Chapter 0x01

INTRODUCTION
Who we are
Here

• Claudio “nex” Guarnieri @botherder
  • Security Researcher at iSIGHT Partners
  • Core Member at The Shadowserver Foundation
  • Full Member at The Honeynet Project
  • Pizza, pasta, Ferrari
  • Cuckoo Creator and Lead Developer
Not here

- Alessandro “jekil” Tanasi @jekil
  - Dude from Hostmap, SecDocs
  - Cuckoo Core Developer and Fussy Bitch Engineer

- Dario “bagode” Fernandes
  - Google Summer of Code 2011 student
  - Cuckoo Windows components developer
Chapter 0x02

AUTOMATED MALWARE ANALYSIS
Problems

• Malwares in the wild are way too many
• Manual analysis takes a lot of time
• Static analysis requires strong skillsets
• Need to deal with packed, polymorphic, self-modifying code
• Performing dynamic analysis manually is a tedious work
Pros

• Can automate the whole analysis process
• Process high volumes of malwares
• Usable by virtually anyone
• Get actual executed code
• Can tweak to do cool sh1t
• Automating is cool
• Automating is cool
• Automating is cool
Lets you focus on more important duties

and still get paid
Cons

• Commercial solutions are very expensive
• Some portions of the malware code could be not triggered
• Environment could be detected
• Difficult to successfully automate exploit analysis
• Without proper consumption of the results, it gets useless
Preparation

• Need to define your requirements and expectations
• Need to design the analysis environment carefully
• Need to design and implement a proper use of the data and integration with other systems and storages
Ask yourself #1

• Why do I need a Sandbox?
• What do I expect to achieve?
• What information is most relevant to me or to my organization?
• Who is gonna consume the results and what for?
• How can I make it easily consumable
Ask yourself #2

• Do I want to analyze PDF exploits?
• Do I want to analyze Office exploits?
• Do I want to analyze PHP and Perl scripts?
• Do I want to analyze browsers’ exploits?
• What else do I want to analyze?
• Do I want it to communicate with the outside?
Chapter 0x03

CUCKOO SANDBOX
What is it

- Automated malware analysis system
- Uses virtualization
- Easy to use
- Easy to customize
- Every single piece of it it’s Open Source!
History

- Google Summer of Code 2010
- DRG Security Innovation Grant 2011 finalist
- Google Summer of Code 2011
- Malwr.com
- Google Summer of Code 2012
- Rapid7 Magnificent7 winner of 1st round
  http://community.rapid7.com
It can

• Analyze PE32, PDF, DOC, URLs, PHP, Perl, Python scripts... you name it
• Be fully customized to do whatever you want
• Be integrated in larger threat intelligence frameworks
It generates

- Win32 functions calls trace
- Dropped files
- Screenshots
- Network traffic dump
- Comprehensive reports
BEING UNSTABLE & BITCHY IS ALL PART OF MY MYSTIQUE
Components

- Analyzer
- Cuckoo Scheduler
- `cmonitor`
- `chook`
Scheduler

• Main component
• **Dispatches** the pending tasks to the pool of machines available
• **Runs all the juicy modules** we’re gonna see in a bit
• 100% Python
Analyzer

• Component that *instruments the guest machine*
• Chosen *depending on the platform* of the selected machine
• Only Windows now, but *can support more*
• Runs the malware and do stuff with it
• 100% Python
Cmonitor

- DLL using chook to install hooks on predefined win32 functions inside process memory
- Gets injected into the target process (QueueUserAPC or CreateRemoteThread)
- Logs the functions calls to files
Chook

- Custom inline hooking library
- Allows definition of custom hook trampolines
- Replaced Microsoft Detours
1 FARPROC addr;
2 addr = GetProcAddress(LoadLibraryA("kernel32.dll"), "CreateFileW");
3
4 if(*(BYTE*)addr == 0xE9) // Hook detected
Reason #2
Execution flow

- Fetch a task
- Prepare the analysis
- Launch analyzer in virtual machine
- Execute an analysis package
- Complete the analysis
- Store the results
- Process and create reports
/cuckoo.py

Cuckoo Sandbox 0.4-dev
www.cuckoobox.org
Copyright (c) 2010-2012

Submission

• From command-line, Python API or SQLite DB
• Specify file path
• Specify analysis package and its options
• Specify machine to be used or operating system
• Specify timeout, priority
Modules & Customization

- Analysis Packages
- Machine Managers
- Processing
- Reporting
- Signatures
Analysis Packages

• **Python classes 😊**

• Defines how the analyzer should **start and interact with the malware**

• Specified at submission or selected upon file type

• Can **create as many as you want** and do whatever you want
from lib.common.abstacts import Package
from lib.api.process import Process

class Exe(Package):
    def run(self, path):
        p = Process()

        if "arguments" in self.options:
            p.execute(path=path, args=self.options["arguments"], suspended=True)
        else:
            p.execute(path=path, suspended=True)

        p.inject()
        p.resume()

        return p.pid

    def check(self):
        return True

    def finish(self):
        return True
from lib.common.abstracts import Package
from lib.api.process import Process

class DOC(Package):
    def run(self, path):
        arg = "\%s\" % path
        p = Process()
        p.execute(path="C:\Program Files\Microsoft Office\Office12\WINWORD.EXE", args=arg, suspended=True)
        p.inject()
        p.resume()

        return p.pid

    def check(self):
        return True

    def finish(self):
        return True
Other examples

• Honeyclient?
• Banking trojan analyzer
• USB Honeypot
• Up to you...
genesis:src nex$ tree -d modules/
modules/
├── machinemanager
│   ├── processing
│   └── reporting
│       └── signatures
└── 4 directories
Machine Managers

• Yes, Python classes 😊
• Define interaction with virtualization software
import subprocess

from lib.cuckoo.common.abstractions import MachineManager
from lib.cuckoo.common.exceptions import CuckooMachineError

class VirtualBox(MachineManager):
    def start(self, label):
        if self.config.getboolean("virtualbox", "headless"):
            subprocess.call(["VBoxHeadless", "-startvm", label])
        else:
            subprocess.call(["VBoxManage", "startvm", label])

    def stop(self, label):
        subprocess.call(["VBoxManage", "controlvm", label, "poweroff"])
        subprocess.call(["VBoxManage", "snapshot", label, "restorecurrent"])

Processing

• **Python classes**, again 😞
• Modules used to **generate a container** of normalized information on the analysis
• Can create as many as you want
from lib.cuckoo.common.utils import File
from lib.cuckoo.common.abstracts import Processing

class FileAnalysis(Processing):
    def run(self):
        self.key = "file"
        file_info = File(self.file_path).get_all()
        return file_info
```python
import os
import urllib
import urllib2
import simplejson

from lib.cuckoo.common.utils import import File
from lib.cuckoo.common.abstractions import import processing

VIRUSTOTAL_URL = "https://www.virustotal.com/vtapi/v2/file/report"
VIRUSTOTAL_KEY = ""

class VirusTotal(Processing):
    def process(self):
        self.key = "virustotal"
        virustotal = []

        if not os.path.exists(self.file_path):
            return virustotal

        md5 = File(self.file_path).get_md5()
        parameters = {"resource": md5, "apikey": VIRUSTOTAL_KEY}
        data = urllib.urlencode(parameters)
        req = urllib2.Request(VIRUSTOTAL_URL, data)
        response = urllib2.urlopen(req)
        virustotal = simplejson.loads(response.read())

        return virustotal
```
Signatures

- **Python classes!**
- Look for patterns or **specific events**
- Assign them a **description** and **severity level**
- Give context to the reports
- Help non-malware experts understand
- Can be used to receive **email alerts**
from lib.cuckoo.common.abstracts import Signature

class CreatesExe(Signature):
    name = "creates_exe"
    description = "Creates a Windows executable on the filesystem"
    severity = 2

    def run(self, results):
        for file_name in results["behavior"]["summary"]["files"]:
            if file_name.endswith(".exe"):
                self.data.append({"file_name" : file_name})
                return True
        return False
from lib.cuckoo.common.abstracts import Signature

class PDFUseFlash(Signature):
    name = "pdf_use_flash"
    description = "PDF document loads embedded Flash " \ 
    "(possibly exploiting a Flash Player vulnerability)"
    severity = 3

    def run(self, results = None):
        if not "PDF" in results["file"]["type"]:  
            return False

        for process in results["behavior"]["processes"]:  
            if process["process_name"] != "AcroRd32.exe":  
                continue

            for call in process["calls"]:  
                if call["api"] == "LoadLibraryW":  
                    for argument in call["arguments"]:  
                        if argument["name"] == "lpFileName":  
                            if "authplay.dll" in argument["value"] or \ 
                            "AuthPlayLib" in argument["value"]:  
                                return True

        return False
Reporting

- OMG Python classes 😞
- Use the normalized results and do something with them
- Can create as many as you want
import os
import json

from lib.cuckoo.common.abstracts import Report
from lib.cuckoo.common.exceptions import CuckooReportError

class JsonDump(Report):
    def run(self, results):
        try:
            report = open(os.path.join(self.reports_path, "report.json"), "w")
            report.write(json.dumps(results, sort_keys=False, indent=4))
            report.close()
        except (TypeError, IOError) as e:
            raise CuckooReportError("Failed to generate JSON report: %s" % e.message)
or mongo!
Community Effort

• Create a **community repository** for sharing modules & signatures
• **Expand our line-up** of developers and contributors
• Make **Malwr.com** a major community resource for malware research
Future Work

• A full-fledged web interface
• Improve Windows analysis components
• Support for other operating systems, Mac OS X?
• Support native machines
Websites

- http://cuckoosandbox.org
- http://github.com/cuckoo-box/cuckoo
- http://blog.cuckoo-box.org
- http://malwr.com
- http://www.honeynet.org
THANK YOU!
NOW LET’S GET SOME LUNCH!

claudio@shadowserver.org