SCIENCE CHINA
Information Sciences

• LETTER •

June 2020, Vol. 63 169301:1–169301:3
https://doi.org/10.1007/s11432-019-9921-7

Multi-variant network address hopping to defend stealthy crossfire attack†

Boyang ZHOU[1], Gaoning PAN[2], Chunming WU[1,2*], Kai ZHU[2] & Wei RUAN[3]

[1]*Zhejiang Lab, Hangzhou* 311121*, China;*
[2]*College of Computer Science, Zhejiang University, Hangzhou* 310027*, China;*
[3]*College of Control Science and Engineering, Zhejiang University, Hangzhou* 310027*, China*

Received 3 January 2019/Revised 20 March 2019/Accepted 18 June 2019/Published online 12 March 2020

Citation Zhou B Y, Pan G N, Wu C M, et al. Multi-variant network address hopping to defend stealthy crossfire attack. Sci China Inf Sci, 2020, 63(6): 169301, https://doi.org/10.1007/s11432-019-9921-7

Dear editor,

Recently, crossfire attack has been witnessed as a new distributed denial-of-service (DDoS) weapon that can effectively cut off the data connections between the chosen target area (w^d) of servers and the end hosts (H). The attack is launched by a network of bot (botnet) to drain bandwidth of persistent routes (PRs) in the indirect and low-rate data flooding towards the decoys that are located in upstream to the target, where the PRs are probed by the adversary who sends Internet control message protocol (ICMP) packets with different time-to-live (TTL) values, e.g., using traceroute [1]. Such flooding is hard to be detected by firewalls or IDSes.

In literature, the moving target defense [2] and blocking [3, 4] approaches can be bypassed by the botnet with the sophisticated behaviors as follows: (a) sending irregular ICMP packets with randomized TTL values, due to their similarity to benign packets [3]; and (b) randomly sending extremely low-rate packets, due to the hardness to adopt high sampling rate in large-scale networks [3, 4].

MVNAH design. Inspired by the cyber mimic defense [5], a novel multi-variant address hopping mechanism (MVNAH) is proposed which creates the variants for the protected decoys and the target with multiple random network addresses, in order to increase the uncertainties and puzzle the adversary on the PR discovery, degrading attack effect. Beneficially, the above botnet detection is not demanded comparing to the current approach.

MVNAH runs on an OpenFlow (OF) controller and interacts with OF switches (Y) via OF protocol v1.3 [6]. In running, MVNAH regularly detects the PR congestions via analyzing the negative correlation between the number of in-flight packets and received packets, to generate a set of PR states, denoted as $\hat{L} = \{(l_i, \kappa_i, e_i)\}$, where each state contains the link ID (l_i), PR congestion level (κ) and event counter (e). In defending, MVNAH firstly partitions $\hat{L}$ into the two subsets: (a) $\Lambda = \{l_i | l_i \in \hat{L} \wedge e(l_i) \geqslant \eta_{\min}\}$ for those congested PRs that are attacked less than $\eta_{\min}$, and (b) $\Theta = \hat{L} \setminus \Lambda$ for the rest PRs, i.e., when some of the disappeared PRs are congested via the previously created variants due to the non-changed sub-path between the variant and the decoys or the target. Then, MVNAH reacts to the attack as follows:

(1) For each PR in Θ, MVNAH generates and enforces the variants policy to create the multiple variants (ν) in the upstream of the PRs for either the protected decoys (w^c), the target (w^d) or their previous variants ($\nu'(w^{c,d})$), with random network addresses. Wherein, each variant works as the transparent proxy in end-to-end communication, with the two additional functions for hiding the PRs from the botnet's ICMP discovery process as follows: (a) blocking the ICMP reply messages

* Corresponding author (email: wuchunming@zju.edu.cn)
† The full version was awarded as an outstanding paper by 1st National Conference on Advanced Computing & Defense.

sent from the target to all hosts, and (b) replying ICMP message to hosts on behalf of the target.

(2) For each PR in Λ, MVNAH generates and enforces the traffic steering policy in the two ways of either: (a) diverging current paths into the new corresponding ones with randomly generated variants, if there is the optional path available for detouring around the PRs; or (b) suppressing their flooding rates without changing the previously created variants, if the optional path is unavailable.

Algorithmic implementation. The two types of policies are generated via the following algorithms:

(1) genVariants ($G, P_{\text{all}}, W^{c,d}, \Theta$) algorithm is given in Algorithm 1, which generates the new variants of $w_i^{c,d}$ or $\nu'(w_i^{c,d})$ according to detected $W^{c,d}$. Wherein, P_{all} is all the forwarding paths; $W^{c,d}$ is a set of decoys and the target attacked by bots; $\Omega\{(h_i, w_i)\}$ contains the set of pairs, where each pair represents w_i has a flow path connecting to $w_i^{c,d}$ in the data plane. In line 1, it declares a priority queue for conducting a breadth first search (BFS) in the later steps. In lines 3–22, it iterates each variant $w_i \in W^{c,d}$ to identify a set of corresponding OF switches to install $V_{w_i^{c,d}}$ (line 21). Line 4 provides a set of links of all the paths originating from $w_i^{c,d}$, denoted as $\tilde{L}$, and line 6 creates an array of $|N|$ elements all initialized with -1 at first. In lines 6–19, the BFS is conducted around $w_i^{c,d}$, where lines 9–19 propagate the PR IDs within all PRs in Θ along the trajectory from a parent node to its child node, so that only one of PR IDs closest to the leaf nodes among the multiple PRs in the same path is propagated to the leaf node. As a result, $|V|$ is minimized (in line 23) due to the maximum separation of bots from connecting to the previous $w_i^{c,d}$ or $\nu'(w_i^{c,d})$, eliminating all the congested PRs targeting to $\{\nu(w_i^{c,d})\}$.

(2) genTrafficSteerings ($P_{\text{all}}, W^{c,d}, \Lambda$) algorithm (in Algorithm 2) generates the traffic steering policy according to $G, P_{\text{all}}, W^{c,d}, \Lambda$. In line 1, it returns $\emptyset$ if no link for traffic steering. Lines 3–11 computes the specific policies for $w_i^{c,d} \in W^{c,d}$. Initially, line 4 try to compute the diverged paths to $G(N, L \setminus \Lambda)$ originating from $w_i^{c,d}$ with the bandwidth and delay constrains (Ξ) using Dijsktra algorithm. If no path can be found for a pair $(w_i^{c,d}, w_j)$, $(w_i^{c,d}, w_j)$ is added to $\Upsilon.S$ (line 6) for the traffic suppressing. Otherwise, the pair is added to $\Upsilon.S$ (lines 8 and 9) for the traffic rerouting, where $w_{\text{dest}}^{c,d}(p_j)$ is the destination node of p_j, and $\nu_{\text{rand}}(w_{\text{dest}}^{c,d}(p_j)))$ is a function to generate a new variant of $w_{\text{dest}}^{c,d}(p_j)$ in a random intermediate switch in p_j. Finally, it returns $(\Upsilon.R, \Upsilon.S)$ as the traffic steering rules.

Finally, the reacting policy for the attack is returned as $\Upsilon(V, R, S)$ for latter enforcement. Further, the smaller value of $\eta_{\min} \geqslant 2$ reflects a better sensitivity for identifying the event of flooding on disappeared PRs, which is configured by needs.

Algorithm 1 genVariants

Require: $G, P_{\text{all}}, W^{c,d}, \Theta$.
1: Q = PriorityQueue(), $N = Y \cup H$, torch = int[$|N|$];
2: visited = boolean[$|N|$];
3: **for** $w_i^{c,d} \in W^{c,d}$ **do**
4: $\tilde{L} = \{l_k | l_k \in P(w_i^{c,d})\}$;
5: Q.add($w_i^{c,d}$);
6: torch[$1 \cdots$] $= -1$, visited[$1 \cdots$] = false;
7: **while** $|Q| > 0$ **do**
8: $u = Q$.poll();
9: **for** $l_j \in$ edges(u) **do**
10: v = peerNode(u, l_j);
11: **if** $l_j \notin \tilde{L} \vee$ visited[v] **then** continue;
12: **if** $l_j \in \Theta$ **then**
13: torch[v] $= l_j$;
14: **else**
15: torch[v] = torch[u];
16: **end if**
17: torch[u] $= -1$;
18: Q.add(v), visited[v] = true;
19: **end for**
20: **end while**
21: $V_{w_i^{c,d}} = (w_i^{c,d}, \{s_j | \text{torch}[j] \neq -1\})$;
22: **end for**
23: **return** $\Upsilon.V = \cup_{w_i^{c,d} \in \widetilde{W^{c,d}}} V_{w_j^{c,d}}$.

Algorithm 2 genTrafficSteerings

Require: $G, P_{\text{all}}, W^{c,d}, \Lambda$.
1: **if** $\Lambda = \emptyset$ **then return** $\emptyset$
2: $\Upsilon.R = \emptyset, \Upsilon.S = \emptyset$, $N = Y \cup H$;
3: **for** $w_i \in W^{c,d}$ **do**
4: $P'(w_i) = \text{shortestPaths}(G(N, L \setminus \Lambda), w_i, \Xi)$;
5: **for** $w_j \in W$ **do**
6: **if** $P'(w_i, w_j) = \emptyset$ **then** $\Upsilon.S = \Upsilon.S \cup \{(w_i, w_j)\}$;
7: **end for**
8: $\widetilde{R} = \{p_j | p_j \in P'(w_i, W) \wedge (w_i, W) \notin \Upsilon.S\}$;
9: $\Upsilon.R = \Upsilon.R \cup \{(p_j, \nu_{\text{rand}}(w_{\text{dest}}^{c,d}(p_j))) | p_j \in \widetilde{R} \wedge \nu_{\text{rand}}(w_{\text{dest}}^{c,d}(p_j)))\}$;
10: **end for**
11: **return** $(\Upsilon.R, \Upsilon.S)$.

Overall, the execution time of the policy generation of MVNAH (Algorithms 1 and 2) is with the complexity in the different cases as follows: (a) in both the expected and best cases, $O[|W^{c,d}| \cdot |N| \cdot (1+g) + |W^{c,d}| \cdot \log|N| \cdot (1 + g \cdot |N|)]$; and (b) in the worst case, $O[|W^{c,d}| \cdot |N| \cdot (1+g) + |W^{c,d}| \cdot \log|N| \cdot (1 + g \cdot |N|) + g \cdot |W^{c,d}| \cdot |N|^2]$. Wherein, $g = \Pr\{|\Lambda| \geqslant 1\}$ (see line 1 of Algorithm 2) which correlates with the attacking policy of the bots whether to sustain flooding regardless of the new variants. The space complexity is with $O(|W^{c,d}| \cdot \log|N| \cdot (1 + g \cdot |N|) + |N|)$ for all cases. Hence, the algorithm is with the scalability in both execution time and space. For overhead incurred by MVNAH, the number of generated variants is in $O(|W^{c,d}|^2 / \log(|W^{c,d}|))$, hence, only relevant to

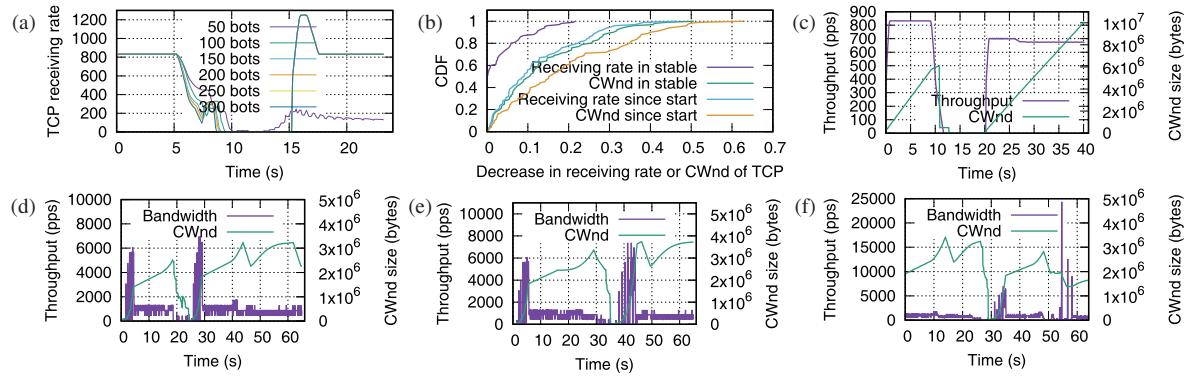

Figure 1 (Color online) Defense effectiveness evaluation of MVNAH. (a), (b) and (c) are the TCP packet receiving rate and congestion window (CWnd) size changes at the target side, tested in NS-3 with $|Y| = 2000$, where the variants and suppressing policies are enforced at 15 s in (a) and between 10–20 s in (c), respectively; (d), (e) and (f) are tested in Mininet with $|Y| = 50$. (a) rate changes for variants; (b) decreased rate changes for reroute; (c) rate changes for suppressing; bandiwdth & CWnd changes for (d) variants, (e) reroute, and (f) suppressing.

the number of the attacked decoys and the target, which is suitable for large scale networks.

Policy enforcement. Υ is enforced at switches completely using the OF protocol v1.3. The variant is installed with the rules of the bi-directional network address translations between $\nu(w_i^{c,d})$ and $w_i^{c,d}$, with the two sequential instructions of the OFPAT_SET_FIELD and OFPAT_OUTPUT [6], in a light-weight manner. At last, MVNAH notifies new variants' addresses to the hosts that then reconnect to decoys or the target for restoring their communications to non-congested states. The number of total control messages required is in $O[|W^{c,d}|^2/\log(|W^{c,d}|)+(1-h)\cdot\log|N|\cdot|W^{c,d}|+h\cdot|W^{c,d}|]$, where $0 < h \leqslant 1$, indicating a scalable message complexity in the policy enforcement.

Defense effectiveness. After the policy enforcement, the botnet is hard to identify previously congested PRs due that: (a) there is no PR in discovered routing paths, since the variants are installed before the PRs; and (b) the sub-pathes between $\{\nu(w_i^{c,d})\}$ and $w_i^{c,d}$ are invisible for the host or bot, and the ICMP packets are replied from the variant with faked TTL values. Thus, the attack is difficult to be launched in next round.

Performance evaluation. The MVNAH is evaluated in both the OF switch simulator in NS-3 [7] and Mininet emulator [8] with the random topology synthesized in the Erdős-Rényi model. The crossfire attack is simulated according to [1], with the sizes of $|Y| = 2000$ and $|H| = 2000$. A certain number of bots (ranging from 5–300) is assigned to flood on the corresponding decoy in rolling manner at the rate of 1.719 kB/s via On-Off model for saturating the PR bandwidth, in order to simulate a sophisticated attacking scheme. Thus, with the MNVNAH defense in $\eta_{\min} = 2$, Figures 1(a)–(c) and (d)–(f) show the TCP performance results evaluated in the NS-3 and Mininet, respectively, demonstrating that: (a) with $\Upsilon.V$, bots are puzzled for launching the attack due to the invisible PRs, and then the PRs are in available state again; (b) with $\Upsilon(R, S)$, the PRs are restored to available state between hosts and the target. The above results verify the defense effectiveness of MVNAH.

The details of MVNAH can be found in [9].

Acknowledgements This work was supported by National Key Research and Development Program of China (Grant Nos. 2016YFB0800102, 2017YFB0803205), Key Research and Development Program of Zhejiang Province (Grant Nos. 2017C01064, 2017C01055, 2018C01088), and Fundamental Research Funds for the Central Universities (Grant No. 2016XZZX001-04).

References

1 Kang M S, Lee S B, Gligor V D. The crossfire attack. In: Proceedings of IEEE Symposium on Security and Privacy, Berkeley, 2013. 127–141
2 Venkatesan S, Albanese M, Amin K, et al. A moving target defense approach to mitigate DDoS attacks against proxy-based architectures. In: Proceedings of IEEE Conference on Communications and Network Security (CNS), Philadelphia, 2016. 198–206
3 Wang J, Wen R, Li J Q, et al. Detecting and mitigating target link-flooding attacks using SDN. IEEE Trans Depend Secure Comput, 2019, 16: 944–956
4 Zheng J, Li Q, Gu G, et al. Realtime DDoS defense using COTS SDN switches via adaptive correlation analysis. IEEE Trans Inform Forensic Secur, 2018, 13: 1838–1853
5 Hu H, Wu J, Wang Z, et al. Mimic defense: a designed-in cybersecurity defense framework. IET Info Secur, 2018, 12: 226–237
6 Open Networking Foundation. OpenFlow specification. v1.3. 2012. https://www.opennetworking.org
7 Riley G F, Henderson T R. The NS-3 network simulator. In: Modeling and Tools for Network Simulation. Berlin: Springer, 2010. 15–34
8 Lantz B, Heller B, Mckeown N. A network in a laptop: rapid prototyping for software-defined networks. In: Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, New York, 2010
9 Zhou B, Gao P, Wu C, et al. Multi-variant network address hopping to defend stealthy crossfire attack. In: Proceedings of the 1st National Conference on Advanced Computing and Defense, 2018. 540–556