10-1-2016

Active snort rules and the needs for computing resources: Computing resources needed to activate different numbers of snort rules

Chad A. Arney
Michigan Technological University

Xinli Wang
Michigan Technological University

Follow this and additional works at: http://digitalcommons.mtu.edu/technology-p

 Part of the Databases and Information Systems Commons, and the Information Security Commons

Active Snort Rules and the Needs for Computing Resources

– Computing Resources Needed to Activate Different Numbers of Snort Rules

Chad A. Arney
Michigan Technological University
Houghton, MI 49931, USA
caarney@mtu.edu

Xinli Wang*
Michigan Technological University
Houghton, MI 49931, USA
xinlwang@mtu.edu

ABSTRACT

This project was designed to discover the relationship between the number of enabled rules maintained by Snort and the amount of computing resources necessary to operate this intrusion detection system (IDS) as a sensor. A physical environment was set up to loosely simulate a network and an IDS sensor monitoring it.

The experiment was conducted in five trials. A different number of Snort rules was enabled in each trial and the corresponding utilization of computing resources was measured. Remarkable variation and a clear trend of CPU usage were observed in the experiment.

Categories and Subject Descriptors

H.3.4 [**Information Systems**]: Systems and Software—*Performance evaluation*

Keywords

Snort; Rule Set; Performance; Utilization of Computer Resources; Tuning

EXECUTIVE SUMMARY

A physical network was set up with two computers to discover the relationship between the number of active rules loaded by Snort and the amount of computing resource that is needed for its operation. One computer (Host A) ran Security Onion Linux providing Snort of version 2.9.8.0. The other (Host B) ran Kali Linux of version 2016.1 to simulate an attacker. The sysstat software tool was used to measure resource utilization by Snort on Host A. Five trials were designed for our experiment. Groups of rules were disabled by categories to achieve the desired number of enabled rules for each trial.

*Corresponding author: Phone: 906-487-1873

RIIT'16 September 28 - October 01 2016, Boston, MA, USA

ACM ISBN 978-1-4503-4453-1/16/09.

DOI: http://dx.doi.org/10.1145/2978178.2978189

Sysstat reports a whole set of resource usages. We analyzed the data and found no significant difference in resource usage between the trials except CPU utilization. Therefore, only the results of CPU usage is presented in this paper.

When looking at the averages over the time period of the experiment, Trials 1-3 do not show any appreciable difference in CPU load although only 60% of the rules are enabled in Trial 3 and 100% in Trial 1. Compared with Trials 1-3, Trials 4-5 generate a much lower average CPU load. This may indicate a nonlinear relationship between the number of enabled rules and the CPU power necessary to maintain Snort operation. The nonlinear relationship may be resulted from the complexity of rule sets. In the experiment, we consider the number of enabled rules solely for simplicity. However, different types of rules will impose various CPU loads. For example, a rule that needs a content search into payload data will demand more CPU power than a rule that logs a short message only. Some of the enabled rules may not be active because of a different port number or protocol.

Our data show a strong time variation in CPU load in all of the five trials. While a general trend of less CPU usage when moving from a larger to a smaller set of enabled rules is observed, three interesting periods in the 90-minute experiment can be identified:

- At the beginning (in the first 20 minutes), there is generally a high CPU load at the very beginning (first 2-10 minutes) in Trials 1-3. Then the CPU load decreases sharply.
- In the period of 20-60 minutes, CPU load increases first and decrease again. In addition, measurements show highly scattered spots.
- After 60 minutes, measurements are clustered together. CPU load maintains a relatively low level in all of the five trials.

It is speculated that this time variation in CPU load in each trial may reflect the Snort use of rule caching and Stream5 preprocessor, which is a target-based TCP reassembly module for Snort and capable of tracking sessions for both TCP and UDP. Both of them will allow Snort to learn and require less CPU processing time after a certain period of time.

Acknowledgment

The work is supported by the National Science Foundation (NSF) TUES grant award#: 1140308.