Unveiling SSHCure 3.0: Flow-based SSH Compromise Detection

Rick Hofstede, Luuk Hendriks
Design and Analysis of Communication Systems (DACS)
Centre for Telematics and Information Technology (CTIT)
University of Twente, Enschede, The Netherlands
Email: {r.j.hofstede, luuk.hendriks}@utwente.nl

***Abstract*—Network-based intrusion detection systems have always been designed to report on the presence of attacks. Due to the sheer and ever-increasing number of attacks on the Internet, Computer Security Incident Response Teams (CSIRTs) are overwhelmed with attack reports. For that reason, there is a need for the detection of compromises rather than compromise attempts, since those incidents are the ones that have to be taken care of. In previous works, we have demonstrated and validated our state-of-the-art compromise detection algorithm that works on exported flow data, i.e, data exported using NetFlow or IPFIX. The detection algorithm has been implemented as part of our open-source intrusion detection system SSHCure.**

In this demonstration, we showcase the latest release of SSHCure, which includes many new features, such as an overhauled user interface design based on user surveys, integration with incident reporting tools, blacklist integration and IPv6 support. Attendees will be able to explore SSHCure in a semi-live fashion by means of practical examples of situations that CSIRT members encounter in their daily activities.

Flow-based intrusion detection is an active area of research, driven by the need for scalable solutions in high-speed networks of 10 Gbps and above. Restrained by the fact that flows, defined as "a set of IP packets passing an observation point in the network during a certain time interval" [1], traditionally only feature traffic information at L3 and L4, flow-based intrusion detection was always believed to only be useful for detecting very specific attacks, such as large-scale flooding attacks and scans [2]. However, also another class of attacks was shown to be well-detectable using flow data: brute-force attacks, against SSH or RDP daemons, for example. The fact that these attacks consist of many login attempts against a daemon using a *dictionary*,[1] makes that such attacks feature a very characteristic behavior at the network-level. Several experimental tools have been developed by academia for detecting such attacks based on flow data. However, like most other intrusion detection systems, even those based on individual packets, these tools merely report on the *presence* of attacks. Since many types of attacks are nowadays considered 'background noise' on the Internet, these attack reports overload Computer Security Incident Response Teams (CSIRTs). As a consequence, a large number of reports may cause important reports to be overlooked.

SSHCure is our flow-based SSH intrusion detection system that is unique in many ways. Besides the fact that there hardly exists any well-maintainted flow-based intrusion detection system, its most distinguishing feature is the detection of compromises [3]. As such, CSIRTs can concentrate on those machines that are known to be compromised and at risk, both for themselves and other machines, and quarantaine or disconnect them from the network.

[1] *Dictionaries* are lists with the most popular username and password combinations.

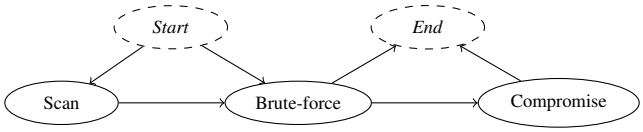

Fig. 1. Dictionary attack phases, from [4].

The detection algorithms used by SSHCure have been completely designed around our SSH attack model since the very first prototypes. This model classifies SSH attacks into three consecutive phases, as shown in Fig. 1:

1) **Scan** – Attackers perform a horizontal network scan to identify active SSH daemons. This phase features a very small number of packets per flow, mostly TCP `SYN` packets.
2) **Brute-force** – Attackers perform a brute-force attack by issuing many authentication requests against one or more targets (i.e., SSH daemons). This is typically done by means of *dictionaries*. The traffic in this phase consists of high-intensity TCP connections that are very similar in size (both packets and bytes) and duration, commonly referred to as *flat* traffic.
3) **Compromise** – Attackers have gained access to a target host by using correct login credentials. This phase typically features either very small flows in case of idle connections, or large flows in case the compromised target is being actively misused.

This attack model was already proposed by our group as early as 2009 [5] and is still at the core of SSHCure. Although it advanced the state-of-the-art in the area of flow-based brute-force attack detection at the time it was proposed, it soon became clear that it yielded too many false detections in production environments. The root cause of these false detections was the assumption that traffic in the *brute-force* phase is observed flat; extensive analysis has shown that

especially attacks from far-away countries can feature quite some retransmissions, for example, causing the network traffic not to be completely flat anymore [6]. Our solution to this problem was a new compromise detection algorithm that is resilient against such variability at the network-layer, based on analysing the behavior of attack tools upon a compromise [4]. The fact that the latest version of SSHCure features an accuracy close to 100% shows that this advancement of the state-of-the-art is reliable enough for usage in production networks.

SSHCure has been open-source and available for free since May 2012, and has seen a rapid adoption by both academia and industry since then. We are aware of various deployments in a plethora of environments, ranging from small Web-hosting companies to backbone networks and government CSIRTs. A wide audience was targeted right from the start of SSHCure's development efforts by providing installation scripts for many Linux and BSD distributions, allowing even those users who are less technical to use SSHCure. Selecting the popular open-source flow collector *NfSen*[2] as the underlying platform for SSHCure has driven SSHCure's wide deployment. Recently, we have moved our development efforts from SourceForge[3] to GitHub,[4] in order to foster community interaction using GitHub's networking features. SSHCure is being actively developed, maintained and supported, and is licensed under the three-clause BSD license.

We will showcase the upcoming release of SSHCure 3.0 during the NetSys 2015 Demonstration Session. This release features both new features and improvements:

- New user interface design, adapted to better suit the needs of SSHCure users.
- Integration with security incident reporting tools, such as Qmanage and AIRT, using XML-RPC, X-ARF and IODEF interfaces. This allows SSHCure to be integrated into existing security incident handling workflows, e.g., Security Information and Event Management (SIEM).
- Integration with the OpenBL SSH blacklist, providing additional insights when analysing detection results.
- Full IPv6 support.

NetSys 2015 attendees will be able to explore SSHCure's new features and analyze attacks that target or originate from the observed networks in a semi-live fashion. In particular, we will demonstrate various situations that CSIRT members encounter in their daily activities and how SSHCure supports them in that. In chronological order, we will guide the audience from the initial phases of attacks until the actual detection and reporting of a compromise. While attacks are started in the background, SSHCure will visualize the various attack phases and provide technical details.

Those attendees wishing to explore SSHCure within their own environment can install SSHCure themselves; the only requirement is a Linux or BSD system that has *NfSen* installed, together with several dependencies that are typically provided by any packet manager. The system should receive flow data exported using NetFlow or IPFIX.

[2]http://nfsen.sf.net

[3]http://sshcure.sf.net

[4]https://github.com/sshcure/sshcure

Acknowledgments

Special thanks go out to David Young for the overhaul design featured in SSHCure 3.0. This work was partly funded by FLAMINGO, a Network of Excellence project (ICT-318488).

References

[1] B. Claise, B. Trammell, and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information," RFC 7011 (Internet Standard), Internet Engineering Task Force, September 2013. [Online]. Available: http://www.ietf.org/rfc/rfc7011.txt

[2] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, "An Overview of IP Flow-Based Intrusion Detection," *IEEE Communications Surveys & Tutorials*, vol. 12, no. 3, pp. 343–356, 2010.

[3] L. Hellemons, L. Hendriks, R. Hofstede, A. Sperotto, R. Sadre, and A. Pras, "SSHCure: A Flow-Based SSH Intrusion Detection System," in *Dependable Networks and Services. Proceedings of the 6th International Conference on Autonomous Infrastructure, Management and Security, AIMS'12*, ser. Lecture Notes in Computer Science, vol. 7279. Springer Berlin Heidelberg, 2012, pp. 86–97.

[4] R. Hofstede, L. Hendriks, A. Sperotto, and A. Pras, "SSH Compromise Detection using NetFlow/IPFIX," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 20–26, 2014.

[5] A. Sperotto, R. Sadre, P.-T. de Boer, and A. Pras, "Hidden Markov Model modeling of SSH brute-force attacks," in *Proceedings of the 20th IEEE/IFIP International Workshop on Distributed Systems: Operation and Management, DSOM'09*, 2009.

[6] M. Jonker, R. Hofstede, A. Sperotto, and A. Pras, "Unveiling Flat Traffic on the Internet: An SSH Attack Case Study," in *Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management, IM'15*, 2015, (accepted for publication).