Article presentation for: The Dark Cloud: Understanding and Defending against Botnets and Stealthy Malware

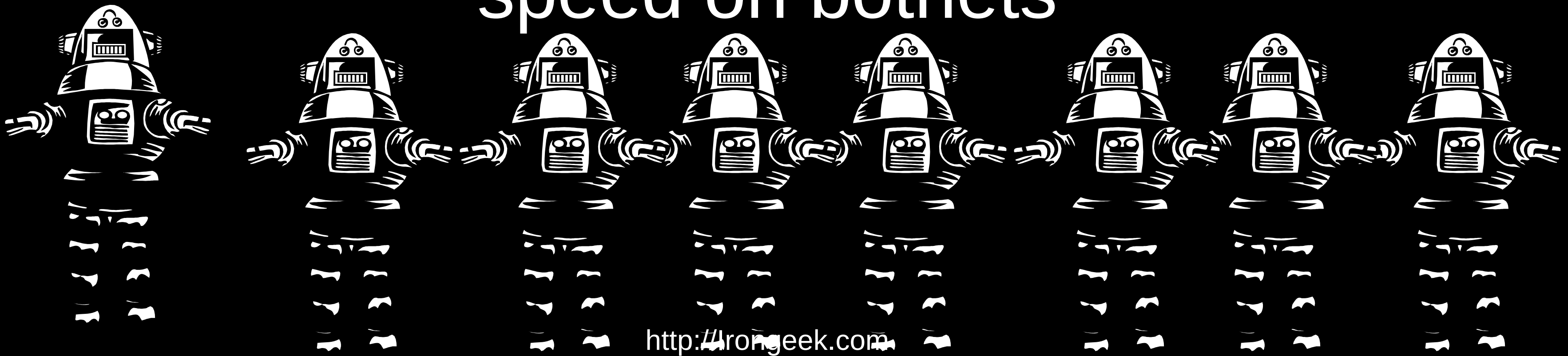
Based on article by:
Jaideep Chandrashekar, Steve Orrin, Carl Livadas, Eve M. Schooler

Available at:
http://download.intel.com/technology/itj/2009/v13i2/pdfs/ITJ9.2.9-Cloud.pdf

This presentation by:
Adrian Crenshaw

Background

A little information to get you up to speed on botnets



So, what is a Botnet?

- A collection of compromised computers that can be sent orders
- Individual hosts in a Botnet are know as bots or zombies
- The administrator of the Botnet is often known as a “Bot Herder”
- A few examples of Botnets include:
  Storm
  Kraken
  Conficker

Botnet life cycle

(As outlined by the article)

- Spread Phase
  - SE Spam, Web drive bys, Network worm functionality, etc.
- Infection Phase
  - Polymorphism
  - Rootkitting
    - Trojan binaries
    - Library hooking
- Command and Control Phase
- Attack Phase

How do hosts become part of a Botnet?

- Drive by malware installs via web browsers
- Automated or targeted network vulnerability attacks
- End users socially engineered to install them via phishing attacks, or confusing browser messages
- Other vectors…

Botnet Source Code Families

- Lots of source code is out there:
  - Agobot
  - Rxbot
  - SDBot
  - Spybot
  - Others…

  http://leetupload.com

  Search for BotNet.Source.Codes.rar

How are Botnets controlled?

- Decentralized Command and Control Channels (C&C)
- Decentralization is important to make C&C harder to shutdown
- By using Command and Control Channels, “bot herders” can change what their Botnet is tasked to do, and update the Botnet’s nodes

Illustration of C&C

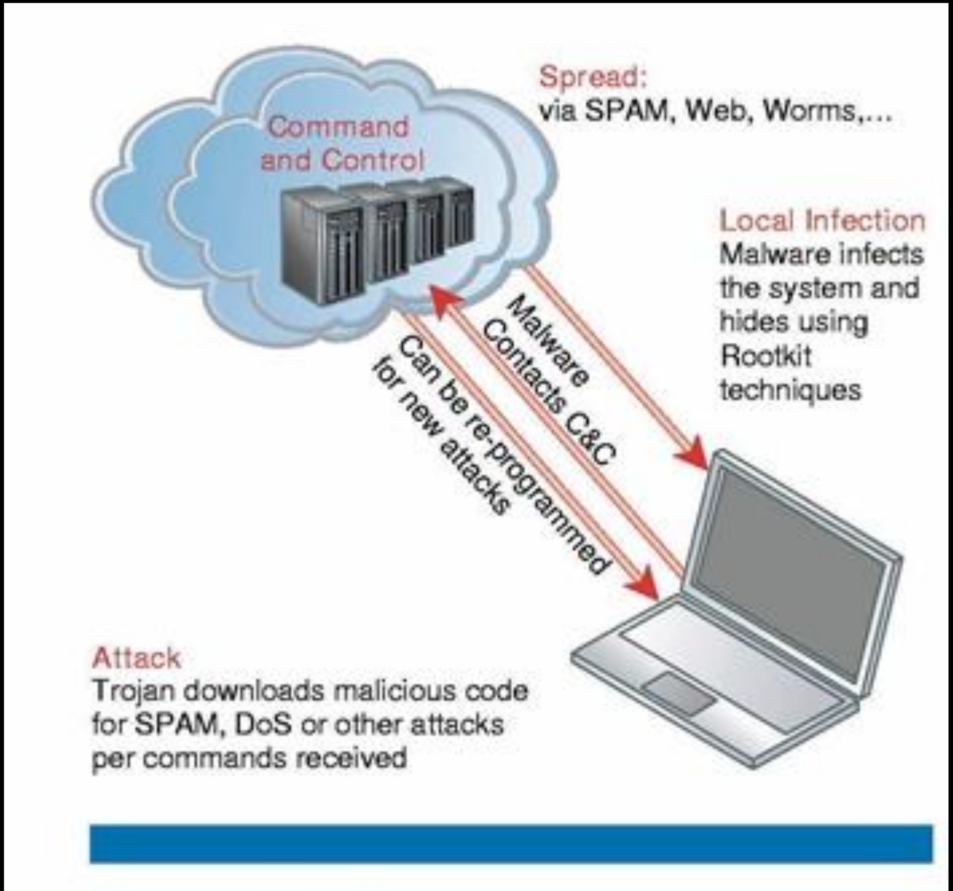
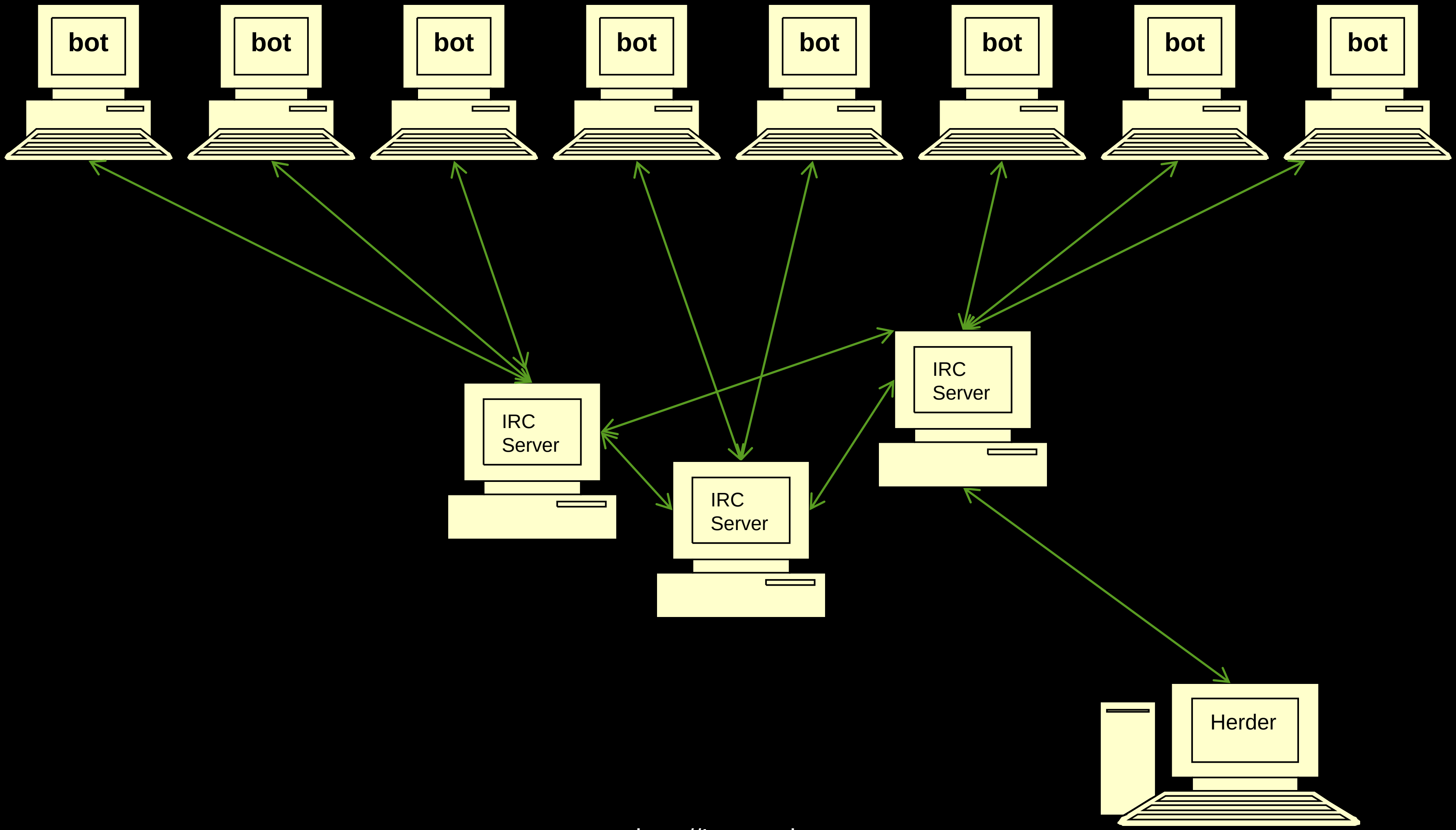

Command and Control
Spread:
via SPAM, Web, Worms,...
Local Infection
Malware infects the system and hides using Rootkit techniques
Malware Contacts C&C
Can be re-programmed for new attacks
Attack
Trojan downloads malicious code for SPAM, DoS or other attacks per commands received

Figure 1: The Botnet Lifecycle

Image Source: Intel Technology Journal; Jun2009, Vol. 13 Issue 2, p130-147

http://Irongeek.com

Illustration of C&C: Another take



http://Irongeek.com

Illustration of C&C: Yet another take

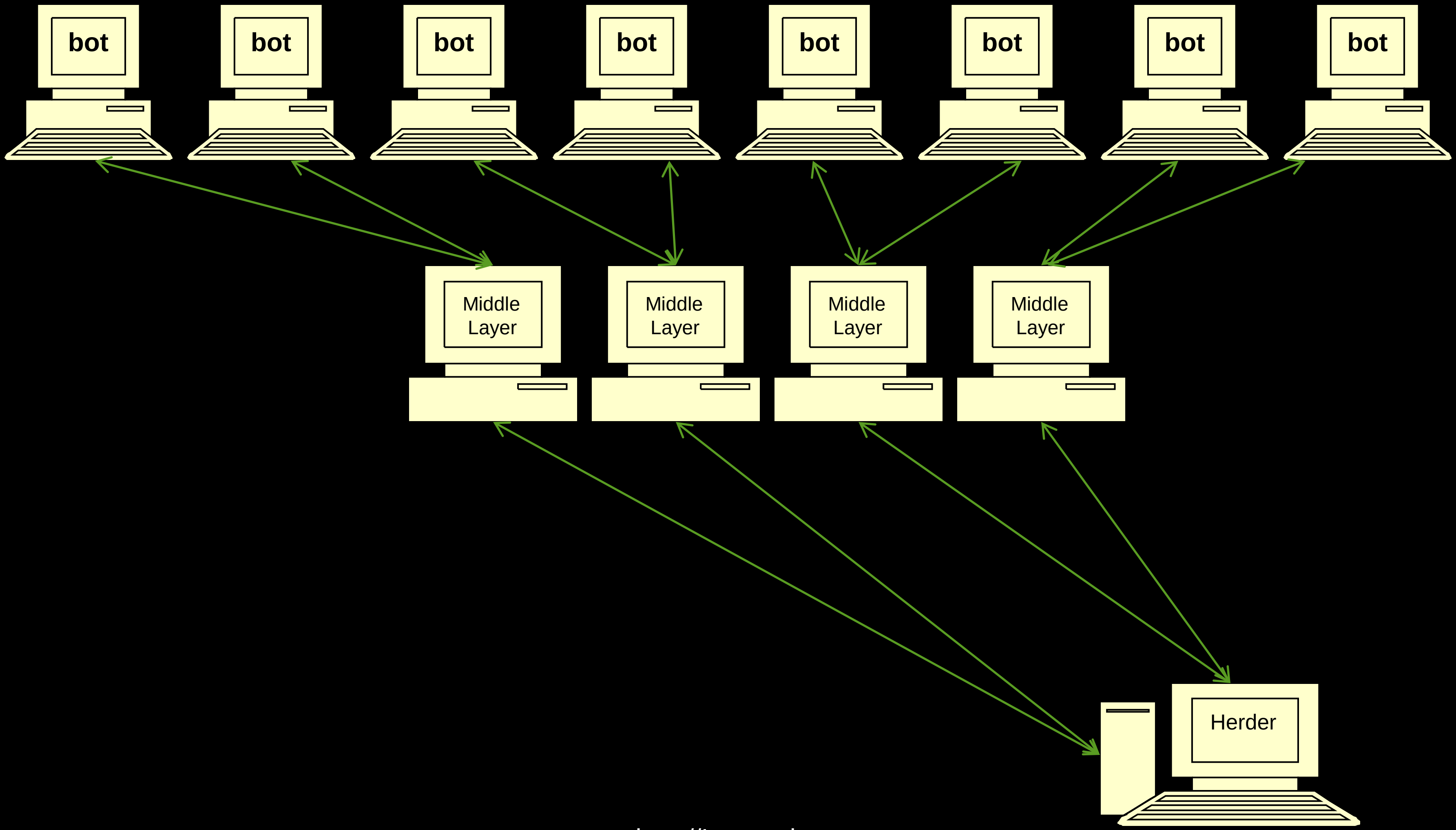
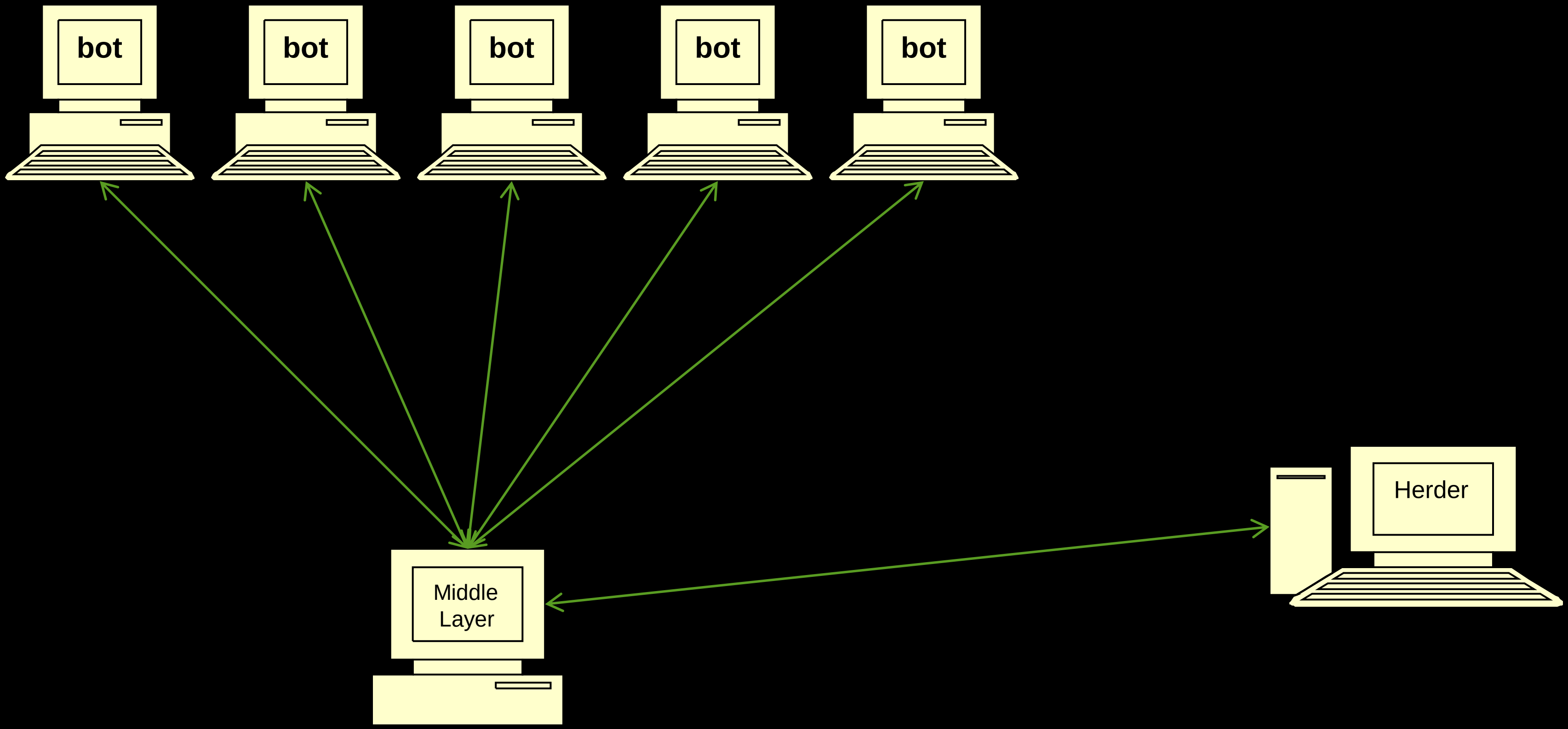


Illustration of C&C: Blind drop



Could be a web site, forum posting, image, etc

Economics of Bot Herding

- So, why would some one want a Botnet?
  - Distributed Denial Of Service (DDoS)
    - Personal vendettas
    - Protection money
  - Spam (both email and web posts)
  - Adware
  - Click Fraud
  - Harvested identities (Sniffers, Key Loggers, Etc.)
- They can also be rented out for tasks
- BBC show Click rents a Botnet: http://www.tudou.com/programs/view/13Cx-LNrTfU/

Problems with detecting/removing a Bot installation

Main points from the article:

- Polymorphism
- Rootkitting
- Only periodic communications back to controller

Others:

- Retaliation Denial of Service
- Distributed
- Fast Flux
- Encrypted channels

Article’s proposal: Canary Detector

Made with three main strategies (paraphrased):

1. Establish a baseline for the network.
2. Use end-host detection algorithm to determine botnet C&C channel, based on destinations that are regularly contacted.
3. Aggregate information across nodes on the network to find commonality.

Canary Detector: Atoms

- Uses the tuple:
  - destIP/dstService = Host being contacted
  - destPort = Port number
  - proto = UDP or TCP
- Examples:
  - (google.com, 80, tcp)
  - (208.67.222.222, 53, udp)
  - (ftp.nai.com, 21:>1024, tcp)
  - (mail.cisco.com,135:>1024,tcp)

Canary Detector: Persistence

- Look for “temporal heavy hitters”
  - Not so concerned about amount of traffic
  - Concerned about regularity
- Starting with a small tracking window (w) time, track if an Atom was contacted or not
- Set an observational time window (W), for example W=10w in duration
- The authors also use multiple time scales 1 through 5

Canary Detector: Commonality

- How common is a destination Atom amongst network nodes?
- The more common the Atom, the more important it is

Canary Detector: Whitelists

- Ignore “safe” Atoms to easy computation
  1. Observe traffic during training period to see common, regularly contacted Atoms (Windows update servers might be an example)
  2. Set nodes to ignore, adjust as needed.
  3. Whitelists are established at both the host and network level.

Canary Detector: Alarm Types

- p-alarms (persistence): When a destination Atom not contained in the host’s whitelist becomes persistent. More for local use, whitelist or flag.
- c-alarms (commonality): When a destination atom is observed at a large number of end-hosts in the same window and is identified as common. More for network use, whitelist or flag.

Using the information

- Article defines thresholds for persistence and commonality (p* and c*) for when to take note
- Suspicious alarms can be acted upon
  - Nullrouting
  - Investigation
  - Cleanup

Tested against real bots

- SDBot: *Controlled over IRC, but easy to spot because of connecting to* irc.undernet.org. Scans ports scans on ports 135, 139, 445, 2097 looking to spread.
- Zapchast: Five IRC service atoms (about 13 distinct IPs). Mostly NetBIOs attack traffic.
- Storm: P2P based. The traces were two orders of magnitude larger than the other botnets tested.

Graph of botnet Atom persistence

- SDBot (Triange)
- Zapchast (Dimonds)
- Storm (Blue Dots)
  - – Note that they only graphed 100 atoms

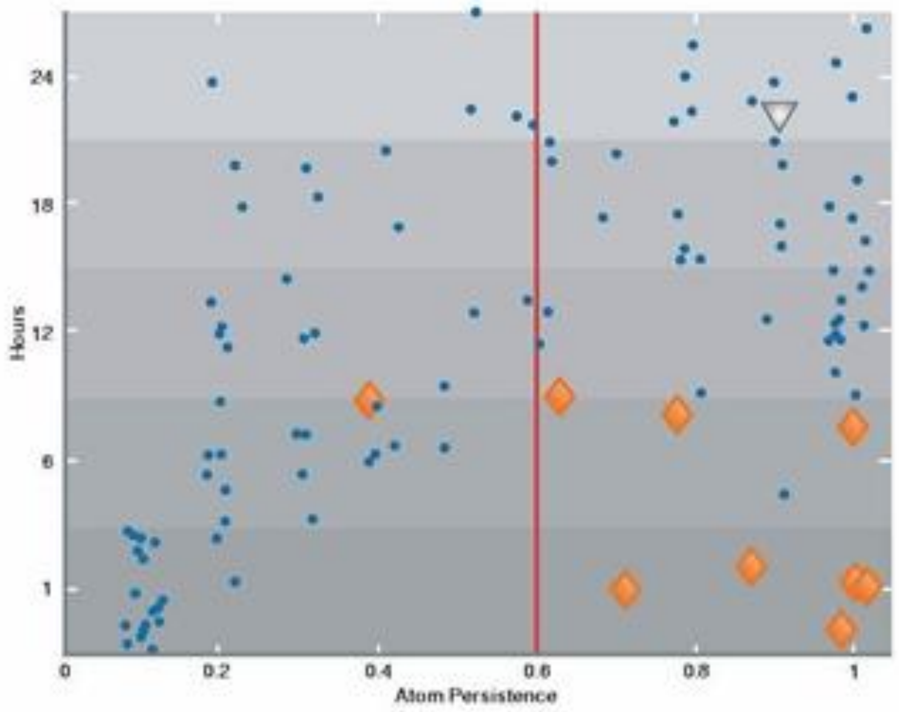


Figure 3: Detection by Persistence of Three Botnets

Image Source:
THE DARK CLOUD: UNDERSTANDING AND DEFENDING AGAINST BOTNETS AND STEALTHY MALWARE
Intel Technology Journal, Jun2009, Vol. 13 Issue 2, p130-147, 18p, 3 charts, 3 diagrams
Diagram; found on p144

Links for more research

- The Dark Cloud: Understanding and Defending against Botnets and Stealthy Malware
  http://download.intel.com/technology/itj/2009/v13i2/pdfs/ITJ9.2.9-Cloud.pdf

- Shadow Server
  http://www.shadowserver.org

- SANs Internet Storm Center
  http://isc.sans.org/

- Honeynet Project
  http://www.honeynet.org

- LAN of the Dead
  http://www.irongeek.com/i.php?page=security/computerzombies

Conclusions/Questions

- How difficult is it to choose good thresholds for persistence/commonality?
- What if Botnets varied their call back times?
- System overhead?
- Whitelisting of services that have become blind drops?
- Audience questions?