Revealing Botnet Membership Using DNSBL Counter-Intelligence

Ramachandran, Feamster, and Dagon
Georgia Tech

Discovering botnets

- If network operators and system administrators could determine whether a host is a member of a botnet, they could take appropriate steps to mitigate attacks perpetrated by the botnet
- State-of-the-art of botnet identification based on
  - User complaints
  - Localized honeypots and intrusion detection systems
  - Correlation of data from darknets

DNS-based Blackhole Lists (DNSBL)

- Used to track IP addresses that originate spam
- Emails sent from these addresses are rejected

Motivation

- Botmasters sell "clean" bots (those not listed in a DNSBL) at a premium
- Botmasters perform lookups on DNSBLs to determine their bots blacklist status

Key idea of the paper

- If one can distinguish DNSBL reconnaissance queries for a botnet from legitimate DNSBL queries (e.g., from a mail server), then one can identify likely bots

Properties of reconnaissance queries

- Based on a DNSBL query graph
  - Nodes are IP addresses
  - Edge from A to B indicates that node A has queried a DNSBL about B
- Use detection heuristics based on expected spatial and temporal characteristics of legitimate and reconnaissance-based lookups
Property 1 (spatial relationships)

- Legitimate mail servers will perform queries and be the object of queries
- Hosts performing reconnaissance-based lookups will only perform queries; they will not be queried by other hosts
- A host that is not itself being looked up by any other mail server is most likely not a mail server
- Hosts that have a high out-degree in the DNSBL query graph and a low in-degree are likely unrelated to the delivery of legitimate mail

Property 2 (temporal relationships)

- A legitimate mail server’s DNSBL lookups reflect actual arrival rates of real email messages
- Reconnaissance-based lookups will not mirror the arrival patterns of legitimate email
- Email traffic tends to be diurnal
- The authors leave this as future work

 Lookup ratio, $\lambda$

- Let $d_{n,\text{out}}$ be the number of distinct addresses that node $n$ queries
- Let $d_{n,\text{in}}$ be the number of distinct addresses that query for node $n$
- The lookup ratio $\lambda_n$ as $\lambda_n = \frac{d_{n,\text{out}}}{d_{n,\text{in}}}$
- Most effective when reconnaissance hosts are disjoint from spamming hosts

Reconnaissance techniques

- Third-party
- Self-reconnaissance
- Reconnaissance using other bots

Third-party reconnaissance

- Botmaster performs DNSBL lookups from a single host (may be the command-and-control host)
- Host performing the lookups will have a high value of $\lambda_n$
- Once identified, the DNSBL operator could monitor the hosts being looked up and identify them as likely bots
- Operator could also return incorrect information about the bot’s status in the blacklist

Self-reconnaissance

- Each bot performs reconnaissance on itself
- Could be used by botmaster to distribute the workload
- Easily identified since legitimate mail server would likely never lookup its own status
Distributed reconnaissance

- Each bot performs reconnaissance on behalf of other bots either in the same botnet or in other botnets
- Property 1 is unlikely to hold, since bots doing the lookup will likely be the target of lookups by mail servers to which they send spam
- If have a small number of known bots, then can assume that lookups they do are for hosts that are also bots

Data collected

- Two datasets collected between 17 Nov 05 and 31 Dec 05
  - DNSBL query log
  - Log of bot connections to a sinkhole for a Bobax botnet
- Corpus of spammer addresses collected at a spam honeypot
  - Used to verify that the IPs in the constructed DNSBL query graph were spammer bots

Constructing the query graph

- Start with DNSBL query log and the set B of IP addresses that attempted to connect to the Bobax sinkhole
- Pruning: consider only those queries for which either the querier or the queried are in set B
- For each query in the pruned set add an edge from querier to the queried to the graph, G
  - If edge is new assign weight of one
  - If edge already exists increase its weight by one

Query graph extrapolation

- To address the problem of overlooking queries where neither the querier nor queried are in B
- Consider the remaining queries in the query log, and add an edge to G if either the querier or queried is in G
- Repeat above step until no new edges are added to G

Detection using the query graph

- Construct $\lambda_n$ for each node in the graph
- Hypothesis:
  - Nodes with highest values of $\lambda_n$ are third-party lookup nodes
  - Queried nodes are known bots

Preliminary results

- Nodes with highest values of $\lambda_n$ are known bots
- Queried nodes in the graph were new, previously unknown bots
- Using data in the spam sinkhole revealed that some of the queried nodes were Windows machines and confirmed spam originators
- Suggests that could start with a small set of known bots and DNSBL query graph to “bootstrap” the discovery of new bots
Starting with an initial set of suspect bots from a spam trap could use the DNSBL query graph to identify new bots

If a bot is also being queried by another high out-degree node, this could indicate a separate botnet

Could develop an algorithm to walk the query graph from parent to parent to identify separate botnets

Reconnaissance poisoning: tune the behavior of the blackhole list server to mislead a botmaster

- Trick the botmaster into thinking a blacklisted bot is “clean”, resulting in the spam sent being dropped
- Indicate that a “clean” host is blacklisted, causing the botmaster to not use that host

If not properly tuned, could give bad information to legitimate mail servers (false positives)
Contributions

- Developed heuristics to distinguish DNSBL reconnaissance queries from legitimate DNSBL queries
- Studied DNSBL reconnaissance techniques and discovered that
  - Bots are performing reconnaissance on behalf of other (possibly newly infected) bots
  - Although some bots perform a large number of queries, much of the reconnaissance activity is spread across many bots each of which issues only a few queries
- Identified new bots using heuristic above
- Outlined possible real-time countermeasures

Questions?