HAS Analysis: Detecting HTTP-based C&Cs based on the Analysis of HTTP Activity Sets

Sungjin Kim, Yujeong Han, Jaesung Lee, Younghan Choi, Byungchul Bae,
Hyunggeun Oh, Kiwook Sohn
The Attached Institute of ETRI
Yuseoung P.O. Box 1
Daejeon, Korea
{ksj1230, yjhan, berise, yhch, bcbae, hgoh, kiwook}@ensec.re.kr

***Abstract*— We focus on two distinctive features of HTTP-based C&C traffic by analyzing HTTP activity sets. First, C&Cs show a few connections at a time (low-density). Second, contents within a request or a response change frequently among consecutive C&Cs (content-change). Based on these two features, we propose a C&C analysis mechanism that detects unknown HTTP-based C&Cs with low false-positives.**

1 INTRODUCTION

Currently, numerous modern targeted attacks and banking trojans use HTTP protocol. HTTP-based C&Cs can be easily blended into normal web traffic obeying the protocol rules. Even more, web related ports are vitally allowed through firewalls. A previous behavioral pattern based approach concentrated on the periodical pattern of C&Cs [1]. However, this approach needs a well-defined whitelist to reduce false-positive rates since there are many benign applications such as an automatic update check having the periodical patterns. Recently, a behavioral malware clustering method [2] was presented. It offers only limited coverage because it does not focus on detecting unknown malwares, but on polymorphic variants of known malware families. Hence, we propose a C&C analysis mechanism focusing on the density and the content change of HTTP activity sets.

Our Approach. We first define a HTTP activity set. A HTTP activity set is a set of request-response pairs generated by the initial application request for a URI. Specifically, it can be generated by human-driven applications (e.g., web browser) or by automatic programs (e.g., RSS feeds and adwares). This is refered to as "HAS" in the rest of this paper. We found two main features of C&C HASes by analyzing 1,124 HASes collected from modern HTTP-based malware samples including banking trojans (e.g., Torpig, Zeus, and Tinba) and bootkits (e.g., Xpaj). We also analyze benign HASes by visiting 500 benign web sites.

1. *Low-density*: A C&C utilizes small number of connections at a time. Figure 1 and 2 show histograms of the number of requests and the referer tree depth in both C&C HASes and benign HASes. We confirmed that the number of requests in C&C HASes was 1.1 on average and the referer tree depth in C&C HASes is always 0. Using this feature, we filter benign HASes utilizing large number of connections in a HAS-D analysis.
2. *Content-change*: The content in a C&C HAS changes frequently at each time during the process of stealing data or changing command. Whereas, benign applications return a similar data for the same request. Figure 3 shows similarity scores between two adjacent contents within a request or a response at each time during connections of an Anti-Virus program and a Zeus malware. Using this feature, we distinguish C&C HASes from low density HASes in a HAS-CC analysis.

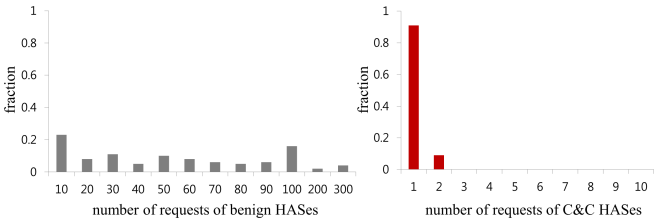

Figure 1: Histograms of number of requests in HASes.

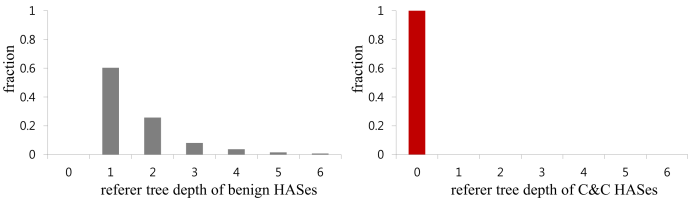

Figure 2: Histograms of referer tree depth in HASes.

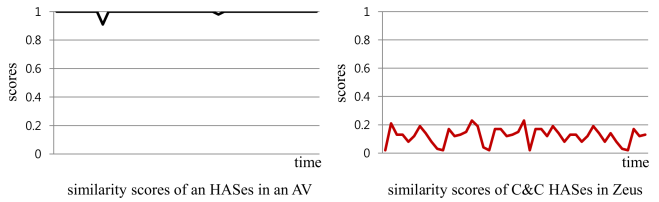

Figure 3: Similarity scores of HASes.

Main Contributions.

- We find two distinctive features of C&C traffic based on the analysis of HTTP Activity Sets.
- We propose a C&C analysis mechanism that detects unknown HTTP-based C&Cs with low false-positive rates. The proposed mechanism also detects non-periodic C&Cs.

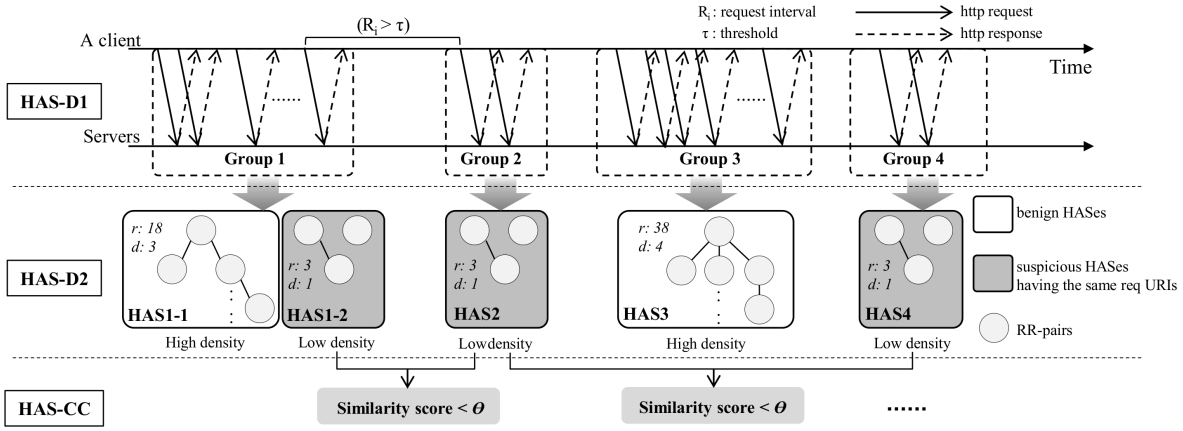

Figure 4: Description of the HAS analysis mechanism.

2 HAS ANALYSIS MECHANISM

The HAS analysis mechanism is composed of two main analysis modules including a `density-check (HAS-D)` and a `content-change-check (HAS-CC)`. Figure 4 gives a full description of the proposed mechanism.

HAS-D Analysis. The objective of the HAS-D analysis is to filter high-density HASes which are regarded as benign. We group the collected request-response pairs (RR-pairs) based on the time interval of http requests (the HAS-D1 analysis) at fisrt. If the time interval between two adjacent requests exceeds the threshold τ, we divide them into separate groups. Then we split a group into HASes by constructing referer trees using referer field information. Each generated referer tree becomes a HAS and the rest of RR-pairs having the same request URIs are merged into a HAS. In figure 4, we can see that the Group1 is divided into two HASes composed of one high-density HAS and the other low-density HAS by constructing referer trees and merging RR-pairs. Finally, we check the density of each HAS (the HAS-D2 analysis). HASes whose number of requests r and referer tree depth d are larger than thresholds (σ_r: a threshold number of the requests, σ_d: a threshold of the referer tree depth) are filtered out. For example, HAS1-1 and HAS3 will be filtered if the threshold σ_r is 4 and the threshold σ_d is 2. After applying the HAS-D analysis, only low density HASes will remain. The HASes generated by automatic programs can cause false-positives. Thus, we should filter the rest of benign HASes.

HAS-CC Analysis. To distinguish C&C HASes from low density HASes, we apply the HAS-CC analysis by comparing similarity scores between two adjacent HASes having the same request URIs. If the similarity score exceeds the threshold θ, those two HASes are stored and considered as malicious.

Evaluations. For the evaluations, we craft a malicious code that exfiltrates documents steadily from certain folders. The visiting pattern to a C&C server of the crafted malware is randomized, not periodic. A test bed consists of 10 benign clients and two infected clients that generate both C&C HASes and benign web page accesses. An overview of the proposed system prototype is presented in figure 5. We measured the request time interval of 500 benign web sites. The mean value of the time interval is 421ms with maximum value 1.8s. Based on the measurements in section 1 and section 2, we determine the thresholds $\tau = 1.8$, $\sigma_r = 4$, $\sigma_d = 2$ and $\theta = 0.5$ heuristically. We identify all of the 126 C&C HASes from 32,026 web sessions composed of 2,519 HASes in the evaluations. The result is presented in Table 1. The false-positive in HAS-D & HAS-CC is caused by an automatic programs that periodically send logged data to an application server. However, most automatic program traffic is filtered out in the HAS-CC analysis.

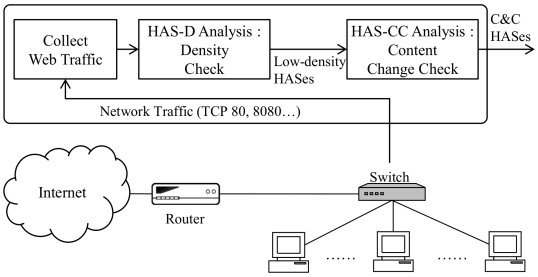

Figure 5: Overview of the proposed system prototype.

Table 1: Comparison of detection rates with f/p

Analysis Mechanism	Detection Rate	False Positive
HAS-D only	100%	9.60%
HAS-D & HAS-CC	100%	0.83%

3 FUTURE WORK

The proposed system is being operated in an enterprise-level network at present. Additionally, we will find the optimal values of the thresholds in future work.

4 REFERNCES

[1] G. Gu, J. Zhang, W. Lee. BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic. In USENIX NDSS, 2008

[2] R. Perdisci, W. Lee, and N. Feamster. Behavioral clustering of http-based malware and signature generation using malicious network traces. In USENIX NSDI, 2010.