Brief Announcement: Trinity

A Distributed Defense Against Transient Spam-bots

Alex Brodsky
University of Winnipeg
Winnipeg, MB, Canada, R3B 2E9
abrodsky@acs.uwinnipeg.ca

Dmitry Brodsky
Worio
Vancouver, BC, Canada, V6B 2T5
dbrodsky@worio.com

ABSTRACT

Transient spam-bots are hijacked computers that are connected to the Internet for short periods of time, during which they send large amounts of spam. These spam-bots have become a principle source of spam; against which, static countermeasures such as DNS Black Lists are largely ineffective, and content-based filters provide only temporary relief without ongoing tuning and upgrading—a never-ending cat-and-mouse game.

This is a brief overview of *Trinity* [1], a distributed, content independent, spam classification system that is specifically aimed at transient spam-bots. Trinity uses source identification in combination with a peer-to-peer based distributed database to identify and track transient spam-bots. Trinity's design load balances the task of tracking the transient spam-bots and provides a robust defense against denial-of-service and malevolent peer attacks.

Categories and Subject Descriptors: C.2.4 [Computer-Communication Networks]: Distributed Systems

General Terms: algorithms, design, reliability, security.

Keywords: e-mail classification, peer-to-peer.

1. SPAM FROM TRANSIENT BOTS

In the past spam came from static sources such as marketing companies, various e-commerce firms, and third-party mailers. In these cases, the spam was sent from a single source with a fixed IP address. E-mail originating at these sources could easily be classified by checking its source against a blacklist of known spammers [5, 12, 13]. If the blacklists are correct and up-to-date, classifying spam that originates at a single source is straightforward.

To circumvent blacklists, spammers use botnets. Botnets, are networks of bots; computers that are hijacked via a virus, worm, or Trojan. In many cases these bots are transient: they are on for short periods of time since most users only turn on their machines when they use them. Furthermore, when turned on, the machines are assigned a dynamic IP that can change between boots on a daily or even hourly basis due to DHCP churn [10]. Consequently, the bot is visible for short periods of time and appears as a different bot each time.

The bot uses its user's Internet Service Provider's (ISP's) SMTP server to relay the spam. Since the server cannot differentiate between e-mail sent by the actual user from that sent by the bot, it becomes an unwilling accomplice in the distribution of spam. Furthermore, since the IP address of the bot can change from day to day, blacklists are rendered ineffective. Adding the ISP's relay to the blacklist cannot be done since it would also block all legitimate e-mails from the entire ISP.

A different approach to identifying spam is to classify e-mail based on its content i.e., the body of the e-mail. This approach does not rely on the source of the e-mail being static and does not suffer from the lag between when a new spammer commences operations, and when it is added to a blacklist. Unfortunately, for every new filter, spammers develop new countermeasures, forcing the filter developers to create new filters. This wastes significant resources, and increases the number of false-positives.

Existing collaborative and distributed e-mail classification systems [2, 4, 7, 9, 15] keep distributed databases of known spam signatures that are matched against incoming e-mail and assume that the same spam message is sent to many recipients. Unfortunately, e-mail messages can be tailored for each recipient, invalidating the signatures—any content based approach has similar weaknesses.

2. OVERVIEW OF TRINITY

Trinity is based on the assumption that transient spam bots send large amounts of e-mail in a short period of time. That is, any e-mail coming from hosts that have recently sent many e-mails is potentially spam, and if the source has a dynamically allocated IP address (or simply a *dynamic IP address*)[1] and the sender is not in the recipient's address book or list of past recipients or senders, then it is almost certain that the e-mail is spam. Consequently, if we can identify the email sources and gauge the rate at which these hosts are sending, we can determine if an e-mail is spam.

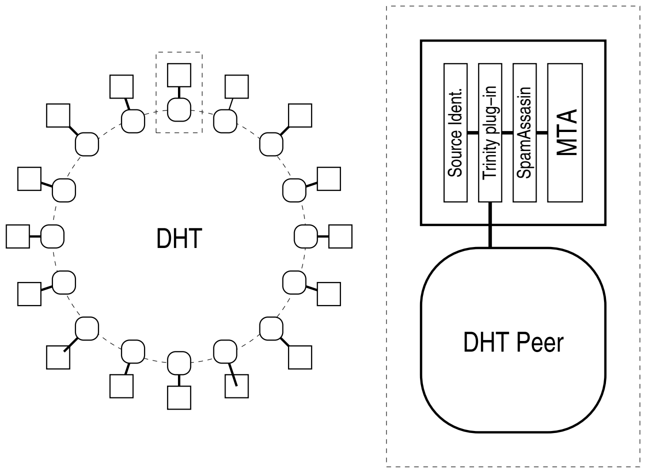
Since a spam-bot typically sends less than 10 e-mails per day to any single domain [6, 10], determining whether a bot is sending spam is inherently a collaborative effort. Trinity, is a peer-to-peer spam detection system that depends on the collaboration of peers: as the number of peers increases, spam-bot identification improves because the sample size increases. Each peer is associated with a mail relay. A peer identifies the source of the e-mails that it receives and stores this information in a distributed database that is used and updated by all peers. This information is then used to gauge the recent e-mail sending rates of the sources.

Operation. E-mail classification consists of several distinct parts: identifying the source of e-mails, keeping track of how many e-mails were recently sent by a source, and disseminating this information for the purposes of classifying future e-mails. Additionally, these tasks must be coordinated for each e-mail as it is received by the mail servers, which are traditionally called mail transfer agents (MTAs). Thus, Trinity comprises four respective components.

When an e-mail arrives at a MTA, it is passed to a general spam

[1]A host with a statically allocated IP address is said to have a *static IP address*.

Copyright is held by the author/owner(s).
PODC'07, August 12–15, 2007, Portland, Oregon, USA.
ACM 978-1-59593-616-5/07/0008.



classification framework such as SpamAssassin [11]. The message is passed to a variety of plug-ins (within SpamAssassin), including a Trinity plug-in, which is responsible for coordinating the source identification and source tracking tasks, and embedding the classification in the e-mail.

The plug-in first passes the message envelope to the source identification server, a locally running process that determines the e-mail's source—a challenging task—and whether the source of the e-mail has a static or dynamic IP address. If the source has a static IP address, then the source is not, by definition a transient spambot, and existing blacklist mechanisms are used, i.e., if its source is listed in one of the blacklists, it can immediately be classified as spam. In this case, no further processing is necessary. Otherwise, the source is tracked via its dynamic IP address.

The plug-in uses this address to update the distributed database that tracks e-mail sources. A local server, which is a peer in the distributed database, propagates the update to the distributed database, stores chunks of the distributed database, and caches updates and queries for the local MTA.

Next, the plug-in queries the distributed database for the *sender score*, which is the number of e-mails that were recently sent, typically within an hour, from the same source as the current e-mail. If the score is high, i.e., many e-mails were sent, then the score is appended to the e-mail's envelope and passed back to the spam framework. Otherwise, If the score is low, this indicates that either the source has not sent many e-mails recently, or that the e-mail may be the first of many e-mails that were sent.

To distinguish these two cases, the message is quarantined for a short period of time, and then the database is queried again. The score from the second query is then appended to the e-mail's envelope and passed back to the framework. The quarantine's impact is minimal because it is only used for e-mails from senders with dynamic IP addresses, and its length is on the order of minutes.

At this point the back-end processing by Trinity is complete. The e-mail, with its appended envelope, is stored to be consumed by mail clients that typically have highly configurable rule-based filters that are used to complete the spam classification. Specifically, a simple rule is used to determine if an e-mail with a high score is from a sender that is neither in the address book, or in the sender or recipient lists. If so, the e-mail is deemed to be spam.

Key Challenges. The key technical challenges of Trinity is to correctly identify the source of an e-mail, to quickly and efficiently update the distributed database, and to ensure that the database is secure. Determining the true source of an e-mail is difficult because the sender or any malicious mail relay can falsify part of the e-mail envelope [3]. Using the information in the e-mail envelope, Trinity identifies the first trusted relay that accepted the e-mail, the host from which the e-mail is accepted is considered the source—determining the first trusted relay is also challenging.

Even though the size of the database is small, a couple gigabytes, each peer cannot maintain its own copy. The database is useless unless it is promptly updated by other Trinity peers whenever they receive new e-mails. Since the number of updates is a sizeable fraction of the 70 billion e-mails sent each day [12], a server that hosts the entire database, and its connection, would be swamped. Trinity uses a distributed hash table (DHT), such as Chord [14], as its database. A DHT suffices because the updates are commutative, may occasionally be lost, and the update latency is on the order of seconds. To minimize traffic queries and updates are sent via UDP since they contain little data.

Denial of service attacks are mitigated by the distributed nature of the DHT and Trinity uses replication and reputation to counteract malevolent peers. Database fragments are replicated on groups of sequential peers, ensuring that no single peer can corrupt its fragment. Furthermore, queries and updates are concurrently sent to several group members, ensuring that inconsistent responses are immediately detected. If a peer's response is inconsistent, its reputation, which is stored in the same database, is reduced—a peer is excommunicated if its reputation drops below a threshold. Once a peer is excommunicated, it is placed on a blacklist and may be readmitted either at the discretion of the system administrator or after some minimum amount of time.

Current and Future Work. We are currently in the process of developing, deploying, and testing a prototype of Trinity. We plan to test our system using PlanetLab [8] in particular focusing on the performance of the distributed database. We also need to determine the threshold for classifying an e-mail source as a spammer. The threshold depends on the number of participating peers and thus may need to be dynamically adjusted as the number of peers increases. We are also investigating mechanisms for securing the peer-to-peer database.

3. REFERENCES

[1] Brodsky, A., and Brodsky, D. A distributed content independent method for spam detection. In *1st USENIX Workshop on Hot Topics in Understanding Botnet* (2007).

[2] Damiani, E., de Vimercati, S. D. C., and Samarati, P. P2P-based collaborative spam detection and filtering. In *4th IEEE Conference on P2P* (2004).

[3] Goodman, J. IP addresses in email clients. In *1st Conference on Email and Anti-Spam* (2004).

[4] Gray, A., and Haahr, M. Personalised, collaborative spam filtering. In *4th Conference on Email and Anti-Spam* (2007).

[5] Iverson, A. Dnsbl resource. http://www.dnsbl.com/, 2007.

[6] Jung, J., and Sit, E. An empirical study of spam traffic and the use of DNS black lists. In *4th ACM SIGCOMM Conference on Internet Measurement* (2004).

[7] Kong, J., Boykiny, P., Rezaei, B., Sarshar, N., and Roychowdhury, V. Scalable and reliable collaborative spam filters: Harnessing the global social email networks. In *3rd Annual Workshop on the Weblogging Ecosystem: Aggregation, Analysis and Dynamics* (2006).

[8] PlanetLab. http://www.planet-lab.org/, 2007.

[9] Prakash, V. Vipul's Razor. http://razor.sf.net, 2007.

[10] Ramachandran, A., Dagon, D., and Feamster, N. Can DNS-based blacklists keep up with bots? In *3rd Conference on Email and Anti-Spam* (2006).

[11] SpamAssassin. http://spamassassin.apache.org/, 2007.

[12] SpamCop. http://www.spamcop.net/, 2007.

[13] Spamhaus. http://www.spamhaus.org/, 2007.

[14] Stoica, I., Morris, R., Karger, D., Kaashoek, F., and Balakrishnan, H. Chord: A scalable peer-to-peer lookup service for internet applications. In *ACM SIGCOMM Conference* (2001).

[15] Vixie, P., and Rhyolite LLC. Distributed Checksum Clearinghouse. http://www.rhyolite.com/anti-spam/dcc/, 2007.