HiNRG Technical Report: 01-10-2007
Measuring the Storm Worm Network

Sandeep Sarat   Andreas Terzis
sarat@cs.jhu.edu   terzis@cs.jhu.edu

Abstract

The Storm worm is a botnet which appeared in the early months of 2007. Its prolific growth, the use of decentralized command and control communication based on the Overnet P2P protocol and fast-flux servers for secondary-stage binary distribution, as well as the capability to aggressively defend itself, make Storm a notable species in the malware ecosystem. Despite considerable interest, Storm's defensive capabilities and its distributed nature have complicated the accurate estimation of its size and understanding of its network behavior.

In this paper, we actively probe the Storm botnet using Overnet queries to estimate its size – approximately 600,000 and 430,000 during the second and third week of October 2007, respectively. At the same time, we found several other surprising artifacts. Unlike traditional DHTs, the distribution of peer IDs is not uniform. Furthermore, we observed a small percentage of nodes which publish a large number of IDs, what we believe is an indication of index poisoning. Taken as a whole, these results provide insights which may facilitate researchers to curtail the Storm phenomenon as well as future P2P-based botnets.

1 Introduction

Botnets, networks of compromised machines under the control of so-called *botmasters*, represent a significant threat to the Internet today. The Storm worm (also known as Trojan.Peacomm), actually a botnet that made its first appearance in January 2007 and captured the headlines due to the massive amounts of unsolicited e-emails it generates, presents the leading edge of this phenomenon. It is unique in its use of a DHT protocol, based on Kademlia [11], to co-ordinate the infected hosts and the use of fast-flux DNS services to distribute binary updates [19]. Moreover, Storm aggressively defends itself by resisting reverse engineering attempts and executing DDoS attacks against external hosts that attempt to probe its operations.

Due to these aggressive defense mechanisms and its distributed nature, little is known about the network of Storm-infected end-hosts. For example, estimates about its size vary from approximately half a million of infected hosts on the low end, to fifty million hosts on the high end [6]. To the best of our knowledge, these reports are based on ad hoc measurement methodologies that have not been externally verified.

In this paper we present estimates about the size of the Storm infection, based on software we developed to crawl Storm's Kademlia-based network. Using this crawler, we estimate that approximately 600,000 and 430,000 end-hosts were members of the Storm botnet during the second and third week of October 2007, respectively. Perhaps more important than the size estimates, are the anomalies we discovered during this process. First, the distribution of keys stored in the DHT is not uniform over the hash space as in other P2P systems. We found that this non-uniformity is caused by keys inserted to the DHT that point to unreachable/non-routable IP addresses (*e.g.* private, multicast, loopback, and unallocated IP addresses). While only 1% of the Storm IPs belong to this category, they publish 45% of the keys found in the network. Moreover, we found a small percentage ($\sim 1\%$) of routable IPs that publish thousands of IDs, while the vast majority of Storm nodes publish only a single ID.

These anomalies cannot be attributed to hosts residing in private address space because Storm includes explicit mechanisms to discover and traverse NAT gateways. Furthermore, they cannot be generated by the churn associated with P2P systems since they occur over long temporal scales. Instead, we believe they represent attempts by entities external to the Storm network to interfere and track its operations. In that respect, they represent a practical application of *index poisoning* attacks previously theorized by a number of researchers [7, 9].

This paper has five sections. The section that follows outlines Storm's P2P architecture and describes our measurement methodology. We present the results from these measurements in Section 3 and cover related work in Section 4. Finally, we close in Section 5 with a summary and future research directions.

2 Background and Methodology

2.1 The Storm C&C Network Protocol

We start by summarizing the behavior of the Storm worm. Given the focus of our work, we outline Storm's Command & Control (C&C) protocol rather than presenting the functionality of its different binary components. Readers interested in these aspects are directed to [3, 13, 17].

Storm uses the Overnet protocol, which in turn is based on the Kademlia DHT [11]. Each peer, as well as each object stored in an Overnet network, is associated with a 128-bit identifier (ID). Peer identifiers are randomly generated using the MD4 cryptographic hash function [4]. Routing in Overnet is based on prefix matching, whereby the distance between two IDs is equal to the XOR of the two identifiers. For example, the distance between $a = 0001$ and $b = 1110$ is $d(a, b) = a \oplus b = 0001 \oplus 1110 = 1111$.

Overnet nodes organize their routing tables as lists of *k-buckets*. Specifically, for each $0 \leq i < 128$, the corresponding k-bucket holds up to $k(= 20)$ (IP address, UDP port, Node ID) triples for nodes whose distance from the current node is in $[2^i, 2^{i+1})$. This routing table resembles an unbalanced routing tree in which a node maintains only a few contacts to peers that are far away and increasingly more contacts to nodes within shorter distance. When an Overnet node receives any message (request or reply) from another node, it updates the appropriate k-bucket with the sender's node ID.

When an Overnet node receives a request for an ID, it returns the triples of the k nodes it knows about, that are closest to the requested ID. These triples can come from a single

k-bucket, or from multiple k-buckets if the closest k-bucket is not full. Routing then proceeds iteratively, by querying each successive peer on the route to the destination. When a new peer joins the network, it inserts itself into the contact list of other nodes by performing a lookup on its own ID. Moreover, peers periodically query their own ID in an effort to keep their own as well as their peers' routing information fresh.

While Overnet suggests that nodes have persistent IDs, we observed that Storm-infected nodes choose different IDs on reboot. The Storm binary has a hard-coded list of over 400 initial peers which it uses to attach itself into the network. Finally, given the large percentage of end-hosts residing in private address space, Overnet includes a special NAT discovery mechanism. Bots use this mechanism to detect whether they reside behind a NAT device and if so to advertise their globally visible IP address (rather than their private address) when they join the network.

In addition to performing periodic queries for their own IDs, Storm bots periodically search for a set of keys stored in the Overnet network. According to [17], bots generate these search keys through a built-in algorithm that uses the current date and a random number from [0...31]. The values associated with those keys contain an encrypted URL that the bots decrypt and download using HTTP. We noticed that nodes change their ID when key searches for a second stage binary fails. The bot subsequently rejoins the DHT with this new ID and restarts its search for the binary. Finally, we note that because Storm uses the same Overnet protocol that popular file-sharing networks use (*e.g.* eDonkey and eMule), non-infected hosts can be used to store keys for Storm. The botnet was probably designed this way, to leverage the existing P2P networks as a bootstrap mechanism during the early stages of its infection [12].

2.2 Measurement Methodology

We use the Overnet protocol described above to crawl the Storm network. Specifically, we developed a crawler that queries the network for randomly generated keys and records the node IDs, IPs, and port numbers that the peers return. We seed this search with a list of peer IPs collected by running an instance of the Storm Worm within a Qemu honeypot environment [2], running Windows XP. Through this process, we gathered a set of $\sim 4,000$ IPs over a period of five hours. Whenever the crawler receives a new usable (*i.e.* routable) IP address it sends a query for another random ID to the corresponding node. The query process continues until the crawler discovers no new nodes.

We perform two types of crawls: a *full crawl* and a *zone crawl*. The IDs that the crawler queries during a full crawl are selected from the entire 128-bit space. On the other hand, the IDs queried during a zone crawl share the same zone prefix. For example the 0x0A 8-bit prefix, contains all 128-bit IDs whose most significant eight bits have value 0x0A. While full crawls provide a more complete view of the network, they are resource intensive in terms of the network traffic they generate and the amount of storage required for the results. Moreover, they require hours to complete during which time the network's membership might change. On the other hand, 8-bit zone crawls typically finish in 10 minutes and generate significantly fewer queries (the reduction is proportional to the size of the zone queried).

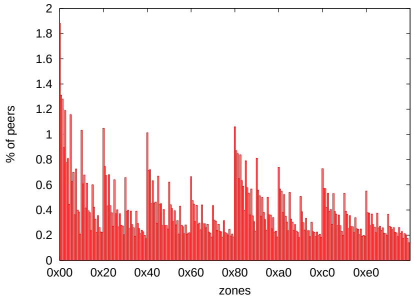


Figure 1: The distribution of Storm bot IDs over the 128-bit hash space.

3 Measurements

We present results derived from measurements we collected over a period of two weeks (10/10/2007 -10/24/2007), using the methodology described above.

3.1 Node ID distribution

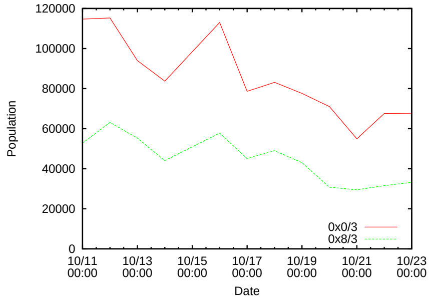
As previously explained, full crawls are slow, resource intensive, and potentially inaccurate. Therefore, we prefer to use zone crawls to estimate the number of peers within these zones and extrapolate the results to the full population. However, in order to do so we need to ensure that the measured zones are representative of the larger network. In general, given that node IDs are generated by a cryptographically secure hash function (*i.e.* MD4), node IDs should be uniformly distributed over the 128-bit address space. Furthermore, results from other Kademlia-based networks have experimentally verified the existence of this uniform distribution [15]. Nonetheless, we conducted a full crawl of the Storm network to verify this claim.

Figure 1 presents the results of this full crawl performed during the first day of our measurements (we observed similar patterns during other full crawls). It is evident that the distribution of node IDs is far from uniform, with recurring ramp-like structures that repeat at the beginning of each 3-bit zone. We defer the discussion about the underlying causes of this surprising non-uniformity until Section 3.3. For now, we use this result to select the length of the zones to crawl.

3.2 Population Estimates

Because the ID distribution has a regular pattern which recurs in every 3-bit zone, we infer the size of the overall population by crawling 3-bit zones. Specifically, we count the number of IPs whose corresponding IDs are within the zone and then multiply this count by the total number of zones.

We chose the two three-bit zones: 0x00/3 and 0x80/3. We selected these two zones because in Figure 1, the 0x00/3 zone accounts for the highest percentage of Storm peers, while the 0x80/3 represents the average case. Figure 2 presents the population estimates



(a)

Week	Zone	Min.	Max.	Avg.
Week I	0x08	44,061	63,127	53,010
Week I	0x00	78,618	115,271	99,882
Week II	0x08	29,480	49,012	36,515
Week II	0x00	54,924	83,143	70,470

(b)

Figure 2: Population estimates: (a) presents the population of the 0x0/3 and 0x8/3 zones during the weeks 10/10/2007–10/17/2007 (Week I) & 10/18/2007–10/24/2007 (Week II);(b) shows minimum, maximum and average population sizes for the same zones for the same period of two weeks.

we derived by crawling both zones at the same time of day over a period of two weeks. Then, using the average estimate from the larger 0x00/3 zone, the size of the whole Storm botnet is approximately 800,000 end-hosts during the first week. If we instead use the two zone averages to extrapolate the whole population, we reach an estimate of approximately 600,000 end-hosts. Using a similar methodology, the average storm population visible in the following week was recorded as 430,000. The values are in accordance with estimates from Microsoft ($\sim$ 500,000) [16].

We further categorized the end-hosts discovered during the zone crawls based on their country of origin. To do so, we use the maxmind database [10] to map IP addresses to countries. Figure 3 presents the 15 countries with the highest number of infected nodes. The continent with the highest percentage of infected hosts is North America, with the United States contributing approximately 30% of the peers. This population distribution deviates from the one observed for the popular KAD P2P file sharing network, which is also based on Kademlia [15]. Furthermore, Figure 3 also indicates that <1% of IP addresses we encountered could not be resolved by the database (listed as NA). This set consists of private, multicast, loopback, and unallocated/reserved IP addresses. The existence of such addresses is unexpected because Storm uses an IP query-response mechanism to traverse NAT boxes and it cannot communicate with unreachable/private IP addresses. Nonetheless, we do not include these addresses in the population estimates presented above.

3.3 Relationship between peer addresses and identifiers

The unexpected non-uniformity of the ID distribution combined with the existence of unreachable/private IP addresses, lead us to investigate the IDs associated with the group NA from Figure 3. To our surprise, we found that this group advertises 45% of the unique peers IDs we recorded, even though it accounts for <1% of the total population of nodes. Furthermore, as Figure 4.(a) illustrates, the IDs associated with these IP addresses are the

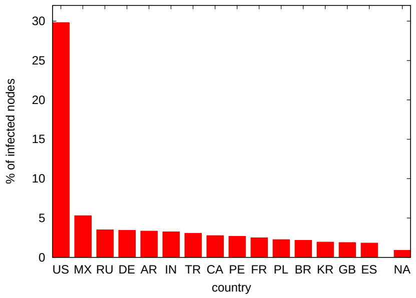


Figure 3: Top 15 countries in which peers are located percentage wise. The last bar NA (Not Available) comprises of non-publicly routable IP addresses.

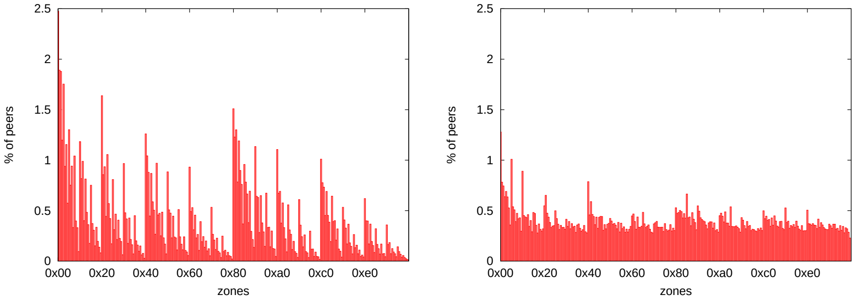


Figure 4: (a) Distribution of IDs attributed to invalid IP addresses ($\in NA$). (b) The distribution of IDs attributed to IP addresses which are not in the set NA. The x-axis represents the 128-bit hash space.

main contributors to the non-uniformity shown in Figure 1. As a matter of fact, the overall identifier distribution becomes considerably more uniform after removing these IDs (see Fig. 4.(b)).

The most plausible explanation for the existence of these IDs is that some peers are deliberately injecting 'bogus' identifiers in an attempt to poison the DHT. First studied by Liang *et al.* [9], index poisoning refers to the process of inserting a massive number of invalid records into a DHT's index, in an attempt to slow down lookups. As a matter of fact, other researchers have previously speculated that index poisoning could be an effective attack against the Storm worm [7].

Once we discovered that IP addresses in the NA group are associated with multiple IDs, we investigated whether the same was true for routable IP addresses. As Figure 5 illustrates, roughly 85% of the addresses are associated with a single ID, while a small percentage of addresses ($< 1\%$) are associated with a large number of IDs, some of them in the thousands. We note that while multiple infected hosts behind the same NAT device would advertise the same IP address, they would use different ports and therefore do not contribute to the phenomenon shown in Figure 5. Moreover, while we noticed particular periods during which our Storm specimen published different IDs every 10 minutes, that behavior cannot account for the very large number of IDs shown in Figure 5. The reason is that stale IDs are removed after 24 hours from the Overnet DHT, while consecutive crawls executed 24 hours apart consistently registered the same IP addresses publishing large number of IDs. We are thus left with the only explanation that some nodes deliberately inject multiple IDs in an effort to interfere or to monitor the Storm network.

To buttress our claim that nodes with large number of IDs are not legitimate Storm nodes, we use an out of band mechanism to verify if a given IP address truly hosts a Storm node. For this, we used data from a few popular DNS blacklists (*i.e.*, CBL [5], TQMCUBE [8] and UCEPROTECT [1]). The Storm bot is known to send out gigantic amounts of pump and dump stock unsolicited emails. As a result, one should expect that active Storm nodes will be listed by these spam traps. Thereby, we use the simple heuristic of classifying the IP addresses found in a zone crawl that are also listed in these black lists as infected hosts. We then plot the percentage of blacklist-identified hosts versus the number of IDs associated to a host. As Figure 6 shows, the percentage of blacklisted nodes decreases with the number of IDs the node publishes. This trend indicates that nodes which publish numerous IDs, do not seem to participate in the malicious activities of the botnet and therefore are not likely to be infected hosts.

3.4 Discussion

In the previous section, we showed a strong reciprocal correlation between the appearance of a node in spam blacklists and the number of IDs published in the Storm network. We believe this is due to the presence of "outsider" nodes aiming to poison the network. This belief is further bolstered by the fact that a few of these nodes explicitly published their IDs even to our crawler. This is in spite the fact that our crawler does not insert itself in the P2P network, but merely queries it. Legitimate Storm nodes we tested against, did not contact our crawler with such messages.

These results raise a number of interesting issues. First of all, botmasters will eventually notice the existence of nodes interfering with their networks. They can then abuse the blacklist query mechanism to launch attacks against the peers that do not appear to partici-

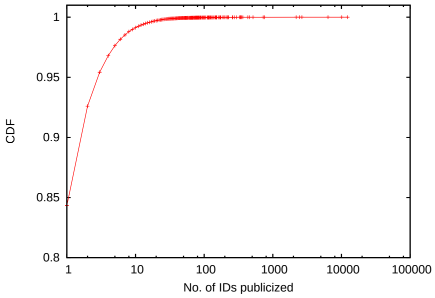


Figure 5: Cumulative density function of the number of IDs associated with a single IP address and port. Unreachable/non-routable IP addresses were not included in this distribution.

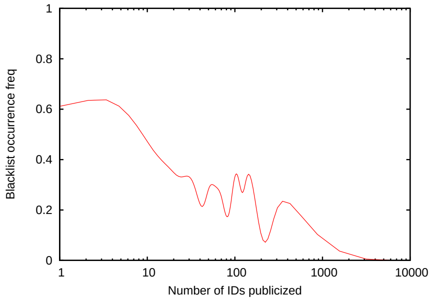


Figure 6: Correspondence between the number of IDs published and listing in spam block lists.

pate in the botnet's malicious activities. Another important point is that of index poisoning and pollution. We did notice a few nodes replying with polluted values while searching for keys, pertaining to the second stage storm binary. This is an effective strategy to deter file sharing networks, as users have to manually sift through bad search results. However, its effectiveness in stopping a bot, rather than just slowing it down, is arguable. A study of the effectiveness of such poisoning techniques in curtailing botnets is an avenue for future work.

4 Related Work

The majority of botnets today use the Internet Relay Protocol (IRC) to disseminate commands to the individual bots. IRC's centralized architecture enables snooping of the C&C channels, thereby potentially revealing botnet membership and commands passed onto the bot armies [14]. Due to these shortcomings, researchers have speculated that botmasters would innovate by migrating to C&C protocols that are harder to detect and infiltrate [20].

The Storm worm is a prime example of this evolution in the botnet ecosystem, due to its DHT-based C&C protocol and aggressive defensive capabilities. Consequently, it has been the subject of multiple research reports, most of them focusing on the binary analysis of the Peacomm binary. Grizzard *et al.*present a case study of Peacomm, by running a single specimen of the infection in a contained honeypot environment [7]. Detailed descriptions of the multiple techniques that Storm uses to disguise itself are provided in [3, 13, 17]. Instead, our work presents an analysis of Storm's DHT protocol and is the first to discover traces of what seems to be widespread attempts to poison Storm's C&C network.

A number of recent measurement studies have focused on the widely deployed KAD P2P network – another variant of the Kademlia DHT that uses a slightly different routing table. Stutzbach *et al.* use a distributed crawler to study the ID lookup performance in KAD [18]. Steiner *et al.* crawl the KAD network to estimate the lifetime of peer sessions in this network [15]. The goal of our work is not to measure the lifetime of the infected hosts, but to measure the actual distribution of hosts and to provide insights into the Storm network. Furthermore, we have discovered important differences between the Storm and the KAD network, such as the non-uniform distribution of peer IDs.

5 Summary and Future Work

We present results from a measurement study of the Storm botnet which uses a decentralized P2P infrastructure to coordinate individual bots. Our findings reveal that although the underlying DHT protocol is similar to the one used in popular P2P file-sharing systems (*e.g.* KAD), the properties of the Storm network, as a whole are considerably different. Specifically, we witnessed widespread attempts to poison Storm's Overnet network by injecting invalid IDs that point to unreachable IP addresses. Moreover, we found that a small number of routable IP addresses inject a large number of IDs, most likely in an attempt to monitor or interfere with the Storm network.

As part of our future work we will study longitudinal trends of the Storm network through long-term crawls. Moreover, we plan to combine the results obtained from these crawls with the binary analysis of bot binaries, to counter the rapidly evolving threat of botnets.

Acknowledgements

We would like to thank Răzvan Musăloiu-E., for his help in deploying the Storm botnet crawler.

References

[1] Admins WebSecurity GbR. Germany's first Spam Protection Database. Available at: http://www.uceprotect.net/en/index.php/.

[2] F. Bellard. Qemu, a fast and portable dynamic translator. In *Proceedings of the USENIX Annual Technical Conference, FREENIX Track.*, 2005.

[3] F. Boldewin. Peacomm.C - Cracking the Nutshell. Available at: http://www.reconstructer.org/papers/Peacomm.C-Crackingthenutshell.zip.

[4] R. Brunner. A performance evaluation of the kad-protocol. Masters Thesis. Corporate Communications Department. Institut Eurocom, France, Nov. 2006.

[5] Composite Blocking List. Available at: http://cbl.abuseat.org/.

[6] S. Gaudin. Storm Worm Botnet More Powerful Than Top Supercomputers. Available at: http://www.informationweek.com/news/201804528, Sept. 2007.

[7] J. Grizzard, V. Sharma, C. Nunnery, B. Kang, and D. Dagon. Peer-to-Peer Botnets: Overview and Case Study. In *Proceedings of the first USENIX workshop on Hot Topics in Botnets (HotBots'07)*, Apr. 2007.

[8] D. C. Hart. Real Time DNSBL and Spam Trap. Available at: http://tqmcube.com/.

[9] J. Liang, N. Naoumov, and K. W. Ross. The Index Poisoning Attack in P2P File Sharing Systems. In *Procedings of the 25th IEEE International Conference on Computer Communications, INFOCOM*, 2006.

[10] MaxMind LLC. MaxMind GeoIP Country Database. Available at `http://www.maxmind.com/`, 2007.

[11] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the XOR metric. In *Proceedings of the Sixth International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.

[12] C. Nunnery and B. B. Kang. Locating Zombie Nodes and Botmasters in Decentralized Peer-to-Peer Botnets. Available at: `honeynet.uncc.edu/papers/P2PDetect_ConceptPaper.pdf`, 2007.

[13] P. Porras, H. Sadi, and V. Yegneswaran. A Multi-perspective Analysis of the Storm (Peacomm) Worm. Available at: `http://www.cyber-ta.org/pubs/StormWorm/report/`, Oct. 2007.

[14] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis. A Multifaceted Approach to Understanding the Botnet Phenomenon. In *Proceedings of ACM SIGCOMM/USENIX Internet Measurement Conference (IMC)*, pages 41–52, Oct., 2006.

[15] M. Steiner, T. En-Najjary, and E. W. Biersack. A global view of KAD. In *Proceedings of the Internet Measurement Conference, IMC*, 2007.

[16] B. Sterling. Microsoft Battles the Storm Worm. Available at: `http://blog.wired.com/sterling/2007/09/microsoft-battl.html`, Sept. 2007.

[17] J. Stewart. Storm Worm DDoS Attack. Available at: `http://www.secureworks.com/research/threats/storm-worm`, Feb. 2007.

[18] D. Stutzbach and R. Rejaie. Understanding churn in peer-to-peer networks. In *Proceedings of the 6^{th} Internet Measurement Conference (IMC)*, pages 189–202, New York, NY, USA, 2006. ACM Press.

[19] The Honeynet Project & Research Alliance. Know Your Enemy:Fast-Flux Service Networks, An Ever Changing Enemy. Available at: `http://www.honeynet.org/papers/ff/index.html`, July 2007.

[20] P. Wang, S. Sparks, and C. C. Zou. An Advanced Hybrid Peer-to-Peer Botnet. In *Proceedings of the First Workshop on Hot Topics in Understanding Botnets (HotBots'07)*, Apr. 2007.