Once the attacker wants to copy a particular file, he can use the SMB functionalities to remote copy it with *"copy \\IP\C$\PATH."*. At traffic level, thus such operation can be detected looking for SMB *ReadAndXRequest* and hence the state transition can be modeled as follow:

Transition from State#1 to State#1.e:
Ev.A
where:
- EventA: SVCCTL ReadAndXReques (SVCCTL Command="0x2e")
- Action: Pass to State 1

Finally, once the attacker has concluded his malicious operations, ha can decide to terminate the SMB session, using the *LogOff AndX Response* command. The state transition can hence be modeled as follows:

Transition from State#1 to State#1.f:
Ev.A ***AND*** *Cond B*
where:
- Event A: SMB LogOff AndX Response (SMB Command="0x74")
- Cond. B: Request packet (SMB ResponseFlag="0")
- Action: Pass to State 0

The evolution of the states is tracked inside a dedicated *Look-Up Table*, which is flushed after a timeout value (configurable for each state). In Figure 4 such timeout transitions are reported as dashed lines.

4.1 Collaborative centralized detection

Many attackers exploit the IP spoofing technique, which consists in sending packets with a false source IP, in order to hide the identity of the sender (e.g. *anonymize* the attacker) or impersonate an host internal to the network in order to evade security control (i.e. a firewall or an IDS).

In order to demonstrate our approach, we identified two attacks, represented in Figure 6, which are based on the following techniques: decoy scan and spoofed Denial Of Service (DOS). In the first case, the attacker aims at scanning a victim host without revealing the attacking source; in order to be anonymized, the attacker spoofs IP decoys (called *zombies*), which are internal hosts reachable by the victim. The attacker is then able to scan the victim hosts without revealing its real identity (i.e. IP address) since the scan appears to be originated from different hosts supposed to be in the trusted victim's network. In the second case, the attacker aims at flooding a victim host by high rate connections, to make it unable to process other legitimate requests. By spoofing the victim's IP address, the attacker can conceal identity and hence the victim cannot identify the real attack sources in order to block the attack.

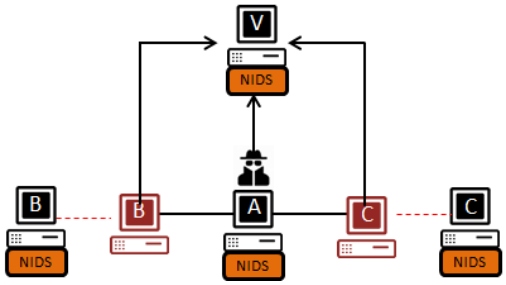

Figure 5. Decoy scan and spoofed DOS scenario

To successfully detect such attacks, it is required to differentiate the modeling of the signatures for each one of the detection levels. At host level, NIDS agents are able to perform a preliminary detection that reveals the specific protocol/technique exploited by the attacker and alert the upper layers. As an example, it is possible to detect a suspicious spoofed DOS or a host scanning by means of the finite-state machine approach proposed in [9], but it is not possible to reveal the real attack source at this stage.

In order to finalize the detection, the first detection level (i.e. D-StreaMon agents) has to send alerts for the mitigation actions. The NIDS Manager collects and correlates such events to perform the final recognition of the ongoing attack and the correct attribution of the attack's source. For this objective, the NIDS Manager observes the source of the incoming warnings: the NIDS of the spoofed hosts (i.e. the *zombies*) will not report anything, while the NIDS of the attacker's anchor host (i.e. the real source of the attack) will report a warning. More specifically, representing a warning through the notation $W_i^{y \to z}$, where i is the NIDS agent reference and y→z indicates the attack direction, the IP spoofing condition can be modeled as:

$$W_v^{B \to V} AND NOT W_b^{B \to V}$$

This detection algorithm can be implemented by means of a simple *Look-Up Table*, able to track all the events aggregated by the NIDS Manager. Once a spoofed IP event is detected, the attribution is performed aggregating, for each attack type, the warning referring to the same victim. The offensive activity is attributed to the host for which the event is recognized both by the own ($W_a^{A \to V}$) and the victim NIDS.

5. Signature validation

In order to validate the proposed detection technique, we implemented a virtual lab with a Kali attacker box, two WinXP victim hosts and two Win7 victim hosts, as shown in Figure 6. In particular, using this virtual lab, we specifically test both the two scenarios described in Section 4.

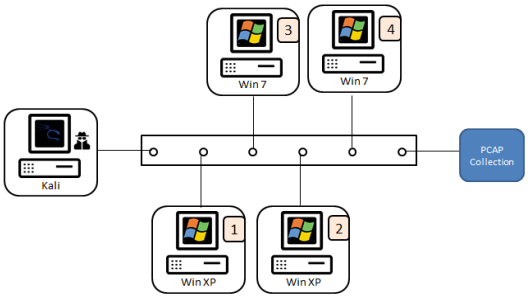

Figure 6. Validation architecture

Firstly, concerning the validation of the signatures described in Figure 4, we registered a 3-hour traffic track, emulating routine operations typical of an office host, as web browsing, shared directories accesses, operating system and antivirus updates. Additionally, we performed different lateral movements (opportunely scheduled) between all the hosts, using all the techniques described in Section 4.1 (for PSEXEC we used the tools described in [18]). Then, in order to assess how to face the truncated chain events and false alarms with an effective timeout management, in some cases we voluntarily used incorrect credentials. From Kali box to host (2), we also performed a SMB brute force (trying to access to shared resources). For this purpose, we used the implementation of the well-known hacking platform Metasploit, by means of the module "*auxiliary/scanner/smb/smb_login*" [18].

We then validated the technique proposed in Section 4.2 to detect and attribute IP-spoofing based attack inside the LAN, represented in Figure 5. In particular, we simulated a decoy scan from the Kali host to a Windows one (considering the remaining hosts as zombies), using the well-known network scanner *nmap* (with argument *"–D"*); the same scenario was reproduced for the spoofed DOS attack, using the opensource penetration tool *hping3*.

5.1 Test results

The offensive activities, that were expected to be detected, were correctly identified by the proposed solution, without any false alarm.

Table 2. Performed lateral movement for validation

HOSTS		PSEXEC TOOL	COUNT
Attacker	Victim		
K	1,2,3,4	Impacket	3
1	2,3,4	PsExec SysInt.	2
2	1,3,4	PsExec SysInt.	2
3	1,2,4	PsExec SysInt.	2
4	1,2,3	PsExec SysInt.	2

In order to understand the advantages related to the use of such approach with respect to the stateless NIDS one, we performed many PSEXEC attacks among the hosts (as shown in Table 2) and we logged all the warnings received using the state transitions individually considered as stateless signatures (apart from transitions 1.a→2 and 2.a1→2.a2, which requires information from the LUT).

Table 3. Avoided false alarms

	Tran. 0-0.1	Tran. 0.1-1	Tran. 1-1.a	Tran. 2-2.a1	Tran. 2.a2-2.a3	Tran. 2.a3-2.a4
Pkts	4615	70	54	222	47	68
Pkts/Tot [%]	6.59	0.10	0.08	0.32	0.07	0.10

Analyzing the results shown in Table 3 (related to a track of ~70k packets), it is possible to verify that:

- if we had analyzed this traffic with a stateless NIDS with those signatures, we would have received many false alarms, mainly associated to administration licit connections;
- the peak present for T0-1 column mainly relates to the SMB brute force attack intentionally performed from Kali (K) machine to WinXP (2) host, which was anyway correctly detected by the signature related to the branch 0→0.1 of Figure 4.

6. Conclusions and Future Works

This work shows how to effectively model malicious lateral movements for detection purposes, by means of finite-state machine signatures to be deployed in IDS agents (host level). The validity of such an approach is strengthened with additional more complex use cases. The possibility to capillary monitor the traffic at host level with the deployable version of StreaMon [9] enables additional detection scenarios. For instance, the events/warnings generated at host level can be aggregated in a centralized NIDS Manager, which can correlate them so as to detect and attribute advanced attacks, such as ones based on IP spoofing which aim at evading the security controls. The manager, correlating the stored information, can then trigger proper mitigation actions (blocking, honeynets and so on), that can be implemented with custom solutions or integrated in a SDN architecture.

7. References

[1] Tankard, Colin. "Advanced persistent threats and how to monitor and deter them." Network security 2011.8 (2011): 16-19.

[2] "APT Detection – Closing the Gaping Hole", 2014

[3] Dan Reis, "It's Only the Beginning for Endpoint Security", FirEye 05

[4] A. Oberle et al., "Preventing Pass-the-Hash and Similar Impersonation Attacks in Enterprise Infrastructures," 2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA), Crans-Montana, 2016, pp. 800-807.

[5] "Pass-the-hash attacks: Tools and Mitigation", SANS, 2010

[6] Jadeja et al. "Implementation and Mitigation of Various Tools for Pass the Hash Attack." Procedia Computer Science 79 (2016): 755-764.

[7] Roesch, Martin. "Snort: Lightweight Intrusion Detection for Networks." LISA. Vol. 99. No. 1. 1999.

[8] Marchetti, Mirco, et al. "Analysis of high volumes of network traffic for Advanced Persistent Threat detection." Computer Networks (2016).

[9] G. Bianchi et al. , "StreaMon: A software-defined monitoring platform" , Teletraffic Congress (ITC), 2014 26th International , pp.1 -6

[10] Raghav, Iti, Shashi Chhikara, and Nitasha Hasteer. "Intrusion Detection and Prevention in Cloud Environment: A Systematic Review." International Journal of Computer Applications 68.24 (2013).

[11] "Mitigating Pass-the-Hash and Other Credential Theft", Microsoft, 2014

[12] G. Rush et al., "DCAFE: A Distributed Cyber Security Automation Framework for Experiments," Computer Software and Applications Conference Workshops, 2014 IEEE 38th International, Vasteras, 2014.

[13] "Detecting “Pass-the-hash” attacks with Sagan in real time." https://quadrantsec.com/about/blog/detecting_pass_the_hash_attacks_with_sagan_in_real_time/

[14] M. Ussath, D. Jaeger, Feng Cheng and C. Meinel, "Advanced persistent threats: Behind the scenes," 2016 Annual Conference on Information Science and Systems (CISS), Princeton, NJ, 2016, pp. 181-186.

[15] Ventre, Pier Luigi, et al. "D-STREAMON-a NFV-capable distributed framework for network monitoring." arXiv preprint arXiv:1608.01377 (2016).

[16] Mitchell, David. "Intrusion Detection from Simple to Cloud" (2015).

[17] Sekar, R., et al. "Specification-based anomaly detection: a new approach for detecting network intrusions." Proceedings of the 9th ACM conference on Computer and communications security. ACM, 2002.

[18] A.Greco, G.Bianchi – ”Detection of offensive lateral movements using finite-state-machine-based patterns” – Global Wireless Summit, Aarhus, 2016

[19] Hintjens, Pieter. ZeroMQ: Messaging for Many Applications. " O'Reilly Media, Inc.", 2013.

[20] A.Greco, A.Caponi, G.Bianchi – “Facing lateral movements using widespread behavioral probes” - 11th International Conference for Internet Technology and Secured Transactions, Barcelona, 2016

8. Acknowledgements

This research was partially supported by the EU Commission within the Horizon 2020 program, SCISSOR project grant no 644425.

