Detection of Botnets Using Combined Host- and Network-Level Information

Yuanyuan Zeng, Xin Hu, Kang G. Shin
University of Michigan, Ann Arbor, MI 48109-2121, USA
{gracez, huxin, kgshin}@eecs.umich.edu

1. INTRODUCTION

Botnets have now become one of the most serious security threats to Internet services and applications; they can mount Distributed-Denial-of-Service (DDoS) attacks, spamming, phishing, identity theft, and other cyber crimes.

To control a botnet, a botmaster needs to use a C&C channel to issue commands and coordinate bots' actions. Although IRC- and HTTP-based C&C have been adopted by many past and current botnets, both of them are vulnerable to a central-point-of-failure. That is, once the central IRC or HTTP server is identified and removed, the entire botnet will be disabled.

To counter this weakness, attackers have recently shifted toward a new generation of botnets utilizing decentralized C&C protocol such as P2P. This C&C infrastructure makes detection and mitigation much harder. A well-known example is the Storm worm (a.k.a. Nuwar, W32.Peacomm, and Zhelatin) [1] which spreads via email spam and is known to be the first malware to seed a botnet in a hybrid P2P fashion. Storm uses peers as HTTP proxies to relay C&C traffic and hides the botmasters well behind the P2P network. A recent spambot Waledac, which appeared at the end of 2008, also spreads via spam emails and forms its botnet using a C&C structure similar to the Storm botnet. Some researchers pointed out that Waledac is the new and improved version of the Storm botnet [2].

To date, most botnet-detection approaches operate at the network level; a majority of them target traditional IRC- or HTTP-based botnets [3, 4, 5, 6, 7, 8] by looking for traffic signatures or flow patterns. We are aware of only one approach [9] designed for protocol- and structure-independent botnet detection; it depends on network traffic analysis and is unlikely to have a complete view of botnets' behavior. We thus need the finer-grained host-by-host behavior inspection to complement the network analysis. On the other hand, since bots behave maliciously system-wide, general host-based detection can be useful. One such way is to match malware signatures, but it is effective in detecting known bots only. To deal with unknown bot infiltration, in-host behavior analysis [10, 11, 12, 13, 14] is needed. However, since in-host mechanisms are vulnerable to host-resident malware, host-based approaches alone can hardly provide reliable detection results and thus we need external, hard-to-compromise (i.e., network-level) information for detection of bots' malicious behavior.

Considering the required coordination within each botnet at the network level and the malicious behavior each bot exhibits at the host level, we propose a C&C protocol-independent detection framework that incorporates information collected at both the host and the network levels. The two sources of information complement each other in making detection decisions. Our framework first identifies suspicious hosts by discovering similar behaviors among different hosts using network-flow analysis, and validates the identified suspects to be malicious or not by scrutinizing their in-host behavior. Since bots within the same botnet are likely to receive the same input from the botmaster and take similar actions, whereas benign hosts rarely demonstrate such correlated behavior, our framework looks for flows with similar patterns and labels them as triggering flows. It then associates all subsequent flows with each triggering flow on a host-by-host basis, checking the similarity among those associated groups. If multiple hosts behave similarly in the trigger-action patterns, they are grouped into the same suspicious cluster as likely to belong to the same botnet. Whenever a group of hosts is identified as suspicious by the network analysis, the host-behavior analysis results, based on a history of monitored host behaviors, are reported. A correlation algorithm finally assigns a detection score to each host under inspection by considering both network and host behaviors.

Our contributions are three-fold. First, to the best of our knowledge, this is the first framework that combines both network- and host-level information to detect botnets. The benefit is that it completes a detection picture by considering not only the coordination behavior intrinsic to each botnet but also each bot's in-host behavior. For example, it can detect botnets that appear stealthy in network activities with the assistance of host-level information. Moreover, we extract features from NetFlow data to analyze the similarity or dissimilarity of network behavior without inspecting each packet's payload, thus preserving privacy. Second, our detection relies on the invariant properties of botnets' network and host behaviors, which are independent of the underlying C&C protocol. It can detect both traditional IRC and HTTP, as well as recent hybrid P2P botnets. Third, our approach is evaluated by using several days of real-world NetFlow data from a core router of a major campus network containing benign and botnet traces, as well as multiple benign and botnet data sets collected from virtual machines. Our preliminary results show that the proposed framework can detect different types of botnets with low false-alarm rates.

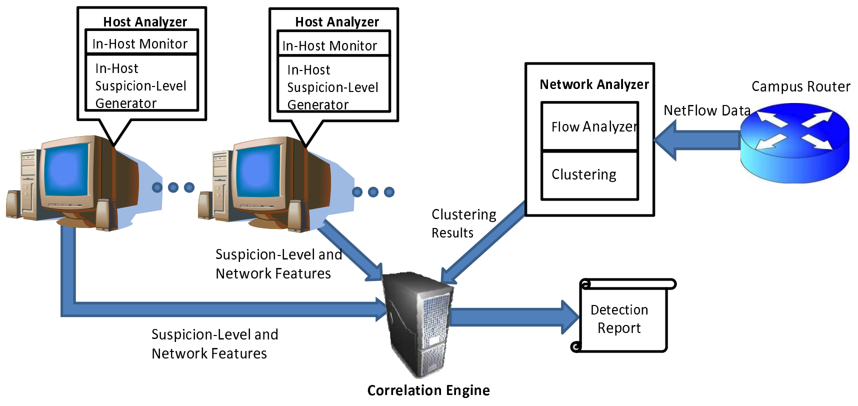

Figure 1: System architecture

2. SYSTEM ARCHITECTURE

Figure 1 shows the architecture of our system which primarily consists of three components: host analyzer, network analyzer, and correlation engine.

As almost all of current botnets target Windows machines, our host analyzer is designed and implemented for Windows platforms. The host analyzer is deployed at each host and contains two modules: in-host monitor and suspicion-level generator. The former monitors run-time system-wide behavior taking place in the Registry, file system, and network stack on a host. The latter generates a suspicion-level by applying a machine-learning algorithm based on the behavior reported at each time window and computes the overall suspicion-level using a moving average algorithm. The host analyzer sends the average suspicion-level along with a few network feature statistics to the correlation engine, if required. The network analyzer also contains two modules: flow analyzer and clustering. The flow analyzer takes the flow data from a router as input and searches for trigger-action botnet-like flow patterns among different hosts. It then extracts a set of features that can best represent those associated flows and transforms them into feature vectors. Those vectors are then fed to the clustering module that groups similarly-behaving hosts into the same cluster, assuming them to be part of a botnet. Whenever a suspicious group of hosts are identified by the network analyzer, their host analyzers are required to provide the suspicion-level and network statistics to the correlation engine, which verifies the validity of the host information by comparing the network statistics collected from the network and the host. The correlation engine finally assigns a detection score to each host and produces a detection result.

3. PRELIMINARY RESULTS

3.1 Data Collection

We evaluated the performance of our framework in detecting 3 types of botnets (i.e., IRC-based, HTTP-based and hybrid-P2P) with real-world traces. We set up VMWare virtual machines running Windows XP, connected via a virtual network to monitor and collect their activities at the Registry, file system, and network stack. Table 1 shows the details of these botnet traces, each containing 4 bot instances. IRC-rbot and IRC-spybot's traces were generated by running their modified source code in the virtual network. We obtained the binaries of HTTP-based BobaxA and BobaxB, and hybrid-P2P-based Storm and Waledac from public web sites. The IRC- and HTTP-based botnets' network-level traces were captured within a controlled environment and transformed from packet data to flow data in our experiment. Since Storm and Waledac botnets were still active in the wild when we collected data, to make it more realistic, we carefully configured the firewall setting and connected virtual machines to the outside network so that the bots actually joined the real Storm and Waledac botnet and the campus router captured all of the bots' traffic. We also collected 5-day NetFlow data from a core router in our campus network which covered the flows generated by Storm and Waledac instances and all other hosts in the network. The 5-day data consists of three sets: (1) 2-day data that contains 48-hour Storm traces; (2) 1-day data including Waledac; and (3) other 2-day data. We divided the third data into two subsets, 1-day each. We overlaid the remaining 4 botnet network traces, one at a time, on this data set, two traces on the first day, and two on the second day. For example, the IRC-rbot includes 4 bot instances, so we randomly selected 4 hosts from the clean 1-day traffic and replaced the bots' IPs with the selected IPs so that they exhibit both benign and malicious flow patterns. We treated Storm traces in the same way and intentionally overlaid the 1-day Waledac traffic on HTTP-intensive benign hosts.

3.2 Detection Results

We now report the detection results on 6 botnets. The performance of our detection framework was measured by false-alarm rates, i.e., false-positive (FP) and false-negative (FN) rates. A false-positive is defined as a benign host mistakenly classified as bot-infected, and a false-negative means that a bot-infected host fails to

Table 1: Botnet traces

Trace	Duration	Number of Bots
IRC-rbot	24h	4
IRC-spybot	32m	4
HTTP-BobaxA	4h	4
HTTP-BobaxB	20h	4
Storm	48h	4
Waledac	24h	4

Table 2: False alarm rates

Trace	Avg FP hosts	Avg FP	Avg FN Hosts	Avg FN	Duration
IRC-rbot	3.208	0.0016	0.125	0.0313	24h
IRC-spybot	2.833	0.0014	0	0	24h
HTTP-BobaxA	1.000	0.0005	0	0	24h
HTTP-BobaxB	1.083	0.0005	0	0	24h
Storm	2.563	0.0013	0	0	48h
Waledac	0.9167	0.0005	0	0	24h

be detected.

Table 2 shows our evaluation results where the average number of FP or FN hosts is calculated during the entire period of evaluation. The average FP or FN rate is the number of FP hosts divided by the total number of benign hosts (which was around 2,000), or the number of FN hosts divided by the total number of bot-infected hosts. Our framework was found to be able to identify almost all of bot-infected hosts. There was only one bot undetected, generating a false-negative. We verified that this bot did not have any C&C or malicious network activity and thus failed to form a suspicious cluster with other bots.

Our framework also performs well in terms of false-positives. The highest false-positive rate was no greater than 0.16%. It turned out that almost all false-positive hosts appeared during the first few hours because our framework relied more on the network-level analysis at that time, reflecting the lack of trust in host-level information. Detection inaccuracy thus occurred when a group of benign hosts demonstrated similar network behaviors among themselves or behaviors similar to bot-infected hosts. As the host-level information was verified to be trustable, the host-level information gradually had a higher weight and can correct the detection results. Network- and host-level information complement each other, and hence combining them while making a detection decision is the key in reducing false alarm rates.

4. REFERENCES

[1] : Storm worm. http://en.wikipedia.org/wiki/Storm-Worm

[2] Nazario, J.: Walking waledac. http://asert.arbornetworks.com/2009/01/walking-waledec/ (January 2009)

[3] Gu, G., Zhang, J., Lee, W.: BotSniffer: Detecting botnet command and control channels in network traffic. In: Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08). (February 2008)

[4] Binkley, J.R., Singh, S.: An algorithm for anomaly-based botnet detection. In: Proceedings of the 2nd conference on Steps to Reducing Unwanted Traffic on the Internet, Berkeley, CA, USA, USENIX Association (2006)

[5] Goebel, J., Holz, T.: Rishi: Identify bot contaminated hosts by irc nickname evaluation. In: HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, Berkeley, CA, USA, USENIX Association (2007)

[6] Karasaridis, A., Rexroad, B., Hoeflin, D.: Wide-scale botnet detection and characterization. In: HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, Berkeley, CA, USA, USENIX Association (2007)

[7] Livadas, C., Walsh, R., Lapsley, D., Strayer, W.: Using machine learning technliques to identify botnet traffic. In: Proceedings of the 2nd IEEE LCN Workshop on Network Security. (November, 2006)

[8] Strayer, W.T., R.Walsh, Livadas, C., Lapsley, D.: Detecting botnets with tight command and control. In: Proceedings of the 31st IEEE Conference on Local Computer Networks. (November, 2006)

[9] Gu, G., Perdisci, R., Zhang, J., Lee, W.: BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In: Proceedings of the 17th USENIX Security Symposium (Security'08). (2008)

[10] Christodorescu, M., Jha, S., Seshia, S.A., Song, D., Bryant, R.E.: Semantics-aware malware detection. In: Proceedings of IEEE Symposium on Security and Privacy. (2005)

[11] Kirda, E., Kruegel, C., Banks, G., Vigna, G., Kemmerer, R.: Behavior-based spyware detection. In: Proceedings of the 15th USENIX Security Symposium. (2006)

[12] Forrest, S., Hofmeyr, S.A., Somayaji, A., Longstaff, T.A.: A sense of self for unix processes. IEEE Symposium on Security and Privacy (1996)

[13] Somayaji, A., Forrest, S.: Automated response using system-call delays. In: Proceedings of the USENIX Security Symposium. (2000)

[14] Sekar, R., Bendre, M., Bollineni, P., Dhurjati, D.: A fast automaton-based method for detecting anomalous program behaviors. In: Proceedings of the IEEE Symposium on Security and Privacy. (2001)