

botsnoopd  
Efficiently Spying on Botnets

Georg Wicherski  
gw@mwcollect.org

GovCERT Symposium 2008

September 16, 2008





- 1 Introduction
- 2 botsnoopd Architecture
- 3 botsnoopd Integration
- 4 Availability
- 5 Observations & Future Work



- Georg Wicherski
  - Student of computer science at RWTH Aachen University
    - B.Sc. expected 2010
  - Software Development Contractor with RWTH Aachen
    - botsnoopd Project for "Bundesministerium für Sicherheit in der Informationstechnik"
  - mwcollect Alliance Founder, developer at mwcollect.org
    - The HoneyNet Project: Giraffe Chapter
  - EmsiSoft GmbH (A-Squared AV), CTO
    - Static Zero Pre-Knowledge Malware Detection

**gw@mwcollect.org**



- Georg Wicherski
  - Student of computer science at RWTH Aachen University
    - B.Sc. expected 2010
  - Software Development Contractor with RWTH Aachen
    - botsnoopd Project for "Bundesministerium für Sicherheit in der Informationstechnik"
  - mwcollect Alliance Founder, developer at mwcollect.org
    - The HoneyNet Project: Giraffe Chapter
  - EmsiSoft GmbH (A-Squared AV), CTO
    - Static Zero Pre-Knowledge Malware Detection

**gw@mwcollect.org**



- $\geq 4k$  Botnets alive at any time (ShadowServer, Sept. 2008)
- $\geq 20k$  DDoS every day (Arbor Networks, Sept. 2008)
- Identity Theft, Fraud, Spam, ...



- $\geq 4k$  Botnets alive at any time (ShadowServer, Sept. 2008)
- $\geq 20k$  DDoS every day (Arbor Networks, Sept. 2008)
- Identity Theft, Fraud, Spam, ...
  
- No public, dedicated botnet monitoring software available (to our knowledge)
  - Manually running irrssi: it's scriptable but not scalable.
  - ShadowServer: Perl Script, one Instance per Network, .ini Files for configuration
    - Dedicated Solution in Development
  - Microsoft's BMAT: GUI Application running on Windows



- $\geq 4k$  Botnets alive at any time (ShadowServer, Sept. 2008)
- $\geq 20k$  DDoS every day (Arbor Networks, Sept. 2008)
- Identity Theft, Fraud, Spam, ...
- No public, dedicated botnet monitoring software available (to our knowledge)
  - Manually running irrssi: it's scriptable but not scalable.
  - ShadowServer: Perl Script, one Instance per Network, .ini Files for configuration
    - Dedicated Solution in Development
  - Microsoft's BMAT: GUI Application running on Windows
- BSI funded development of a new GPL solution for the (closed) community
  - "If we provide the operational community with tools, we will benefit in the end."



- 1 Introduction
- 2 botsnoopd Architecture**
- 3 botsnoopd Integration
- 4 Availability
- 5 Observations & Future Work



- POSIX compatible Daemon written in C++ with portability in mind (autotools)
  - Scalable, distributable, location independent (remote access)
  - PostgreSQL database as backend, allows webfronted or other frontends



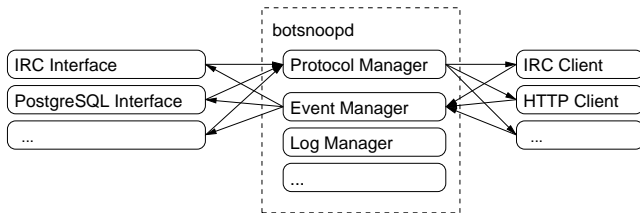
- POSIX compatible Daemon written in C++ with portability in mind (autotools)
  - Scalable, distributable, location independent (remote access)
  - PostgreSQL database as backend, allows webfronted or other frontends
- Asynchronous single-threaded implementation using (e)poll for maximum performance
- Module (Plug-In) based design to support continuous development and addition of new C&C protocols
- Networking and utility code (configuration parsing, modularization, ...) in a separate library `libnetworkd` for maintainability



- Client modules for connecting to the botnets
  - `client-irc` for classical IRC botnets
  - `client-http` for HTTP poll based botnets
- Interface Modules for Communication with the outside world
  - `interface-postgres` gets networks from database and stores commands there
  - `interface-irc` for debugging
- ...



- Client modules for connecting to the botnets
  - `client-irc` for classical IRC botnets
  - `client-http` for HTTP poll based botnets
- Interface Modules for Communication with the outside world
  - `interface-postgres` gets networks from database and stores commands there
  - `interface-irc` for debugging
- ...





- Botnet monitoring daemon not limited to IRC networks
  - Generic representation of a network desired, so layout doesn't need to be adjusted for new C&C protocols
- One Network
  - GUID, host, port, authentication information
  - $n$  locations
    - name, authentication
  - $n$  parameters
    - name, value
- Direct realisation in the DB schema
  - Tables networks, locations, parameters



- Traditional IRC Botnet:
  - `host = "own3d.mwcollect.org"`
  - `port = 31337`
  - `authentication = "s3kr1t"`
    - Send as `PASS` at connection begin
  - One C&C channel → one location (row):
    - `name = "#botnet"`
    - `authentication = "m4st3r"`
  - A couple of parameters:
    - `name = "nick", value = "USA|2K|12345"`
    - Username, Realname, overwriting of keywords, ...
- HTTP Botnet: host and port identify webserver, locations identify relative paths



After all this dry technical stuff...

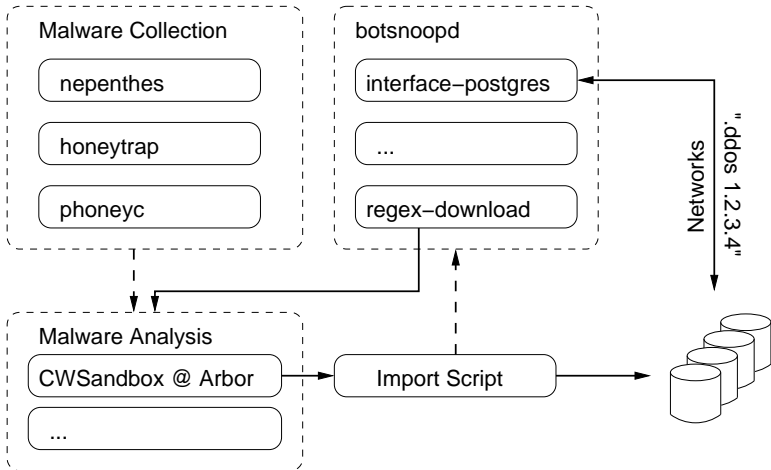
**Fancy DEMO**



- 1 Introduction
- 2 botsnoopd Architecture
- 3 botsnoopd Integration**
- 4 Availability
- 5 Observations & Future Work



# botsnoopd Setup Scenario: mwcollect.org Alliance





- Requirement from the BSI: Easy to use for LE officers, yet powerful data correlation possibilities
- Web 2.0 thanks to ExtJS, allows Drag & Drop of Filters and data correlation
- Maintenance of Networks: Add or remove networks, trigger reconnect, change parameters, ...



- Requirement from the BSI: Easy to use for LE officers, yet powerful data correlation possibilities
- Web 2.0 thanks to ExtJS, allows Drag & Drop of Filters and data correlation
- Maintenance of Networks: Add or remove networks, trigger reconnect, change parameters, ...

After more of this technical stuff...

# Another DEMO



- 1 Introduction
- 2 botsnoopd Architecture
- 3 botsnoopd Integration
- 4 Availability**
- 5 Observations & Future Work



- Code is licensed under the GNU Public License
- Copyright at the programmers, not BSI or RWTH → really "free as in beer"
  - Mostly me for botsnoopd, Mark Schlößer for BotMon
- BSI funded the development with one man-year and some hardware
- Gentlemen's agreement with the BSI not to publically distribute botsnoopd
- We can however pass it to whomever we consider trusted as can they

**<https://botsnoopd.mwcollect.org/>**



- ① Run your own instance of botsnoopd
  - Allows you to customize and automate event handling
  - Needs some time to setup
  - You need your own sources of malware (analyses) to feed
  - If everybody does this, soon there will be  $n$  drones in each channel



- ① Run your own instance of botsnoopd
  - Allows you to customize and automate event handling
  - Needs some time to setup
  - You need your own sources of malware (analyses) to feed
  - If everybody does this, soon there will be  $n$  drones in each channel
  
- ② Participate in the mwcollect Alliance's Setup
  - You save the time to set everything up
  - We'd ask you to run a proxy server for us
  - You're relying on a third party



- 1 Introduction
- 2 botsnoopd Architecture
- 3 botsnoopd Integration
- 4 Availability
- 5 Observations & Future Work**



- The usual stuff:
  - DDoS against some Dial-Up lines
  - Collection of pstore data (Firefox saved passwords)
  - {DCOM, LSASS, VNC, ...} Scanning



- The usual stuff:
  - DDoS against some Dial-Up lines
  - Collection of pstore data (Firefox saved passwords)
  - {DCOM, LSASS, VNC, ...} Scanning
- Boring modifications of the IRC protocol
  - JOIN → J01N



- The usual stuff:
  - DDoS against some Dial-Up lines
  - Collection of pstore data (Firefox saved passwords)
  - {DCOM, LSASS, VNC, ...} Scanning
- Boring modifications of the IRC protocol
  - JOIN → J01N

=qgarAdHHPnHisPyXB4DN3p8k48W4nDad0T7/W9KAUQAI sdk j65Q

- *Encrypted* Commands / Topics
  - reverse base64 decode → 32bit XOR → bitwise ROR
  - XOR and ROR keys differ from specimen to specimen in family
  - Bruteforce feasible if commands are known



- Automatic understanding and interpretation of commands
  - Heuristics, educated guessing
    - Match for known command strings like `.skysyn`, etc.
    - If command string is not known, guess from parameters and order: `?foo google.com 300 600` could be rbot DDoS
  - Unreliable, unsexy, doesn't help with *encryption*



- Automatic understanding and interpretation of commands
  - Heuristics, educated guessing
    - Match for known command strings like `.skysyn`, etc.
    - If command string is not known, guess from parameters and order: `?foo google.com 300 600` could be rbot DDoS
  - Unreliable, unsexy, doesn't help with *encryption*
- Semi-Dynamic analysis of bot samples
  - Dataflow tracking to determine input processing routines
  - Combine with Controlflow tracking to identify command string meanings
- Sophisticated, hard to implement, being worked on ...



# Questions?



Georg Wicherski

[gw@mwcollect.org](mailto:gw@mwcollect.org)

<http://www.mwcollect.org/contact>