Poster: A Reputation-Based Resilient P2P Botnet

Jie Yin[1,2], Xiang Cui[1,2], Ke Li[3]

[1]Institute of Information Engineering, Chinese Academy of Sciences

[2]School of Cyber Security, University of Chinese Academy of Sciences

[3]Beijing University of Posts and Telecommunications

Beijing, China

Email: {yinjie, cuixiang}@iie.ac.cn, like_bupt@foxmail.com

***Abstract*—Hybrid P2P botnets relying on automated peer-list exchange represent one of the emerging trends in advanced botnets [1, 2, 3]. Although such kinds of botnets are immune to index poisoning, they're vulnerable to peer-list pollution attack. Defenders may use polluters (we assume only polluters exist since a Sybil node is equal to multi-polluters) to inject a large quantity of fake peers into botnet aggressively. Once majority of bots' peer lists are polluted, all the polluters would stop working simultaneously; result in the botnet disabled (e.g., Waledac [4]). Consequently, how to keep the authenticity of peer-list, given that it's inevitable to update peer list by exchanging with other unauthenticated bots, poses a great challenge. In this paper, we introduce an advanced hybrid P2P botnet which exploits a novel reputation-based and self-repairing mechanism to make sure high purity of peer-list.**

I. Reputation-based Mechanism

The bots in the proposed botnet are divided into two categories. One is called servant bots that are accessible from global Internet, they have static IP addresses, and behave with both client and server features. The other is known as client bots, they will not accept incoming connections because of firewall, private address, DHCP, etc.

Peer-list Format. The peer-list is contained in each bot, each entry of peer-list is a four tuple *<IP, PT, FC, CL>*, *IP* and *PT* denote a peer's IP address and service port, respectively; *FC* denotes the cumulative fail count, the FC value will increase when the following happens: (1). The bot fails to be connected; (2). The bot is successfully connected but cannot provide available information over a certain period of time; (3). The bot provides fake information (e.g., a standard public/private key pair or CommandID can be adopted to ensure authentication). *CL* denotes the confidence level of a peer. Specifically, *CL* is the core of the reputation mechanism, it further falls into three categories: *source*, *authentic* and *available*. Here, *source* is only used by servant bots to label itself as a potential infection source; *authentic* represents the peer is completely trustable; *available* means the peer works well currently (although it may be a polluter and may stop working at any time). We suppose $B=\{b_i\}_{|i=0,1\ldots n}$ denote the entire set of servant bots in a botnet. Note that only *servant bots* could be candidates in peer lists since client bots cannot be connected from global Internet. Thus, the peer list of a bot b_0 is

$$PL_0 = \begin{Bmatrix} < IP_1, PT_1, FC_1, CL_1 > \\ < IP_2, PT_2, FC_2, CL_2 > \\ < IP_3, PT_3, FC_3, CL_3 > \\ \ldots \\ < IP_M, PT_M, FC_M, CL_M > \end{Bmatrix} \quad (1)$$

where M is the size of peer list in each bot. To enhance the robustness of the proposed botnet, a larger peer-list is desirable. However, if a bot is captured by defenders, it means more peers will be exposed. Thus, M must be reasonable. It is feasible to choose an optimal M through mathematical analysis and simulations [1].

Peer-list Construction. The reputation-based peer-list construction mechanism is inspired by the fundamental fact that only infection source is trustable since polluters would never indeed propagate; otherwise, they are in fact contributing to the botnet, let alone ethical and lawful issues. Thus, when A infects a victim B, B could copy A's peer list unsuspectingly. More importantly, if A is a servant bot, B marks A as *authentic*. It's clear that the trust relationship could be delivered safely, that is, when B infects C, C will mark both A and B as *authentic* because B is *authentic* and A is trusted by B. Besides, if B is a servant bot and peer list is full, it will replace a *high-FC*, *low-CL* peer with itself, and label its *CL* value as *source*. In this way, the *authentic* peers in each bot will expand continuously until reaching peer-list size limit. Reinfection should also be taken into consideration through which a bot could obtain more *authentic* peers. Fig.1 and Fig.2 illuminate the procedure of infection and reinfection, assuming the peer-list size is four, A and B are servant bots and C is a client bot, D is already infected.

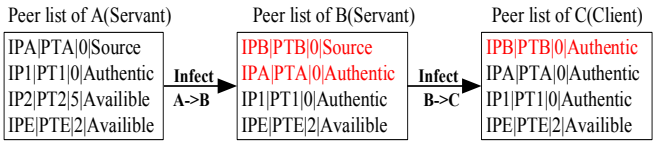

Fig. 1. Reputation-based Peer-list Delivery via Infection

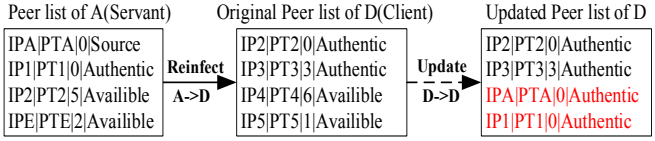

Fig. 2. Reputation-based Peer-list Update via Reinfection

Peer-list Exchange. For polluters, the only chance to pollute a botnet is through peer-list exchange; therefore, the peer list must be updated cautiously: (a). Only peers labeled as *available* are permitted to be updated; (b). Any incoming peer list, even from *authentic* peers, should never be marked as *authentic,* because any peer may be taken over by defenders; (c). Peers with *high-CL*, *low-FC,* and from *authentic* peers are given priority of being chosen to replace the randomly-selected unqualified peers (i.e., with *high-FC*); (d). The proportion of *available* peers should never exceed 50% while *authentic* peers are permitted to occupy the whole peer-list. This design ensures that the *authentic* entries of peer-list always keep clean and unaffected even when enormous polluters coordinate to launch pollution attack aggressively.

II. Self-repairing Mechanism

The reputation-based mechanism keep the authenticity of peer-list, moreover, it is possible to form a self-repairing network since the partially polluted peer-list has great chances to recover via subsequent exchanging with benign peers. In case that a bot A publish its peer-list to bot B, B can choose peers with *high-CL*, *low-FC* to replace unqualified peers. Besides, if majority bots in peer-list are invalid or the *FC* value exceed a certain threshold (set by botmaster), B will actively request new peers from bots that it can connect to. Fig.3 illuminates the update procedure of E after it receives the pushed peer-list by A and C in turn. Fig.4 shows that F requests new peers from A to update invalid peer (shown as the blue entry).

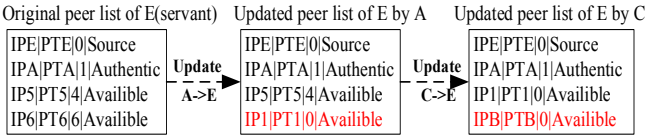

Fig. 3. Unauthenticated Peer-list Update via Exchange

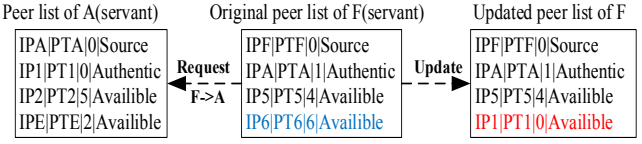

Fig. 4. Active Peer-list Update via Exchange

Two-step Bootstrap. This initial procedure of finding and joining a P2P network is usually called "bootstrap" procedure. Bootstrap is an essential part of network formation in P2P networks. In the proposed botnet, considering the extreme case that defenders could shut down all initial hard-coded servant bots shortly after botnet release, we design two-step bootstrap to provide the foundation for self-repairing: (a). Initially, it only hard-codes a reasonable amount of peers; (b). After reasonable delay, it exploits URL shortening services (USS) to build a stealthy and robust message transmission channel [5]. A list of newly compromised authentic servant bots which have been monitored by botmaster can be transmitted stealthily using this channel. When all initial hard-coded servant bots are shut down shortly after botnet release or all of peers in peer-list are invalid, the bot will initiate the second bootstrap procedure. USS can replace arbitrary URLs with shorter ones and subsequently redirect all requests for the shortened URL to the original URL, Some USSes (e.g., TinyURL) permit users to customize a short URL. Both bots and botmaster share the same SUGA (Short URL Generation Algorithm), the botmaster publishes encrypted peer information to Text Sharing Services (e.g., PasteBin, FreeTextHost) and get an original URL, then submits both the original URL and a customized shorten URL to USS. Once the URLs have been submitted successfully, the USS redirects all requests for the short to original URL. a bot can obtain new peers and join the botnet through enumerating the short URLs that are generated by SUGA until the correct one is identified. The USS based message transmission for bootstrap procedure is shown in Fig.5.

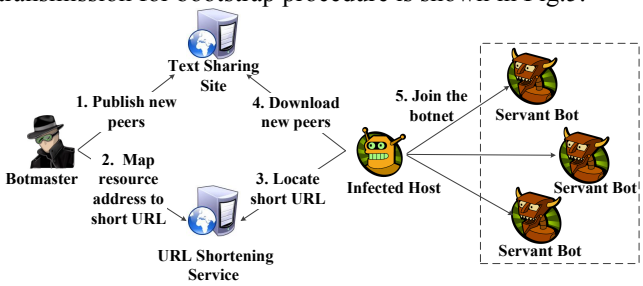

Fig. 5. USS based Message Transmission for Bootstrap Procedure

III. Conclusion and Future Work

In this work, we focus on how to solve the credibility issue amongst bots so that the peer-list can keep clean. We present a new type of hybrid P2P botnet that exploits a novel reputation-based and self-repairing mechanism to against peer-list-based pollution mitigation techniques. In future work, we will further study the robustness of the proposed botnet and mitigations design. The goal of our work is to increase the understanding of advanced botnet designs, such that more efficient detection and countermeasures can be developed.

Acknowledgement

This work is supported by the Ministry of Science and Technology of China (No. 2016QY08D1602).

References

[1] P. Wang, S. Sparks et al. An advanced hybrid peer to peer botnet. In Proc. of the First Workshop on Hot Topics in Understanding Botnets (HotBots'07), 2007.

[2] A.Sanatinia and G.Noubir, "Onionbots: Subverting privacy infrastructure for cyber attacks," in The Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2015.

[3] Rossow C, Andriesse D, Werner T, et al. Sok: P2pwned-modeling and evaluating the resilience of peer-to-peer botnets[C]//Security and Privacy (SP), 2013 IEEE Symposium on. IEEE, 2013: 97-111.

[4] Dittrich D. So You Want to Take Over a Botnet..[C]//LEET. 2012.

[5] Lee S, Kim J. Fluxing botnet command and control channels with URL shortening services[J]. Computer Communications, 2013, 36(3): 320-332.