirely and perfectly fetch an vague session, the VSF uses an association means VSF on anchor packets, five tuples and time, and similarity for the first packet, first connection, and whole session, respectively. It computes the degree of resemblance among packets to obtain an vague session holding multiple connections. I defined variation and entireness/perfectness as the indexes to appraise the VSF. The testing demonstrates that 97% of fetched sessions have low variation, and the average entireness/perfectness is around 85%. Also presented four VSF studies, First is a P-FN and the other is a P-FP, and second is a N-FN and the other is a N-FP found by the VSF and confirmed by the IPS developers.***

Keywords: IPS, Vague, evident Sessions, entireness, perfectness, P-FN, P-FP, N-FN, N-FP.

I. INTRODUCTION

It is a tricky job for Intrusion Prevention System to distinguish among the traffic fleeting on the network on the basis of whether the traffic is obvious or vague. When the traffic whatever is expected touching towards intended path will be declining into the category of Obvious and the one which is causing damage to the network will be a Vague one. It is hard for an Intrusion Prevention System to differentiate between them so an plentiful amount of traffic has to pass during the IPS for adequately extensive amount of instance. Then on the basis of that patterns/signatures there should be differentiation made so that damaging traffic can be prevented .For some of the IPSes same sessions can be obvious and vague.

We design a system of vague session Fetching (VSF) to create a pool of vague traffic traces. The VSF captures, replays, and fetches real traffic. First, it captures real traffic from the mirror port of an Ethernet switch. The captured traffic traces are generally in the PCAP format and contain packets from a variety of applications such as FTP, Web. Second, it replays traffic traces to IPSes of different vendors and trigger them to generate logs. From the logs, we can determine that some attack logs are generated or not generated only at a certain IPS. The former are P-FPs, while the latter are P-FNs to that IPS. The VSF system fetches vague traffic and offers fetched traffic traces to developers for enhancing the correctness of their IPSes.

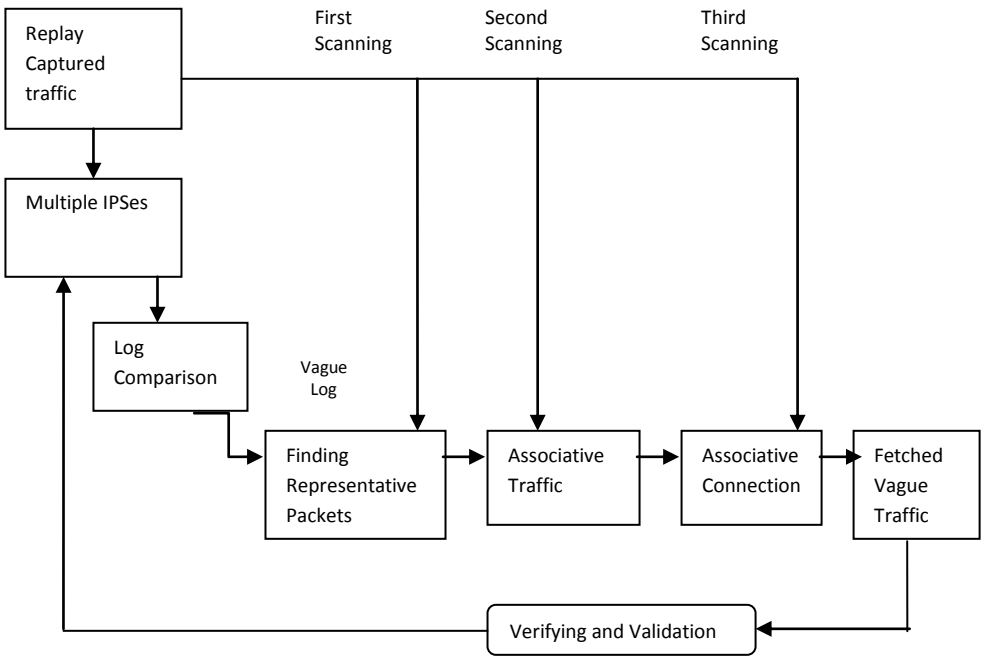
Our major anxiety for traffic fetching is entireness. IPS developers generally need an whole session for exact analysis. The vague traffic trace must hold not only packets that can trigger an FN or FP, but also the complete session that the packets belong to. The fetching takes some clues to vague traffic from IPS logs, such as source IP address, source port number, destination IP address, destination port number, transport layer protocol and time. According to the clues, the VSF can find representative packets that trigger the FN or FP, i.e., the critical/essential packets that can make IPS generate logs. Without any one of the representative packets, IPSes would not generate the log. The VSF then associates the other packets with the representative packets if they belong to the same TCP or UDP connection. However, completely fetching a connection is still inadequate, since a session could consist of multiple connections and IPSes do not log all the connections in that session. For example, most IPSes log only the first connection of a DDOS attack session to evade information bang in the log system. Associating related connections is required for the VSF.

The remainder of this paper is organized as follows: in Section 2, we discuss background knowledge and related work. Section 3 makes a detailed description of the design and implementation of the VSF. In Section 4, we evaluate the VSF with two indexes and discuss the results. Four cVSF studies of FN and FP are illustrated in Section 5. We conclude with Section.

II. BACKGROUND

2.1 Fetching Vague Sessions

In this work, we capture and replay real traffic to IPSes, and look for some clues in the IPS logs to help identify Vague sessions. The steps of capture and replay have been adopted for performance evaluation of IPSes. Open source tools such as Tcpdump (www.tcpdump.org) and Tcpreplay (tcpreplay.sourceforge.net) can be used to capture and replay real traffic, respectively. Wireshark also helped to guide about Capture, The former captures real traffic into files in the PCAP format as traffic traces, while the latter replays the traffic traces to the IPSes. However, the volume of traffic traces may be huge, and it is non-trivial to fetch an vague session involving multiple connections from the traces. This work designs a method to fetch an vague session bVSFd on representative packets, five tuples and time, and similarity.



2.2 Wireshark

❖ 2.2.1. What is Wireshark?

Wireshark is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible.

You could think of a network packet analyzer as a measuring device used to examine what's going on inside a network cable, just like a voltmeter is used by an electrician to examine what's going on inside an electric cable (but at a higher level, of course).

In the past, such tools were either very expensive, proprietary, or both. However, with the advent of Wireshark, all that has changed.

Wireshark is perhaps one of the best open source packet analyzers available today.

❖ 2.2.2. Some intended purposes

Here are some examples people use Wireshark for:

- Network administrators use it to troubleshoot network problems
- Network security engineers use it to examine security problems
- Developers use it to debug protocol implementations
- People use it to learn network protocol internals Beside these examples, Wireshark can be helpful in many
- Other situations too.

❖ 2.2.3. Features

The following are some of the many features Wireshark provides:

- Available for UNIX and Windows.
- Capture live packet data from a network interface.
- Open files containing packet data captured with tcpdump/WinDump, Wireshark, and a number of other packet capture programs.
- Import packets from text files containing hex dumps of packet data.
- Display packets with very detailed protocol information.
- Save packet data captured.
- Export some or all packets in a number of capture file formats.
- Filter packets on many criteria.
- Search for packets on many criteria.
- Colorize packet display bVSFd on filters.
- Create various statistics.

❖ 2.2.4. Live capture from many different network media. Wireshark can capture traffic from many different network media types - and despite its name - including wireless LAN as well. Which media types are supported, depends on many things like the operating system you are using.

❖ 2.2.5. Import files from many other capture programsWireshark can open packets captured from a large number of other capture programs.

❖ 2.2.6. Export files for many other capture programsWireshark can save packets captured in a large number of formats of other capture programs.

❖ 2.2.7. Many protocol decoders

There are protocol decoders (or dissectors, as they are known in Wireshark) for a great many protocols.

❖ 2.2.8. Open Source Software

Wireshark is an open source software project, and is releVSFd under the GNU General Public License (GPL). You can freely use Wireshark on any number of computers you like, without worrying about license keys or fees or such. In addition, all source code is freely available under the GPL. Because of that, it is very easy for people to add new protocols to Wireshark, either as plugins, or built into the source, and they often do!

❖ 2.2.9. What Wireshark is not

Here are some things Wireshark does not provide:

- Wireshark isn't an intrusion detection system. It will not warn you when someone does strange things on your network that he/she isn't allowed to do. However, if strange things happen, Wireshark might help you figure out what is really going on.
- Wireshark will not manipulate things on the network, it will only "measure" things from it. Wireshark doesn't send packets on the network or do other active things (except for name resolutions, but even that can be disable.

2.3 Types of attack:

Classes of attack might include passive monitoring of communications, active network attacks, close-in attacks, exploitation by insiders, and attacks through the service provider. Information systems and networks offer attractive targets and should be resistant to attack from the full range of threat agents, from hackers to nation-states. A system must be able to limit damage and recover rapidly when attacks occur. There are four types of attack:

1) Passive Attack

A passive attack monitors unencrypted traffic and looks for clear-text passwords and sensitive information that can be used in other types of attacks. Passive attacks include traffic analysis, monitoring of unprotected communications, decrypting weakly encrypted traffic, and capturing authentication information such as passwords. Passive interception of network operations enables adversaries to see upcoming actions. Passive attacks result in the disclosure of information or data files to an attacker without the consent or knowledge of the user.

2) Active Attack

In an active attack, the attacker tries to bypass or break into secured systems. This can be done through stealth, viruses, worms, or Trojan horses. Active attacks include attempts to circumvent or break protection features, to introduce malicious code, and to steal or modify information. These attacks are mounted against a network backbone, exploit information in transit, electronically penetrate an enclave, or attack an authorized remote user during an attempt to connect to an enclave. Active attacks result in the disclosure or dissemination of data files, DoS, or modification of data.

3) Distributed Attack

A distributed attack requires that the adversary introduce code, such as a Trojan horse or back-door program, to a “trusted” component or software that will later be distributed to many other companies and users Distribution attacks focus on the malicious modification of hardware or software at the factory or during distribution. These attacks introduce malicious code such as a back door to a product to gain unauthorized access to information or to a system function at a later date.

4) Insider Attack

An insider attack involves someone from the inside, such as a disgruntled employee, attacking the network Insider attacks can be malicious or no malicious. Malicious insiders intentionally eavesdrop, steal, or damage information; use information in a fraudulent manner; or deny access to other authorized users. No malicious attacks typically result from carelessness, lack of knowledge, or intentional circumvention of security for such reasons as performing a task.

III. THE SYSTEM OF VAGUE SESSION FETCHING (VSF)

Figure 1 illustrates the steps in the VSF system. After re-playing the traffic traces to multiple IPSes and the voting scheme, the system only knows which packets trigger the logs, but it is still necessary to extract the entire session for helping the IPS developers identify an FP or FN pre- cisely. The session extraction involves three-pass scanning through the traffic traces: (1) finding the anchor packets that trigger the P-FP or P-FN from the five-tuple information in the IPS logs, (2) associating the other packets with the anchor packets if they belong to the same TCP or UDP connection, and (3) associating the other connections with the connection which determine a time frame. The system then just needs to search for the anchor packets within the same time frame in the traffic trace.

1. Replaying traffic traces and voting

In this step, the VSF system leverages the knowledge in signature databVSFs on commercial IPSes and the open-source Snort IPS [19] to find ambiguous traffic traces. The system replays captured traces to multiple IPSes, and then compares the logs among them in the voting scheme to find out ambiguous logs, i.e., those only generated (P-FP) or not generated (P-FN) at a certain IPS. These logs will be correlated with the packets to help the session extraction.

2. Finding out anchor packets (the first pass)

This step finds out anchor packets in the ambiguous traces, i.e., the critical/essential packets that can trigger IPS logs. For this purpose, we implement an Alarm Log Table (ALT) to record alarm logs from IPSes, and a Replay Log Table (RLT) to record the time of each packet sent from Tcpreplay. The two tables can keep the useful information for correlating the logs and packets. Matching the five tuples should be sufficient to identify the anchor packets, but unfortunately, some IPSes do not log the complete five tuples of packets, e.g., only the IP addresses of both ends. The VSF system therefore needs to match the time information in both tables. The time of a log entry in the ALT is correlated with that in the RLT to associated with the anchor packets to be in the same connection.

3. Packet Association (the second pass)

This step looks for all the other packets that have the same five tuples as the anchor packets. Those packets are the the anchor packets belong to, if they belong to the same session. The sessions associated with the anchor packets are therefore extracted in the scanning. They are replayed to the IPSes again to verify the correctness of extraction

4. Connection Association (the third pass)

An ambiguous session with multiple attacking sources, it is insufficient to only depend on the five tuples and timestamp of packets. We design a session association algorithm bVSFd on the observation that such an ambiguous session often consists of only the TCP ACK or SYN segments, as well as a number of packets mostly having the same data payload. After finding the anchor packets of an ambiguous session, the algorithm checks each subsequent packet to see if its source IP address or destination IP address is identical to the victim's IP address of the anchor packet. If not, the packet will be considered not belonging to this attack; otherwise, the algorithm will continue to compare the payload of each packet that may belong to this attack for similarity. If the similarity is high for a packet, the algorithm will duplicate its copy in the DDoS attack buffer for later judgment.

The similarity is defined according to the longest common subsequence (LCS) [20] of two packet payloads. Given a sequence X = (x1, x2, . . . , xm), a sequence Z = (z1, z2, . . . , zm) is a subsequence of X if there is a strictly increasing sequence (i1, i2, . . . , ik) of indices of X, such that xij = zj for all j = 1, 2, . . . , k. Given two sequences X and Y , Z is a common subsequence of X and Y if Z is a subsequence of both X and Y . The longest common subsequence is the longest one of the all common subsequences. Consider the payloads of two packets as two sequences of bytes, S1 and S2. LCS(S1, S2) denotes the longest sequence of bytes that are subsequences of S1 and S2. The similarity between two packet payloads is defined by

$$\text{Similarity}(S1,S2) = \frac{2 * |LCS(S1,S2)|}{|S1| + |S2|} * 100$$

The similarity threshold is 80% in the proposed algorithm because the packets collected from the DDoS attacks are often minimum Ethernet packets of 64 bytes. Excluding the 14-byte MAC header, 20-byte IP header, 20-byte TCP header and 4-byte checksum, the remaining payload is only 6 bytes long. We observe the packet payloads of the DDoS or DoS attacks we collected are often the same, and the difference is only one byte if the payloads are different. The similarity in this cVSF is 83.33%. This work therefore sets the similarity threshold to 80%. After identifying similar packets, the session association algorithm watches the source IP address and the destination IP address at the same time. This step stores only the packets between the attacker and the victim, and drops the other packets. This step also distinguishes the attacks that have probably one attacker from those that are probably DDoS attacks. The algorithm keeps watching the subsequent packets, and returns the packet count in the DDoS attack buffer. The ambiguous session might be a DDoS attack if the count is larger than 200; otherwise the attack type is 1-1 or 1-N. An anchor packet A triggers an log on an IPS. The problem, consequently, turns into looking for the packets having the payload highly similar to A's and the packets having the same source IP address or destination IP address as A's. Figure 2 lists the session association algorithm, and Table 2 summarizes the notations in the algorithm. The five tuples of the packet Pi is defined by

Extracting ambiguous sessions and verifying Finally, we replay the extracted ambiguous sessions to the IPSes to verify the correctness of the extraction. The extracted sessions should trigger exactly the same alarm logs as the whole traffic does. If an IPS product does not generate the same logs, the extraction is invalid.

IV. Study of False/True positives/Negatives

1. Trues are generally a good thing. Either a signature fired if offending traffic was detected or it did not fire for regular, non-malicious traffic.
2. Falses, on the other hand, are not a good thing. Because a signature did not fire when it should have or did fire when it should not.

The true/false can be seen as the whether the item was correctly/incorrectly identified and the positive/negative can be seen as whether the correct/incorrect action was taken.

	True(T)	False(F)
Positive(P)	*Alarm: Yes* *Attack: Yes*	*Alarm: Yes* *Attack: No*
Negative(N)	*Alarm: No* *Attack: No*	*Alarm: No* *Attack: Yes*

Table 1. Understanding False/True Positives/Negatives

True Positive: True positive is a good thing. It is an event when an intrusion did happen and the IPS did react on it. This is a normal thing. So, a legitimate attack which triggers an IDS to produce an alarm.

False Positive: False positive is "kind-a" bad thing. It is not an intrusion but rather a situation when normal user traffic triggers an alarm. This can cause a lot of log entries or even drop normal user traffic. This is NOT normal. So, an event signaling an IDS

to produce an alarm when no attack has taken place.

False Negative: False negative is the worst cVSF. This is a situation in which an intrusion did happen and IPS totally missed it! This is also NOT normal. So, A failure of an IDS to detect an actual attack.

True Negative: True negative is even better. No intrusion happened and no action is taken by an IPS. This represents a normal traffic flow. This is also normal. So, when no attack has taken place and no alarm is raised.

V. RESULT ANALYSIS

5.1 Attacks Detected

1. Multi Tab / Multi Browser Attack

A unique session ID is created for every session so that whenever the person is logged in system is not allowing other person to login in another tab and also not even in another Browser

2. Bruit force attack

In cryptography, a brute-force attack, or exhaustive key search, is a cryptanalytic attack that can, in theory, be used against any encrypted data (except for data encrypted in an information-theoretically secure manner). Such an attack might be utilized when it is not possible to take advantage of other weaknesses in an encryption system (if any exist) that would make the task easier. It consists of systematically checking all possible keys or passwords until the correct one is found. In the worst cVSF, this would involve traversing the entire search space. When password guessing, this method is very fast when used to check all short passwords, but for longer passwords other methods such as the dictionary attack are used because of the time a brute-force search takes. When key guessing, the key length used in the cipher determines the practical feasibility of performing a brute-force attack, with longer keys exponentially more difficult to crack than shorter ones. A cipher with a key length of N bits can be broken in a worst VSF time proportional to 2^N and an average time of half that. Brute-force attacks can be made less effective by obfuscating the data to be encoded, something that makes it more difficult for an attacker to recognize when he/she has cracked the code. One of the measures of the strength of an encryption system is how long it would theoretically take an attacker to mount a successful brute-force attack against it. Brute-force attacks are an application of brute-force search, the general problem-solving technique of enumerating all candidates and checking each one. For the purpose when user types incorrect password continuously four times fifth time account is locked and account unlock url is sent to registered email.

3. Locked-Account-Login Try Attack

Locked-Account-Login Try Attack is used to unlock the account blocked by Bruit force attack .In this lock key generated by brute force attack is removed to login again.

4. *Location Attack*

Time and Distance parameters for Login are exactly verified

A unique session ID is created for every session so that whenever the person is logged in system is not allowing other person to login. If the person is login from one place and within required time if it is possible for logging at some distance place then only the system will allow the user to login otherwise it will reflect that the session is Vague.

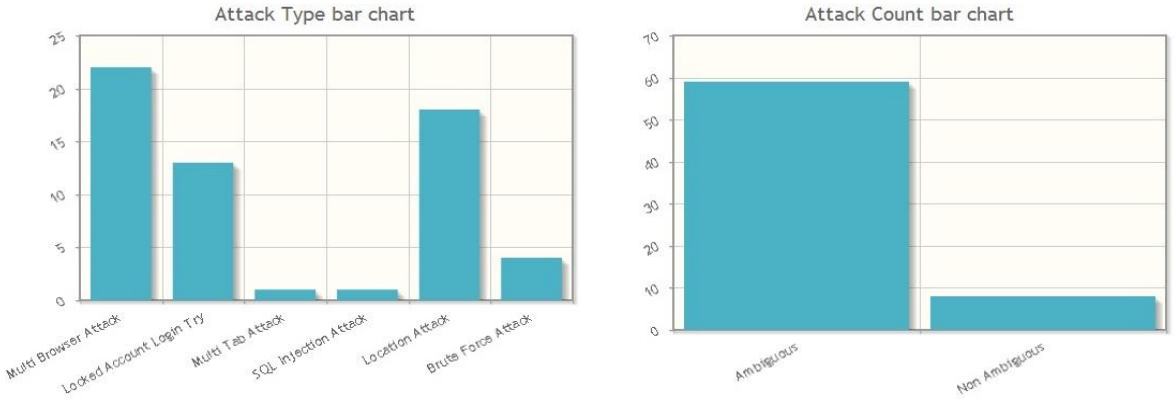


Figure 5.1. Attack Type bar Chart

Attack Name	Attacks Count
Multi Browser Attack	22
Locked Account Login Try Attack	13
Location Attack	18
Brute Force Attack	4

Table: Results according to Attacks detected

VI. CONCLUSION

This work proposes an VSF system to fetch vague sessions from real traffic as entire and perfect as possible. The vague sessions cause P-FPs or P-FNs to an IPS and can be used for further analysis by IPS developers to improve the accuracy of an IPS. The system associates related packets with an attack by scanning traffic traces three times — to identify the representative packets, the packets in the same connection and the connections in the same session. For connection association in a session, similarity between two packets is defined to extract an ambiguous session of the N-1 type. We define variation and entireness/perfectness to evaluate the VSF. 97% of the attack traces have low variation. Also, the average completeness/purity is around 85%. This method could be extended to other detection systems such as anti-virus and P2P management.

References

1. False Positives and Negatives from Real Traffic with Intrusion Detection/Prevention Systems Cheng-Yuan Ho1, Ying-Dar Lin1-Cheng Lai2,I-Wei Chen1, Fu-Yu Wang1 and Wei-Hsuan Tai1 ,National Chiao Tung University, Taiwan , National Taiwan University of Science and Technology, Taiwan
2. T. Bhaskar, N. Kamath B and S. D. Moitra, "A hybrid model for network security systems: Integrating intrusion detection system with survivability," International Journal of Network Security, vol.7, no.2, pp. 249–260, Sept. 2008.
3. J. Zeng and D. Guo, "Agent-bVSFd intrusion detection for network-bVSFd application," International Journal of Network Security, vol.8, no.3, pp. 201–210, May 2009.
4. P. Kabiri and A. A. Ghorbani, "A rule-bVSFd temporal alert correlation system," International Journal of Network Security, vol.5, no.1, pp. 66–72, July 2007.
5. Virustotal, http://www.virustotal.com.
6. G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee, "Bothunter: Detecting malware infection through IDS-driven dialog correlation," Proc. of the16th USENIX Security Symposium, Aug. 2006.
7. Anti Phishing Working Group. \Phishing Activity Trends Report (3rd Quarter 2009)www.antiphishing.org/reports/ apwg_report_Q3_2009.pdf

AUTHOR(S) PROFILE

Ms. Prajakta Suhasrao Kale PG Student Department of CSE Shreeyash College of Engineering, Aurangabad, Maharashtra, India

