Evaluation of passiv stepping stone detection techniques under chaff and delay

Michael Shell
Georgia Institute of Technology
someemail@somedomain.com

Homer Simpson
Twentieth Century Fox
homer@thesimpsons.com

James Kirk
and Montgomery Scott
Starfleet Academy
someemail@somedomain.com

***Abstract*—Bla**

I. Introduction

Network attackers frequently use a chain of compromised intermediate nodes to attack a targetmachine and maintain anonymity. This chain of nodes between the attacker and the target is calleda stepping stone chain.

A. Related work

[?]

[?]

[?]

Stepping Stone Detection Techniques: Classification and State-of-the-Art, bad though

Metrics: A Study on the Performance Metrics forEvaluating Stepping Stone Detection (SSD) Stepping Stone Detection: Measuring the SSD Capability

II. Selected approaches

A. Packet-correlation-based approaches

Efficient multi-dimensional flow correlation

Detecting Connection-Chains: A Data Mining Approach

Correlating TCP/IP Packet contexts to detectstepping-stone intrusion 2011

B. RTT-based approaches

Detecting Stepping-Stone Intrudersby Identifying Crossover Packets in SSH Connections

RTT-based Random Walk Approach to Detect Stepping-Stone Intrusion

Detecting Stepping-Stone Intruders with Long Connection Chains? 2009

C. Anomaly-based approaches

Crescenzo et al. [1]

Network and Distributed Systems Security (NDSS) Symposium 2020
23-26 February 2020, San Diego, CA, USA
ISBN 1-891562-61-4
https://dx.doi.org/10.14722/ndss.2020.23xxx
www.ndss-symposium.org

D. Detecting Anomalies in Active Insider Stepping Stone Attacks

Huang et al. [2].

E. Neural networks

Performance of neural networks in stepping-stone intrusion detection 2008 but no good results, better in Neural networks-based detection of stepping-stone intrusion

III. Dataset creation

A. Simulating stepping stones with SSH-tunnels and Docker

insert figure with one and with three stepping stones

SSH tunnel on respective port on the starting point of the chain, tunnels to port on the next point in the chain. Finally, Refer to figure.

1) Adding network congestion: Docker communication takes place over virtual bridge networks, so the throughput is far higher and more reliable than in real-world networks. To retard the quality of the Docker network to realistic levels, we rely on the emulation tools Netem. Netem add reference is a Linux command line tool that allows users to artificially simulate network conditions such as high latency, low bandwidth or packet corruption in a flexible manner. We apply Netem commands to the network interface of each container, which adds correlated delays to incoming and outgoing packets that are drawn from a normal distribution with mean μ, variance σ^2, and correlation ρ_1. We furthermore apply correlated packet loss and corruption drawn from a binomial distribution with probability p and correlation ρ_2.

We set the network settings for the starting point and the end point container individually and draw each of the given parameters from a suitable distribution (should I specify which one for each? Seems a bit much...) before each run to allow for a good amount of variation in the generated data.

2) Adding delays and chaff: To add artifical delays to forwarded packets on a stepping stone host for detection evasion, we can again use NetEm. We draw delays for departing packets from a uniform distribution, as suggested by add reference, covering the interval $[0, \delta_d]$, with no packet correlation.

To add chaff packets to the relayed connection, we forward two additional ports through the SSH-tunnel of a stepping stone host. We then use NetCat add reference to send data to both ports from either direction and collect it at the other side.

Figure ... depicts this setup for an individual tunnel. The data sent through tunnel i consists of strings with random size x drawn from a Cauchy-distribution with mean xx_i, and is sent in intervals of random length δ_c drawn from an exponential distribution with mean yy_i. By adjusting yy_i, we can control the amount of chaff sent through a tunnel.

B. Simulating interactive SSH-traffic

In order to generate enough data instances representing interactive stepping stone behaviour, we automatised the communication between the start point and the end point of the stepping stone chain. To do so, we generate a script with SSH-commands at the start of each execution that is passed and run by the starting point of the chain. The generated script consists of a sequence of ordinary SSH-commands list them here?, which are drawn randomly from a command catalogue and are each separated by *sleep*-commands for a time t that is drawn each time from a Cauchy-distribution. The average sleep-time is around insert. The length of the script is reached when the *end*-command is drawn from the catalogue. Insert example

C. HTTP-interactions

In order to provide an additional, different type of interaction between the starting point and end point, we directed HTTP traffic over the stepping stone chain. Here, the starting point hosts Scrapy, a web crawling service insert citation, that surfs the 1 million most popular website by clicking links on them. The requests are sent over the stepping stone chain to the web.

This type of traffic is not meant to necessarily represent realistic stepping stone behaviour, but to provide an additional source of interactive traffic that differs substantially from SSH in order to test detection methods from another angle.

SSH 1 node	no pert.	var. delays	var. chaff	delay&chaff
HTTP 1 node	no pert.	var. delays	var. chaff	delay&chaff
SSH 3 node	no pert.	var. delays	var. chaff	delay&chaff
HTTP 1 node	no pert.	var. delays	var. chaff	delay&chaff

[3]

IV. Results

A. Unperturbed data

References

[1] G. Di Crescenzo, A. Ghosh, A. Kampasi, R. Talpade, and Y. Zhang, "Detecting anomalies in active insider stepping stone attacks." *JoWUA*, vol. 2, no. 1, pp. 103–120, 2011.

[2] S.-H. S. Huang and Y.-W. Kuo, "Detecting chaff perturbation on stepping-stone connection," in *2011 IEEE 17th International Conference on Parallel and Distributed Systems*. IEEE, 2011, pp. 660–667.

[3] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," in *2010 IEEE Symposium on Security and Privacy*, May 2010, pp. 305–316.

Appendix