# CRESTBOT: A New Family of Resilient Botnets

Duc T. Ha, Hung Q. Ngo, Madhusudhanan Chandrasekaran
Department of Computer Science and Engineering
State University of New York at Buffalo
Amherst, NY 14260, USA
Email: {ducha,hungngo,mc79}@buffalo.edu

***Abstract*—We show that it is possible to design botnet structures called CRESTBOT based on extractor graphs which are highly resilient to command-and-control (C&C) take-downs, yet do not require significant changes to existing botnet designs and codes, and do not suffer from the implementation complexity of P2P-based and hybrid structures. The UDP family of CRESTBOT is shown to be able to send commands from the botmaster much faster than traditional botnet. Our analyses are validated by extensive experiments on Emulab.**

**Our results prove that current C&C-takedown solutions are ineffective against well designed botnets such as our CRESTBOT. Secondly, short UDP commands can be as reliable as TCP commands with much less time consumption. Third, extremely fast command issuing is possible, which at first glance might seem beneficial to the attacker; however, it might also be of use for the "good guys" when certain race conditions are desired such as software patching or quick bot takedowns.**

***Index Terms*—CRESTBOT, bot, botnet, resiliency, expander**

## I. Introduction

Bot networks (botnets) are an extremely serious problem of today's networking infrastructure, economy, and national security. These networks of large numbers of compromised machines (bots) are used by attackers (botmasters) to profit from DDoS attacks, spam relaying, identity theft, click fraud, malwares, etc. Industry estimates have suggested that botnets have become ever-so prevalent involving more than 10 million Internet hosts [1]

Most current botnets employ a hierarchical design with the botmaster controlling the compromised machines using Internet Relay Chat (IRC). Commands are issued by the botmaster to a small number of specially designated nodes called command-and-control (C&C) nodes where they are relayed to the remaining bots. The botmaster's identity is thus hidden behind a layer of C&C, making source-traceback very difficult.

Recent defense solutions have focused on C&C node detection and take-down to disrupt and limit botnets' scope. A C&C node taken down means all the bots subscribing to it are pruned, thereby reducing the botnet's size and capacity. Moreover, as each C&C node serves a relatively large number of bots, it is possible to track them using honeypots and DNS-based analysis.

Consequently, sophisticated botnet variants have recently emerged which adopt peer-to-peer (P2P) structure using some well known P2P-based protocol such as Kademlia [2], Gnutella, etc. Unlike C&C-based botnets, P2P-based botnets are more resilient to node failures. Compromising a node only gives out its neighbors. Though effective, direct adoption of P2P-based botnets is not straightforward due to the complex and resource-demanding nature of P2P protocols. A hybrid P2P-based botnet has also been proposed [3] which draws on the advantages of P2P-based protocol while having C&C like nodes to assist in faster and efficient command propagation. However, implementation of such a hybrid approach is rather unconventional, requiring significant changes to the existing botnet design and code.

We will show that it is possible to design botnet structures which are highly resilient to C&C take-downs, yet do not require significant changes to existing botnet designs and codes, do not suffer from the implementation complexity of P2P-based and hybrid structures. Our proposed structure is named CRESTBOT, for C&C-based RESilienT Botnet. Resiliency often requires some degree of redundancy, which may demand other resources such as time and space. We will show that the space requirement of CRESTBOT is negligible; and, more importantly, it is possible to keep command broadcast time almost as fast as non-resilient structures by using UDP commands instead of TCP as before. CRESTBOT is designed based on *extractors*, a kind of expanding graphs very pervasive in computer science nowadays [4]. We verify our results both analytically and experimentally using Emulab [5].

The drive-home points of this research are as follows. First, C&C-takedown solutions are ineffective against CRESTBOT, which is very simple to implement and requires minimal changes to existing botcodes. Second, with properly designed dissemination structures, short UDP commands can be as reliable as TCP commands with much less time consumption. This could potentially be used in other areas of networking. Third, extremely fast command issuing is possible, which at first glance might seem beneficial to the attacker; however, it might also be beneficial to the "good guys" when certain race conditions are desired (patching, bot takedowns, etc.).

The remainder of the paper is organized as follows. Section 2 reviews selected related works. Section 3 analyzes the resiliency of existing C&C-based botnet and proposed new structures with strong resiliency guarantee. In Section 4, we experimentally validate the analyses. Finally, we conclude and discuss defense approaches drawn from our work in Section 5.

## II. Selected Related Work

Existing studies on botnets fall under the following two main categories: (i) Perform case-study of botnet behaviors and structures; (ii) Propose techniques for botnet detection and disruption. For example, Freiling et al. [6] infiltrated a real botnet to identify C&Cs and study bot commands. Rajab et al. [7] employed a multifaceted and distributed infrastructure to study botnet behaviors. Others works focused on measuring botnet size by various techniques such as by DNS redirection [8] or DNS cache snooping [9]. Recently, several techniques have also been proposed to detect botnet existence: anomaly detection [10], traffic or network activity statistics analyses [11].

To the best of our knowledge, this paper is the first to analyze traditional botnet and their variances' resiliency and show how command issuing can be made very fast. Incidents from the past have showed us that malicious programs always evolve into instances with more and more advanced features and techniques (the UDP worm Slammer is a good example). In fact, botnets with more resilient topologies using P2P structures have already arisen [12]. Research on hypothetically powerful botnets such as ours is thus important to prepare for the worst, to develop, test, and benchmark counter-measure systems in advance.

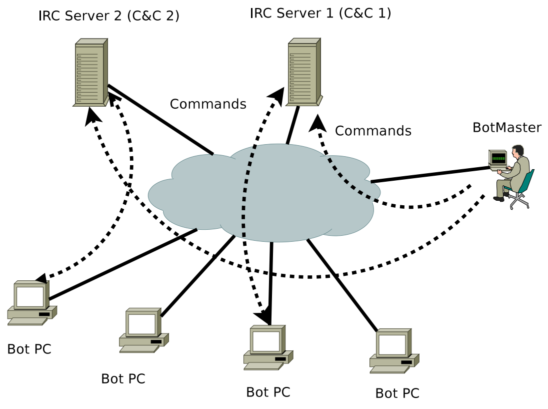
## III. Botnet Resiliency and crestbot

### A. Resiliency of a Traditional Botnet Topology

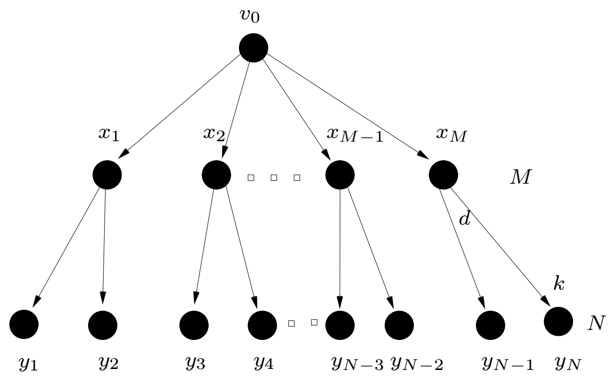
For a traditional C&C botnet (Figure 1(a)), the botmaster first infects (directly or indirectly) vulnerable PCs with malicious softwares called bots. Each bot then contacts a central server through which the botmaster relays commands as well as collects feedback information from the bots. The C&Cs can either be public servers (often Internet Chat Protocol or IRC servers) or dedicated machines set up by the botmaster to function as C&Cs. Depending on how the botmaster commands reach the bots, botnets can also be categorized into "push" or "pull" style. In a push style C&C, the C&Cs forward commands from the botmaster to the bots in real time, while in a pull style, bots contact the C&C for new commands. Most common bot families such as AgoBot, PhatBot, GTBot [6] belong to this category. HTTP-based botnets on the other hand are pull-style botnets, where bots query (e.g using GET and POST) a web server for new commands from the botmaster.

Whichever the botnet's type is, its resiliency can be measured by the expected number of bots who still receive commands (in push-style botnets) or still can reach the botmaster (in the pull-style botnets) given the probability that C&Cs are down (or the fraction of C&Cs which are down). C&C based botnets can be viewed as a 3-level tree topology shown in Figure 1(b). Node $v_0$ represents the botmaster, $x_i, 1 \leq i \leq M$ represents the C&Cs and $y_j, 1 \leq j \leq N$ represent the bots.

For the sake of clarity, we assume that each C&C is in charge of $d$ bots, though our analysis applies for any C&C degree. Let $p_r$ be the probability that a C&C is taken down, the expected number of unreachable bots is then $N(1 - p_r)$.



(a) Typical Botnet Scenario



(b) Modeled Topology

Fig. 1. Typical and Modeled C&C Botnet Topology

This traditional topology is thus not very resilient: the number of affected bots is linearly proportional to $1-p_r$. An adversary (e.g a system administrator or security analyst) with the right removal strategy of the C&Cs (e.g. remove heavily loaded C&C) can isolate a major portion of the botnet. Since the number of C&Cs is much smaller than the number of bots, an adversary then needs relatively little effort to cripple a large botnet.

### B. First Attempt at a More Resilient Structure

One potential way to improve resiliency is to employ P2P topologies or to rely on existing P2P networks. For example, Peacomm [12] employs a P2P network for its C&Cs. However, P2P protocols impose extra routing and maintenance overhead, making the botnet more prone to being detected. On the other hand, the communication supportive nature of IRC, the availability of public IRC servers or web servers make it much easier for the botmaster to organize a C&C-based botnet. We thus believe C&C-based botnets and the likes will remain popular in the near future.

To increase the traditional structure' resiliency, a natural approach is to increase the number of paths from the root to each bot. It is thus sensible to let each bot connect to $b$ C&Cs. Let $Z$ denote the number of reachable bots, then

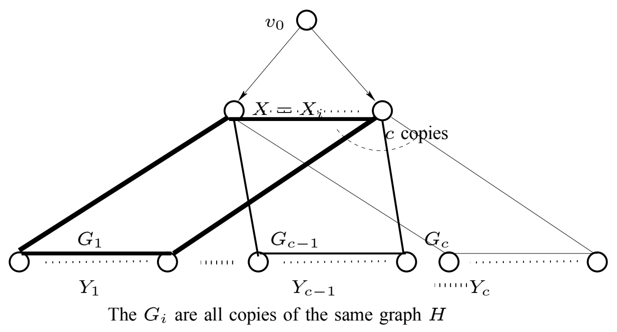


Fig. 2. The Resilient Topology $R$

$E[Z] = N(1-p_r^b)$. The negative effect of removed C&Cs on the number of bots is now reduced exponentially while the botcode changes required are minimal.

In order to guarantee that $(1-\epsilon)N$ bots are reachable with probability at least $(1-\delta)$, how large should $b$ be? The distribution of $Z$ is binomial with parameter $(1-p_r^b)$. Thus, it will be concentrated around the mean. To guarantee that $Z > (1-\epsilon)N$ with high probability, we thus will need $p_r^b$ to be quite a bit less than $\epsilon$. More concretely, by Chernoff bound

$$\text{Prob}[Z < (1-\epsilon)N] = \text{Prob}\left[Z < \tfrac{(1-\epsilon)}{(1-p_r^b)}(1-p_r^b)N\right]$$
$$< \exp\left\{-\tfrac{N}{2}(1-p_r^b)\left(\tfrac{\epsilon-p_r^b}{1-p_r^b}\right)^2\right\} = \exp\left\{-\tfrac{N}{2}\tfrac{(\epsilon-p_r^b)^2}{1-p_r^b}\right\}$$

which is at most $\delta$ if, after simple calculus,

$$p_r^b \le \epsilon - \frac{1}{2N}\ln(1/\delta) - \sqrt{(1-\epsilon)\frac{1}{N}\ln(1/\delta) + \frac{1}{4N^2}\ln^2(1/\delta)}$$

This guarantee may not even be possible with parameters for which

$$\epsilon \le \frac{1}{2N}\ln(1/\delta) - \sqrt{(1-\epsilon)\frac{1}{N}\ln(1/\delta) + \frac{1}{4N^2}\ln^2(1/\delta)}$$

(like when $\epsilon$ is relatively small). Even when $\epsilon$ is not too small, the required value of $b$ may be too large to be useful practically. Firstly, the physical capacity of C&Cs is usually limited. For example, a chat channel typically hosts around a hundred users or less, and a chat server normally handles only several thousand users. Secondly, higher values of $b$ increase the risk of being detected for the bots. We will next design a more resilient structure to provide the above "epsilon-delta" guarantee.

## *C. The extractor-based* CRESTBOT *design*

A $(w,\epsilon)$-*extractor* is a bipartite graph $H = (L \cup R, E)$ with left part $|L| = M \ge w$ and right part $|R| = y$, such that every subset of at least $w$ left vertices has at least $(1-\epsilon)y$ neighbors on the right. It is known that, for any $\epsilon$ and any $w \le M$, there exist $(w,\epsilon)$-extractors in which all left vertices have degree $d = \Theta(\log M)$, $y = \Theta(wd)$, and the distribution of right degrees are close to uniform, i.e. of degree $Md/y = \Theta(M/w)$ [13].

We now construct a new resilient structure $R$ (Figure 2) as follows: for $i = 1,\dots,c$, let $G_i = (X_i \cup Y_i; E_i)$ be copies of the same $(w,\epsilon)$-*extractor* $H$. To construct $R$, we "glue" together the $X_i$-side of $G_1,\dots,G_c$, i.e. identify vertices in the $X_i$ in any one-to-one manner. Let $X$ denote the glued $X_i$, and $Y = Y_1 \cup \cdots \cup Y_c$. The source $v_0$ has an edge to each vertex in $X$. In total, the vertex set of $R$ is $\{v_0\} \cup X \cup Y$. The following theorem states the resiliency of this structure:

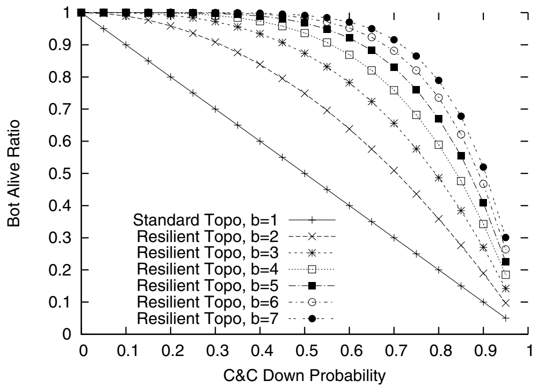


Fig. 3. Resiliency

*Theorem 1:* Fix any constant $\epsilon > 0$. Let $\alpha > 0$ be any constant such that $\alpha > p_r$. Set $w = (1-\alpha)M$. Then with probability at least $1 - \exp(-\Theta(M))$ (i.e. exponentially close to 1), where $M$ is the total number of C&Cs, a $(1-\epsilon)$-fraction of all bots will remain reachable in the structure $R$.

*Proof:* The important point to notice here is that, by definition of extractors, if less than a $(1-\epsilon)$-fraction of bots are reachable, then more than $M - w = \alpha M$ C&Cs are down. Let $S$ be the number of unavailable C&Cs, then, noting that $\text{E}[S] = Mp_r$,

$$\text{Prob[less than } (1-\epsilon)N \text{ bots reachable]} \le \text{Prob}[S > \alpha M]$$
$$= \text{Prob}\left[S > \left(1 + \tfrac{\alpha - p_r}{p_r}\right)p_r M\right]$$
$$\le \exp\left\{-M(\alpha - p_r)^2/(3p_r)\right\}$$

■

Figure 3 plots the resiliency of a topology $R$ for $M = 20, N = 600$ with various right degree value $b = \frac{Md}{N}$. Here the corresponding ratios of alive bots are computed using the worst case value. At $b = 1$, $R$ degrades back to the traditional topology where $p_r$ is inversely proportional to the ratio of the remained bots. As $b$ increases, the impact of $p_r$ is reduced exponentially. In particular, for $b = 5$, when 80% C&Cs are down, only 30% of bots are affected, compared to 80% of the traditional topology. At $b = 7$, more than 90% of the bots are still reachable even when up to 70% of the C&C are down.

## IV. A New Class of UDP BotNet

We define CRESTBOT (C&C-based RESilienT Botnets) as a family of botnets using the above topology construction. The existence of resilient topologies give rise to a potential class of CRESTBOT using UDP as the underlying transmission protocol. UDP CRESTBOT employ resilient topologies to compensate any potential loss caused by using UDP. UDP offers two powerful weapons: speed and flow stealth against defense systems, especially those relying on stable flow information [11]. We name such botnets as UCRESTBOT. In the following sections, we will evaluate and compare the broadcast time of UCRESTBOT with that of a tradition TCP botnet. The broadcast time is defined as the amount of time from when the botmaster sends the first bit of a message until the last bit of that message disappears from the network.

*1) Lossless Transmission:* To simplify the estimation, we consider the case when the bandwidth of each link is uniform, denoted as $r$. We also assume the latency of all links in each layer (between the botmaster and C&Cs, between C&Cs and bots) are the same, denoted as $L_1, L_2$ respectively. Denote the length of a message as $m$, and denote the message broadcast time as $T_{UCRES}$. Each C&C can send any message to its subscribed bots in a consecutive manner without little or no delay between each transmission. The following lemma states the expected message broadcast time of UCRESTBOT.

*Lemma 2:* Let $Q = \frac{m}{r} + L_p$. The expected broadcast time of a UCRESTBOT can be approximated as:

$$E[T_{UCRES}] = Q\left[M + \frac{p_r(p_r^M - 1)}{1 - p_r}\right] + (1 - p_r^M)(dQ + L_1 + L_2). \quad (1)$$

Here $d$ is the left degree of each C&C as defined in the previous section, $L_p$ is the processing delay at each node.

*Proof:* Let $x_I$ be the last C&C that remains at the second layer. $I$ is a random variable with probability Prob$[I = j] = p_r^{M-j}(1-p_r)$. Denote $Z_I$ as the time the last bot connected to $x_I$ receives the message. The broadcast time of UCRESTBOT is: $T_{UCRES} = \begin{cases} Z_I & \text{if } I > 0, \\ Z_0 = 0, & I = 0 \end{cases}$. We then have:

$$\begin{aligned} E[T_{UCRES}] &= \sum_{j=1}^{M} Z_I \,\text{Prob}[I = j] \\ &= \sum_{j=1}^{M} (j.Q + d.Q + L_1 + L_2) p_r^{M-j}(1 - p_r) \\ &= Q\left[M + \frac{p_r(p_r^M - 1)}{1 - p_r}\right] + (1 - p_r^M)(dQ + L_1 + L_2) \quad (2) \end{aligned}$$

■

We now compute the broadcast time for a TCP botnet using the traditional topology, denoted as $T_{TCP}$. In general, the overall end-to-end delay for a TCP link is dependent of the specific features of the underlying TCP stack such as window size, acknowledgment method, congestion control algorithm etc. However, we observe that most bot commands are **short** (less than a few hundred bytes) thus can fit in a transmission window or even in one data segment. Since we are not considering packet loss, a single message end-to-end delay will consist mainly of one direction propagation delay and the Round Trip Time for the connection setup. In addition, we assume that the C&C servers can multiplex simultaneous connections during the connection setup (e.g by using multiple threads, a common technique used by many worms) to reduce the waiting time. The result in the previous lemma can then be modified to compute the message broadcast time for a traditional TCP botnet:

*Lemma 3:* Define $Q$ as before. The expected propagation time for TCP is:

$$E[T_{TCP}] = Q\left[M + \frac{p_r(p_r^M - 1)}{1 - p_r}\right] + (1 - p_r^M)(dQ + 3L_1 + 3L_2). \quad (3)$$

Note the coefficient 3 associated with each $L_1, L_2$ term come from the three way handshake process in TCP.

*2) Lossy Transmission:* Denote $p_e$ as the packet loss rate on each link. Since UDP provides no retransmission, the end-to-end delay for each successful connection remains the same. The new expected broadcast time of CRESTBOT can then simply be upper bounded (it can be smaller when there's no transmission at the second layer) as in (1),(3) with a new failure probability $p_a = p_e + p_r - p_e * p_r$ . The delay for TCP botnet however is fairly complicated. We borrow the well established TCP model in [14] to estimate the delay of a short TCP connection. The expected delay of a short TCP connection over a link with latency $L$ is modeled as:

$$E[T_{con}(L)] = E[L_h] + E[T_{ss}] + E[T_{loss}] + E[T_{ca}] + E[T_{delack}] \quad (4)$$

where $E[L_h], E[T_{ss}], E[T_{loss}], E[T_{ca}]$ and $E[T_{delack}]$ are the expected value of handshake, slow start, retransmission time-outs or fast recovery, remaining data transmission, and delayed acknowledgment time respectively. The exact formula of each term can be found in [14]. The expected time for TCP can then be lower bounded as:

$$\begin{aligned} E[T_{TCP}(p_e)] &> (1 - p_r^M)\{T_{TCP}(L_1)(1 - p_e) + p_e E[T_{con}(L_1)]\} \\ &+ (1 - p_r^M)\{T_{TCP}(L_2)(1 - p_e)^d + [1 - (1 - p_e)^d] E[T_{con}(L_2)]\} \quad (5) \end{aligned}$$

where $T_{TCP}(L) = \frac{M}{r} + 3L$ is the approximated delay of a short TCP link with latency $L$ when there is no packet loss.

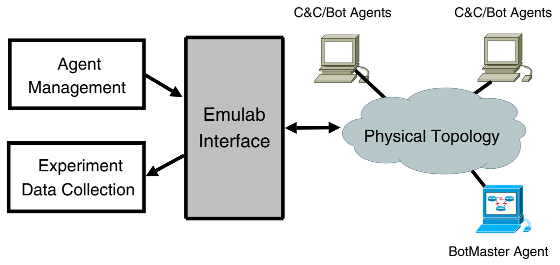
## V. EXPERIMENT

To evaluate the accuracy of our estimation, we conducted extensive experiments on Emulab [5]. Emulab is a network testbed that offers realistic experiments with both actual hardware and software platforms. Compared to traditional simulation methods, Emulab provides a testing environment with much higher fidelity to the real world.

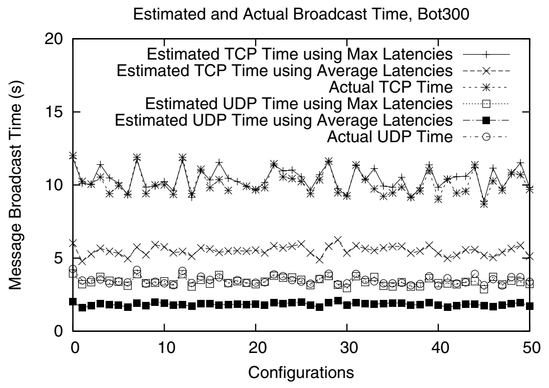
### A. Experiment Setup

In practice, the effective size (the number of live bots) for a botnet is just around several thousands, as pointed out by Rajab et al [9]. However, Cooke [15] noted that botnets of size a few hundreds are also common because they are difficult to detect and easier to sell or rent. We hence consider two small botnets of size 300 and 600, with 10 and 20 C&Cs respectively. Note that in this setting a C&C controls 30 bots on average. For this setting, the set of C&Cs is roughly 3% of the bot population (Wang's resilient topology [3] requires 25%). In the next parts of this section, we will refer to these botnets as Bot300 and Bot600.

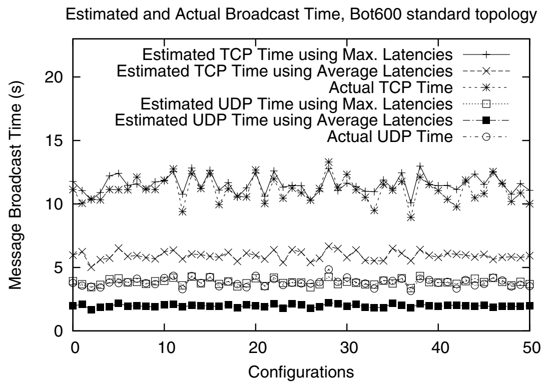
For the physical network topology, we used BRITE [16], a universal topology generator designed to synthesize realistic network topologies. BRITE supports multiple generation



(a) Experiment Setup



(b) Estimated and Actual Broadcast Time, Bot300



(c) Estimated and Actual Broadcast Time, Bot600

Fig. 4. Experiment setup and results

models for flat AS, flat Router and hierarchical topologies. In each topology generated by BRITE, nodes are grouped into multiple ASes and communicate through inter-AS links, which are generated in resemblance to their Internet counterparts.

Figure 4(a) illustrates the setup of our experiment. We developed special C&C and Bot agents to simulate C&C and Bot's communication activities. The agents are randomly distributed on different machines to simulate the variation of network topologies. For each round, the Agent Management Component first deploys agents following the resilient overlay topology, then instructs BotMaster Agent to send commands to C&C Agents, which in turn forward the commands to the Bots. After each round, experiment logs are collected from C&C and Bot Agents by the Data Collection Component.

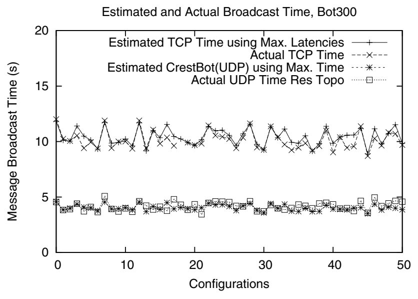
## B. Results

We first validate the equations (1),(3) established in the previous section. Recall that we assumed uniform bandwidth and latency on all the links between the botmaster and the C&Cs, as well as between the C&Cs and the bots. Since these assumptions rarely hold in practice, one can relax them by using the average values (computed over all latencies at each layer). However, our experiment results show that it can lead to underestimation when the latencies varies widely and the broadcast time can be dominated by the largest latencies at each level. As can be seen from Figure 4(b,c), estimated broadcast time using average latency values is significantly smaller than the actual time. On the other hand, estimation using the maximum values yields very close results. This holds true for both Bot300 and Bot600.

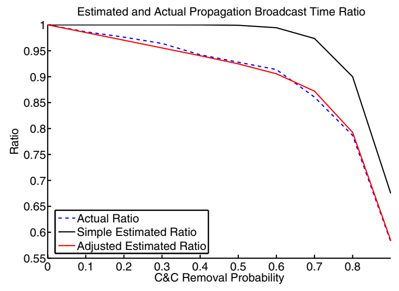
We next validate the estimated (using maximum latencies) and actual broadcast time for a TCP botnet with traditional topology and UCRESTBOT. We use a Bot300 for the experiment and use $L_p = 4$ ms for our estimation. Figure 5(a) shows the estimated and actual time for these botnets. It can be seen that UCRESTBOT broadcast time is only one third of the counterpart in a traditional TCP botnet.

We also validate the estimation of UCRESTBOT in the formula (1) when $p_r$ varies. Figure 5(b) shows the estimated and actual ratio $E[T_{UCRESTBOT}]$ when $p_r$ varies over the case when $p_r = 0$. It can be seen that the actual ratio decrease much faster than the estimated ratio using (1) and the estimated ratio is not accurate. Upon closer examination of Emulab mechanism, we observe that $L_p$ on Emulab depends on the number of network connections and thus on $p_r$ almost linearly. After adjusting $L_p$ by a factor of $1 - p_r$, the estimated ratio becomes very close to the actual result.

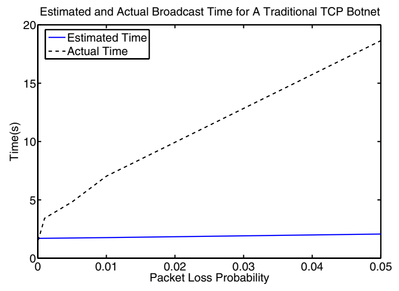
For lossy conditions, we first ran experiments on Emulab using the packet loss simulation provided by Emulab. However, the data collected showed abnormal delay compared to existing TCP models in literature. For example, Figure 5(c) shows the gap between the average single connection delay at the first layer of a traditional TCP Botnet300 and its estimated value according to [14]. This additional delay is due to the extra processing delay incurred by the packet loss simulation in Emulab. This delay however does not appear under a lighter packet loss simulation setup. To confirm this, we measured the TCP connection delay between 2 machines running FreeBSD, connected by a direct link with bandwidth 10Mbps, propagation latency 600ms. Figure 6(a) shows the actual and estimated TCP delay under this setup. It can be seen that the model in [14] agrees closely with the actual results. Finally, Figure 6(b) shows the ratio between a TCP Bot300 and UCRESTBOT Bot300 broadcast time when the packet loss rate $p_e$ varies, estimated using (5). As $p_e$ increases, the ratio also increases almost linearly from 3 during the displayed $p_e$ range. As $p_e$ reaches higher values, the ratio will follow the same trend in Figure 6(a).



(a) Traditional and UCRESTBOT Broadcast Time

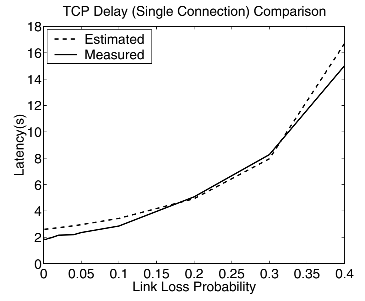


(b) Actual, Estimated Time Ratio of UCRESTBOT

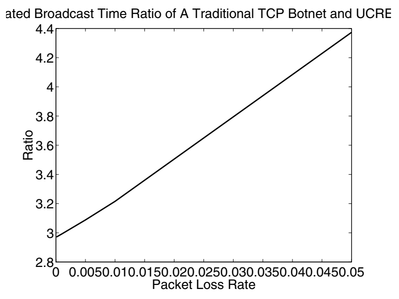


(c) Actual and Estimated TCP Delay

Fig. 5. Message broadcast time and delay validation



(a) Single TCP Connection Delay Validation



(b) Broadcast Time Ratio UCRESTBOT and TCP Bot300

Fig. 6. TCP delay validation and estimated time ratio

## VI. Conclusions

In this paper we presented a hypothetical family of botnets named CRESTBOT with resilient topologies based on extractors. We showed that CRESTBOT are able to guarantee a high level of resiliency, yet do not require substantial modification to the current C&C-based structures. We also studied a special UDP member of the CRESTBOT family, UCRESTBOT, capable of sending commands from the botmaster at least three times faster than the traditional botnet. UCRESTBOT and CRESTBOT pose considerable challenge to system administrators by their resiliency. To defense against these botnets, early intervention seems to be the only effective method. This is because most bots are initially setup to connect to some seed C&Cs and thus are vulnerable during this phase. Early detection and removal of these seeds are then critical to cripple these botnets. Techniques as discussed in the related work section can then be applied accordingly.

## References

[1] http://www.mcafee.com/us/local_content/white_papers/wp_botnet.pdf.
[2] Maymounkov and Mazieres, "Kademlia: A peer-to-peer information system based on the XOR metric," in *International Workshop on Peer-to-Peer Systems (IPTPS), LNCS*, vol. 1, 2002.
[3] P. Wang, S. Sparks, and C. C. Zou, "An advanced hybrid peer-to-peer botnet," *Proceedings 2007 USENIX First workshop on Hot Topics in Understanding Botnets*, Apr. 2007.
[4] S. Hoory, N. Linial, and A. Wigderson, "Expander graphs and their applications," *Bull. Amer. Math. Soc. (N.S.)*, vol. 43, no. 4, pp. 439–561 (electronic), 2006.
[5] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *Proc. of the Fifth Symposium on Operating Systems Design and Implementation*. Boston, MA: USENIX Association, Dec. 2002, pp. 255–270.
[6] Freiling, Holz, and Wicherski, "Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks," in *ESORICS: European Symposium on Research in Computer Security*. LNCS, Springer-Verlag, 2005.
[7] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "A multifaceted approach to understanding the botnet phenomenon," in *Internet Measurement Conference*, J. M. Almeida, V. A. F. Almeida, and P. Barford, Eds. ACM, 2006, pp. 41–52.
[8] D. Dagon, C. Zou, and W. Lee, "Modeling botnet propagation using time zones," in *NDSS*. The Internet Society, 2006.
[9] M. Rajab, J. Zarfoxx, F. Monrose, and A. Terzia, "My botnet is bigger than yours (maybe, better than yours)," *Proceedings 2007 USENIX First workshop on Hot Topics in Understanding Botnets*, Apr. 2007.
[10] J. R. Binkley and S. Singh, "An algorithm for anomaly-based botnet detection," in *SRUTI'06: Proceedings of the 2nd conference on Steps to Reducing Unwanted Traffic on the Internet*. Berkeley, CA, USA: USENIX Association, 2006.
[11] G. Gu, V. Yegneswaran, and Martin, "Bothunter: Detecting malware infection through ids-driven dialog correlation," pp. 167–182. [Online]. Available: http://www.usenix.org/events/sec07/tech/gu.html
[12] J. Grizzard, V. Sharma, C. Nunnery, B. Kang B., and D. D., "Peer-to-peer botnets: Overview and case study," *Proceedings 2006 1st USENIX Workshop on Hot Topics in Understanding Botnets*, Apr. 2007.
[13] C.-J. Lu, O. Reingold, S. Vadhan, and A. Wigderson, "Extractors: optimal up to constant factors," in *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*. New York: ACM, 2003, pp. 602–611 (electronic).
[14] N. Cardwell, S. Savage, and T. Anderson, "Modeling tcp latency," *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, pp. 1742–1751 vol.3, 26-30 Mar 2000.
[15] E. Cooke, F. Jahanian, and D. Mcpherson, "The zombie roundup: Understanding, detecting, and disrupting botnetns," in *Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI)*, June 2005, pp. 39–44.
[16] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: an approach to universal topology generation," in *Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2001. Proceedings. Ninth International Symposium on*, Cincinnati, OH, USA, 2001, pp. 346–353.