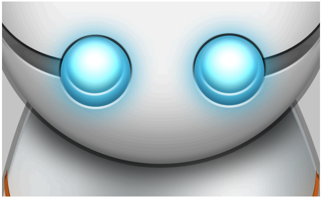
A brief Incursion into Botnet Detection

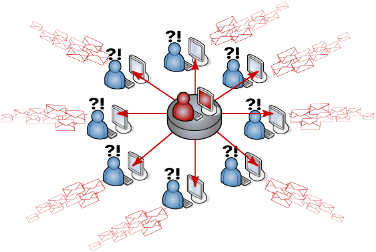
Anant Narayanan



Advanced Topics in Computer and Network Security
October 5, 2009

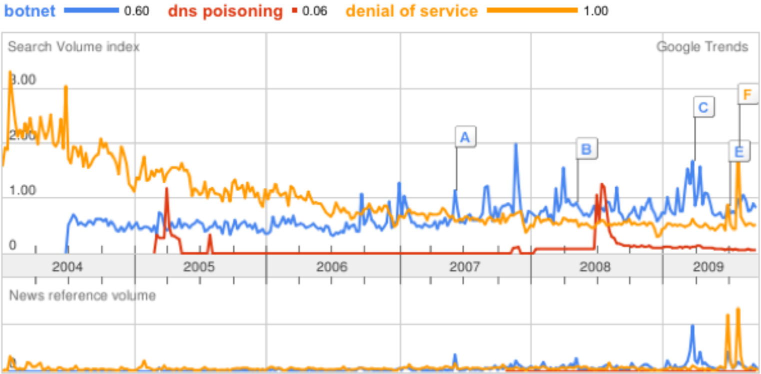
What We're Going To Cover

What Are Botnets?



- Networks of "zombie" computers
- The perpetrator compromises a series of systems using various tools on existing security holes
- Then, he simply controls these bots to do his bidding

Why Are They Bad?



How Do They Work?

PULL

- HTTP(S) is the most commonly used protocol
- A simple GET request at regular interval to receive commands

PUSH

- IRC(S) is the most commonly used protocol
- All bots join a chat room and wait for commands

How Can We Stop Them?

- Prevent computer from being infected in the first place? Impractical, given the thousands of vulnerable machines that will probably never be patched
- Actively prevent commands from reaching bots, or prevent bots from acting on those commands (use the network)
- Passively detect a botnet's presence and take offline action

Detecting C&C Traffic

Botnet C&C Traffic is difficult to detect because:

- Uses normal protocols in ordinary ways
- Traffic volume is low
- Number of bots in a monitored network may be small
- Traffic may use encrypted channels

Spatial-Temporal Correlation!

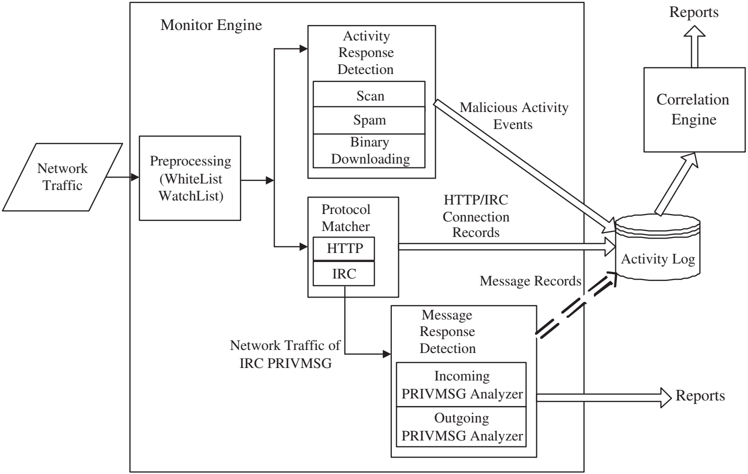
Pre-programmed response activities

- Command is sent to all bots around the same time (especially true for PUSH models)
- Bots process and usually perform some network operation in response
- Ordinary network traffic is unlikely to demonstrate such synchronized or correlated behavior

Response Types

- Message response: Execution result, status or progress
- Activity response: Actual (malicious) network activity

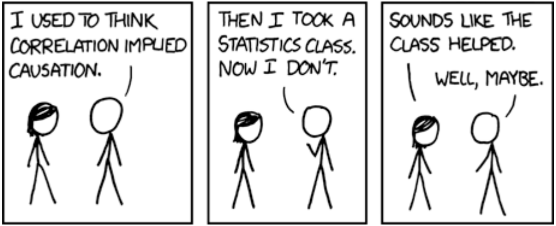
BotSniffer: Architecture



Monitor Engine

- Preprocessing:
  - Unlikely protocols
  - White lists
- Protocol Matcher
  - Currently focuses on IRC/HTTP
- Message Response Detection
  - IRC PRIVMSG responses
- Activity Response Detection
  - Abnormally high scan rates
  - Weighted failed connection rates
  - SMTP connections

Correlation Engine



- First, the BotSniffer groups clients according to their destination IPs and ports
- Then, it perform correlation analysis on these groups

Group Activity Response

Response-Crowd-Density-Check

$H_0 \rightarrow$ "*Not Botnet*", $H_1 \rightarrow$ "*Botnet*", $Y_i \rightarrow i^{th}$ *group member*

$$\wedge_n = ln\frac{P_r(Y_1,\ldots,Y_n|H_1)}{P_r(Y_1,\ldots,Y_n|H_0)} = \sum_i ln\frac{Y_i|H_1}{P_r|H_0}$$

User chooses α (false positive rate) and β (false negative rate)

Threshold Random Walk

When $Y_i = 1$, increment by $ln\frac{\theta_1}{\theta_0}$

When $Y_i = 0$, decrement by $ln\frac{1-\theta_1}{1-\theta_0}$

If the walk reaches $ln\frac{1-\beta}{\alpha}$ it is a botnet

If it reaches $ln\frac{\beta}{1-\alpha}$ it is not

Otherwise, we watch the next round

Group Message Response

Instead of looking at *density*, let's look at *homogeneity*

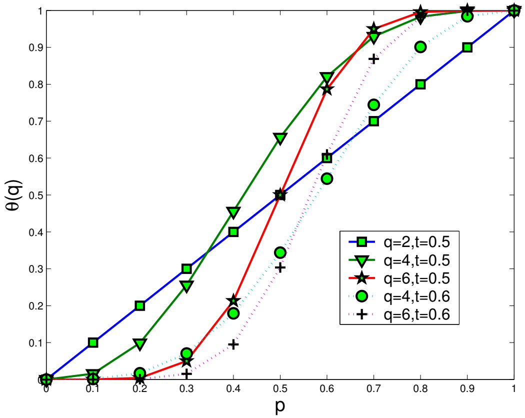
Response-Crowd-Homogeneity-Check

Let Y_i denote if the i^{th} crowd is *homogenous* or not
Homogeneity is decided by the *Dice* factor

$$Dice(X, Y) = \frac{2|ngrams(X) \cap ngrams(Y)|}{|ngrams(X)| + |ngrams(Y)|}$$

Now, for q clients in the crowd, compare all unique pairs and calculate their *Dice* distances. If (for eg.) $> 50\%$ are within a threshold t, the crowd is marked as *homogenous*

Selecting q and t



Single client detection

IRC

We can make use of the fact that IRC is a broadcast protocol and apply the homogeneity check on incoming messages to a single client

HTTP

Bots have strong periodical visiting patterns (to connect and retrieve commands)

Did it Work?

Trace	trace size	duration	Pkt	TCP flows	(IRC/Web) servers	FP
IRC-1	54MB	171h	189,421	10,530	2,957	0
IRC-2	14MB	433h	33,320	4,061	335	0
IRC-3	516MB	1,626h	2,073,587	4,577	563	6
IRC-4	620MB	673h	4,071,707	24,837	228	3
IRC-5	3MB	30h	19,190	24	17	0
IRC-6	155MB	168h	1,033,318	6,981	85	1
IRC-7	60MB	429h	393,185	717	209	0
IRC-8	707MB	1,010h	2,818,315	28,366	2,454	1
All-1	4.2GB	10m	4,706,803	14,475	1,625	0
All-2	6.2GB	10m	6,769,915	28,359	1,576	0
All-3	7.6GB	1h	16,523,826	331,706	1,717	0
All-4	15GB	1.4h	21,312,841	110,852	2,140	0
All-5	24.5GB	5h	43,625,604	406,112	2,601	0

Did it Work?

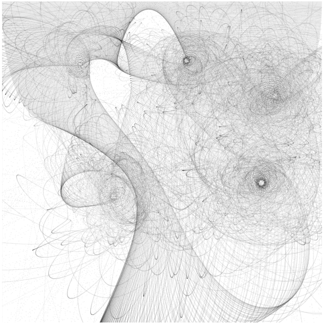
BotTrace	trace size	duration	Pkt	TCP flow	Detected
B-IRC-G	950k	8h	4,447	189	Yes
B-IRC-J-1	-	-	143,431	-	Yes
B-IRC-J-2	-	-	262,878	-	Yes
V-Rbot	26MB	1,267s	347,153	103,425	Yes
V-Spybot	15MB	1,931s	180,822	147,921	Yes
V-Sdbot	66KB	533s	474	14	Yes
B-HTTP-I	6MB	3.6h	65,695	237	Yes
B-HTTP-II	37MB	19h	395,990	790	Yes

Passive Detection

DNSBL

- DNS Blackhole Lists contain IP addresses that are sources of spam. Botmasters sell bots *not* on any DNSBL at a premium price
- Thus, Botmasters themselves perform lookups on DNSBLs to determine the status of their bots. Can we use this?

Heuristics



Spatial

A legitimate mail server will perform queries and be the object of queries. Bots will only perform queries, they will be not be queried for by other hosts

Heuristics

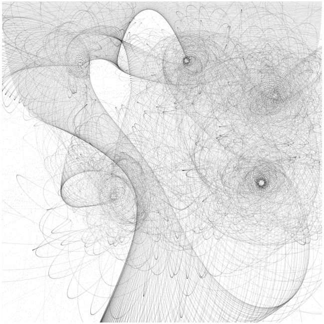
Botnet Detection

Introduction

BotSniffer
Control Channels
Architecture
Algorithms
Results

DNSBL Method
Counter-intelligence
Reconnaissance

Conclusion



Temporal

Legitimate lookups are typically driven automatically when emails arrive at the mail server and will this arrive at a rate that mirrors arrival rates of emails

Types

- Self Lookup: Each bot looks up it's own DNSBL record. Usually a dead giveaway, thus not used
- Third-party Lookup: All bots are looked up by a single dedicated machine. If that machine isn't a mail server, we can simply use Spatial heuristics and detect botnet membership
- Distributed Lookups: Each bot looks up a set of records for other bots in the network. Complicated to implement and spatial heuristics will fail. Temporal heuristics, however, may help in detection

Thanks for Listening

Detecting botnets is hard work, but certainly possible!

Questions?