Ethical Hacking and Countermeasures

Version6

Module LXIII

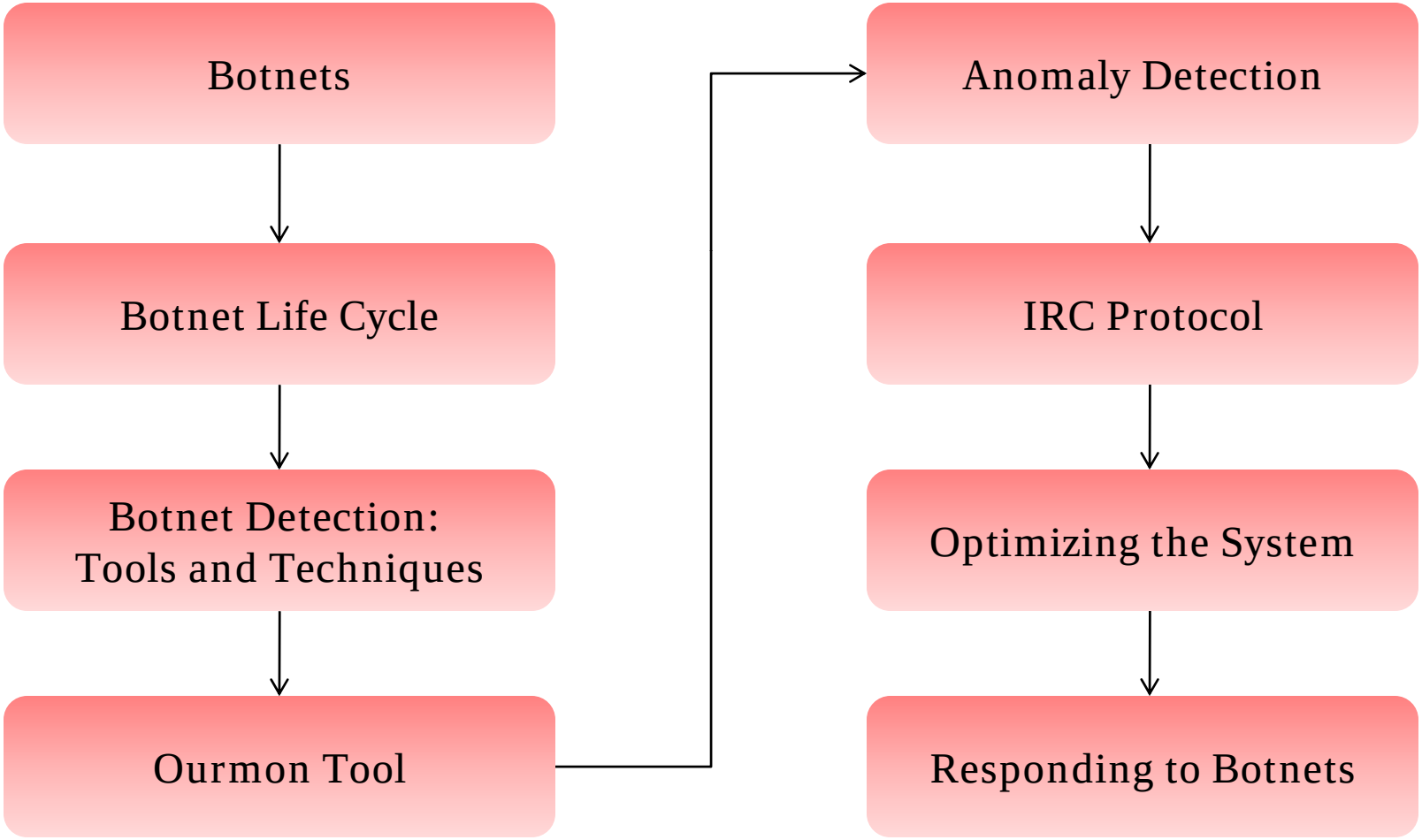
Botnets

Module Objective

This module will familiarize you with:

- Botnets
- Botnet Life Cycle
- Botnet Detection: Tools and Techniques
- Ourmon Tool
- Anomaly Detection
- IRC Protocol
- Optimizing the System
- Responding to Botnets

Module Flow



What Is a Botnet?

A botnet consists of at least one bot server or controller and one or more botclients in many thousands

The ability of the botnet to act in a coordinated fashion with all or some parts of the botnet is fundamental to the botnet concept

Botnets are managed by a botherder

Hackers are attracted to botnets because botnet clients carry out their orders on computers that are at least two computers removed from any computer directly connected to them

The Botnet Life Cycle

Computer exploited and becomes a botclient

New botclient rallies to make botherder aware that he/she has joined the botnet

Retrieve the latest Anti-A/V module

Secure the new botclient from A/V, user detection, and other hacker intervention

Listen or subscribe to the C&C Server/Peer for commands

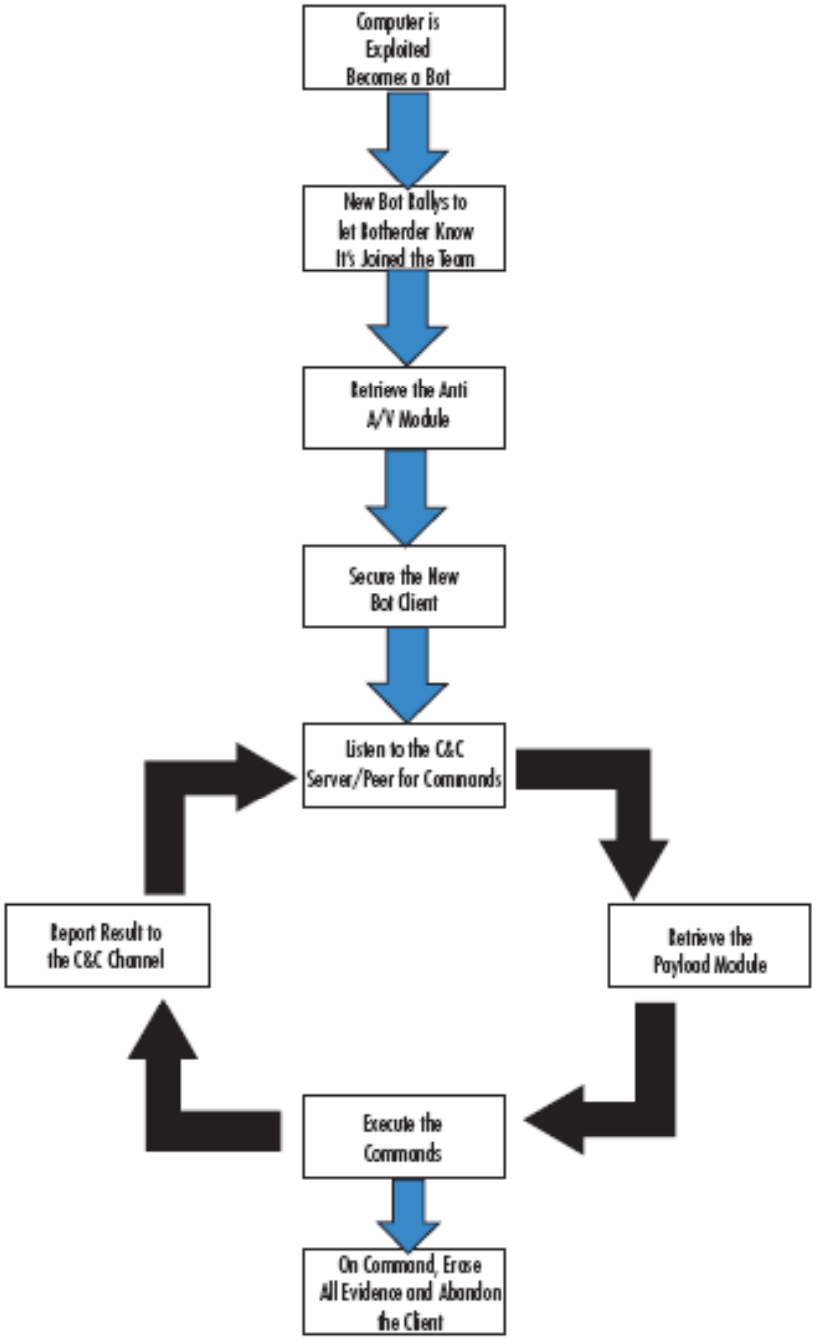
Retrieve the payloads modules

Execute the commands

Report results back to the C&C server

On command, erase all evidence and abandon the client

The Botnet Life Cycle (cont'd)

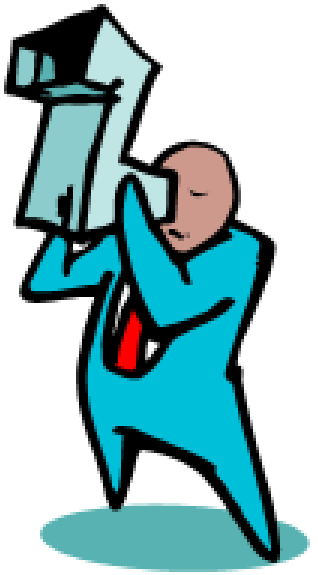


Uses of Botnets

- Recruits other botclients (sniffing for passwords, scanning for vulnerable systems)
- Conducts DDoS attacks
- Harvests identity information and financial credentials
- Conducts spamming and phishing campaigns
- Scams adware companies
- Installs adware for pay without the permission of the user
- Stores and distributes stolen or illegal intellectual property (movies, games, etc.)

How to Identify Whether Your Computer is a Botnet

- If your computer runs slower than normal
- If the network activity light on your DSL modem or NIC card flashes rapidly
- If your antivirus program shuts off by itself
- If it is still running, it may detect several types of malicious code simultaneously
- Run TCPView and examine all the network connections and the processes that are associated with them
- Run Process Explorer and examine all the processes to see if any process is running that does not run on your computer normally
- Check the security event log for login failure for network type 3 where the workstation's name does not match the local computer's name



Common Botnets

- SDBot
- RBot
- Agobot
- Spybot
- Mytob

SDBot

SDBot takes advantage of the insecure network shares or uses known vulnerability exploits to compromise systems

Once SDBot is able to connect to a vulnerable system, it executes a script that downloads and executes SDBot to infect the system

It typically includes some sort of backdoor that allows an attacker to gain complete access to compromised systems

It spreads primarily via network shares and seeks out unprotected shares or shares that use common usernames or weak passwords

It modifies the Windows registry to ensure that it is started each time Windows starts

RBot

RBot was the first of the bot families to use compression or encryption algorithms

It uses one or more runtime executable packing utilities such as Morphine, UPX,ASPack, PESpin, EZIP, PEShield, PECompact, FSG, EXEStealth, PEX, MoleBox, or Petite to encrypt the bot code

It also leverages a variety of known software vulnerabilities in the Windows operating system and common software applications

It terminates the processes of many antivirus and security products to ensure it remains undetected

Agobot

Agobot infects the computer with the botclient and opens a backdoor to allow the attacker to communicate and control the machine

It has the capability to spread via peer-to-peer (P2P) networks

It modifies the host's file to block access to certain antivirus and security firm web sites

It steals the CD keys from a preconfigured group of popular games

It uses predefined groups of keywords to create filenames designed to entice P2P downloaders

Spybot

- Spybot's core functionality is based on the SDBot family
- It incorporates aspects of spyware, including keystroke logging and password stealing
- It spreads via insecure or poorly secured network shares and by exploiting known vulnerabilities common on Microsoft systems
- It connects to a designated IRC server specified by the Spybot variant and joins an IRC channel to receive commands from a botherder
- It propagates through the same standard means as other bot families

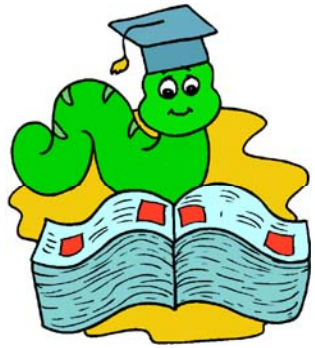
Mytob

Mytob is actually a mass-mailing worm, not a bot, but it infects target systems with SDBot

A hybrid attack that provides a faster means of spreading and compromising systems to create bot armies

It harvests e-mail addresses from the designated file types on the infected system

It eliminates addresses with certain domains to avoid alerting antivirus or security firms of its existence



Botnet Detection: Tools and Techniques

- Abuse E-mail
- Network Infrastructure: Tools and Techniques
- Intrusion Detection
- Darknets, Honeypots, and other snares
- Forensics techniques and tools for Botnet detection

Abuse E-mail

Abuse e-mail list can help to learn about malware at your site

The global registry WHOIS mechanism can help you learn whom to contact at other sites

Spam from your site can cause your site to be blacklisted

Be wary of open proxies in general, and note that they can be the side effect of a malware infection

Network Infrastructure: Tools and Techniques

- Switches have port-mirroring features that allow you to send packets to a sniffer
- Tcpdump and Wireshark are open-source sniffers
- If you find a bot client with a sniffer, also remember to watch for any suspicious external hosts talking to the bot client
- SNMP using RRDTOOL graphics can be useful for seeing DoS attacks via graphics
- Netflow data is more compact than packets and can give you a log of recent network activity
- Netflow tools include open-source tools like flow tools and Silktools
- Netflow can be used to see DoS attacks and scanning as well as more conventional traffic monitoring

Network Infrastructure: Tools and Techniques (cont'd)

- Firewall ACLs can alert you about hosts that have been hacked via their logs
- Firewalls should minimally block Microsoft File Share ports such as 135-139 and 445 as well as SQL ports 1433 and 1434
- Data link layer suffers from various forms of attack, including ARP spoofing, which can lead to MITM attacks
- It can suffer from switch forwarding table overflow attacks, which can lead to password-guessing attacks
- Its switch features can include various security measures such as port security, DHCP snooping, IP Source Guard, and dynamic ARP inspection, especially on recent Cisco switches

Intrusion Detection

Intrusion detection systems are either host or network based

NIDS should focus on local and outgoing traffic flows as well as incoming Internet traffic

HIDS can pick up symptoms of bot activity at a local level that can not be seen over the network

IDS can focus on either anomaly detection or signature detection

Intrusion Detection (cont'd)

Snort is a signature-based NIDS with a sophisticated approach to rule sets, in addition to its capabilities as a packet sniffer and logger

Tripwire is an integrity management tool that uses a database of file signatures to detect suspicious changes to files

The database can be kept more secure by keeping it on read-only media and using MD5 or snefru message digests

Darknets, Honeypots, and Other Snares

- A darknet is an IP space without active hosts and therefore there is no legitimate traffic
- Any traffic that does find its way in is due to mis-configuration or attack
- Intrusion detection systems in that environment can therefore be used to collect attack data
- A honeypot is a decoy system set up to attract attackers
- A low-interaction honeypot can collect less information than a high-interaction honeypot, which is open to compromisation and exploitation
- A honeynet consists of a number of high-interaction honeypots in a network, monitored transparently by a honeywall

Forensics Techniques and Tools for Botnet Detection

Digital forensics is concerned with the application of scientific methodology for gathering and presenting evidence from digital sources to investigate criminal or unauthorized activity, originally for the judicial review

The forensic process at the judiciary level involves strict procedures to maintain the admissibility and integrity of the evidence

There is no single and simple approach present for investigating a suspected botnet

Forensics Techniques and Tools for Botnet Detection (cont'd)

Make the best of all the resources that can help you out from spam and abuse notifications to the logs from your network and system administration tools

Automated reports generated from log reports by tools like Swatch helps to monitor the systems

In the event of a security breach, these tools (reports) give an immediate start on investigating what has happened

Tool: Ourmon

Ourmon detects network anomalies based on hosts that attacks other hosts via denial-of-service (DoS) attacks or by network scanning

It is based on promiscuous mode packet collection on Ethernet interfaces and uses port mirroring via an Ethernet switch

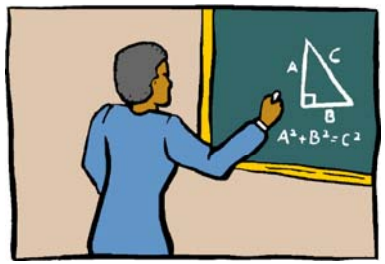
It collects IRC information with its IRC module and uses the TCP report in particular to attempt to figure out if an IRC channel is actually a botnet

A probe collects packets deemed important and sends internally defined tuples back to a graphics display system which may or may not be on the same host

How Ourmon Works

- Ourmon architecturally has two main components, a probe (sniffer) used for packet capture and a back-end graphics engine that makes web pages
- The probe produces outputs in every 30 seconds
- The back-end software produces base-lined data including hourly and daily ASCII reports
- RRDtool graphs include daily, weekly, monthly, and yearly graphs
- Ourmon dynamically creates web pages and logs
- The logs may be used for extracting more details about a particular case and are also used internally by ourmon to produce hourly summarizations

Anomaly Detection



- Anomaly detection depends on baselining of data
- It can point out new anomalies which are abnormal
- Signature detection can tell you if a particular packet or file is evil
- It cannot recognize new evil packets or new evil files and hence is not good at zero-day attacks
- It may only detect anomalies and might not be able to explain them

TCP Anomaly Detection by Ourmon

- The basic 30-seconds TCP port report is a snapshot of individual hosts using TCP; the main goal is to catch TCP-based scanning hosts
- It is sorted by ascending IP address and allows you to spot hacked hosts on the same subnet
- The basic TCP port report includes only hosts with nonzero TCP work weights which may show large parallel scans
- The TCP work weight is a per-host measurement of TCP efficiency
- The TCP port report shows a number of attributes per host, including L3 and L4 destination counts
- These are unique counts of L3 IP destination addresses and L4 TCP destination ports during the sample period

TCP Anomaly Detection by Ourmon (cont'd)

- The TCP port report also includes a SA/S statistic that can indicate that a host is mostly acting as a server
- The report includes a port signature at the end, which is sorted in the ascending order
- The port signature can show that more than one host is doing the exact scan
- The TCP worm graph shows the overall number of scanners, remote or local, as an RRDTOOL graph

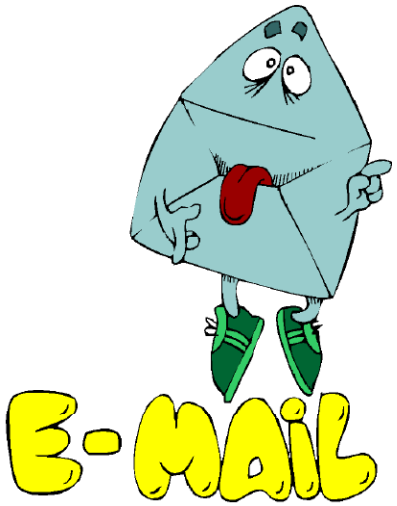
UDP Anomaly Detection by Ourmon

- Ourmon has a 30-second UDP port report that is similar to the TCP port report
- The port report is sorted by the UDP work weight, which represents a per-host value based on the number of UDP packets sent and ICMP errors returned
- The UDP work weight for the top host is graphed in the UDP work weight graph in every 30 seconds
- The UDP anomaly mechanism typically captures UDP scanning systems or UDP DOS attacks
- The default UDP work weight threshold is 10000000
- Any events with UDP work weights larger or equal to this threshold are put in the event log

Detecting E-mail Anomalies using Ourmon

The e-mail syn report has a 30-seconds and hourly summarized form

An e-mail-specific work weight is given so that e-mail connections can be distinguished from other kinds of connections



The e-mail syn report is sorted by e-mail SYN count which is anamoly-based

The e-mail reports may show a local host sending spam

IRC Protocol

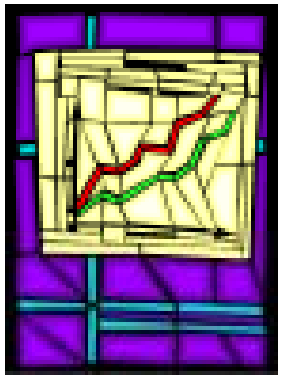
- Internet Relay Chat (IRC) is an Internet Engineering Task Force specified Protocol
- Channels are the fundamental target of data messages; channels are strings in IRC
- The ngrep tool can be used to directly sniff strings on the network
- An IRC network consists of a set of servers and hosts
- Users join a channel and can then send messages to other users
- The messages are distributed by the servers to clients interested in the channel

IRC Protocol (cont'd)

Ourmon looks for four fundamental IRC messages:

- JOINS
  - These are used by an IRC client to log into a channel on a server
- PINGS
  - These are sent from a server to a client to discover if the client is still interested in the channel or not
- PONGS
  - These are returned from the client to the server to show that it does not want to be logged out and still exists
- PRIVMSG
  - It contains both the channel name and data sent to the channel name

Ourmon's RRDTOOL Statistics and IRC Reports



All IRC statistics are found on the irc.html page

The IRC data has three parts:

- RRDTOOL global IRC stats
- Weekly summarizations, including the daily report
- 30-seconds IRC report

The IRC RRDTOOL graph shows message counts for PING, PONG, JOIN, and PRIVMSG IRC messages

The IRC ASCII report shows global, per channel, and per-host statistics

Ourmon's RRDTOOL Statistics and IRC Reports (cont'd)

- The most important parts of the ASCII report are the two channel sorts at the top
- It includes the evil channel sort and the max message sort, as well as the breakdown of each channel with per-host statistics
- The evil channel sort shows IRC channels sorted by the number of scanning hosts in the channel
- The max message sort shows IRC channels sorted by the total number of all four kinds of IRC messages
- The per-channel host statistics show the IP addresses of hosts in an IRC channel as well as other data
- The maxworm field in the per-host statistics is the TCP work weight

Detecting an IRC Client Botnet

An IRC channel having more than a few clients with high maxworm (work weight) values could be a botnet channel

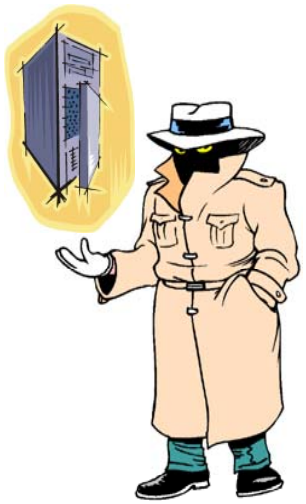
If there are only a few hosts with high work weights, one should search the TCP port report logs to see if the host has been scanning

Non-scanning hosts in an “evil channel” are likely remote botnet servers

It is good to watch those hosts’ behavior with a sniffer

Detecting an IRC Botnet Server

- High and anomalous counts in the RRDTOOL IRC statistics graph could indicate the presence of a local botnet server
- Botnet servers typically have unusual host counts
- They could have unusual counts for remote IP destinations (L3D)
- They might appear in the evil channel sort
- This is due to connection failures by remote exploited hosts



Automated Packet Capture

Ourmon has an automated packet-capture feature that allows packet capture during certain types of anomalous events

Automated packet capture is turned on in the probe config file

Trigger-on and -off events are logged in the ourmon event file, which can be found from the main web page

Triggers of interest for anomaly detection include the trigger_worm trigger, the UDP work weight trigger, and the drops trigger

Automated Packet Capture (cont'd)

- The trigger_worm trigger is used to capture packets when the supplied threshold of scanning IP hosts is exceeded
- The UDP work weight trigger is used for capturing packets when the supplied threshold is exceeded
- The drops trigger is used to capture packets when a supplied dropped packet threshold is exceeded
- This trigger has a poor signal-to-noise ratio and is more likely to succeed if most packets are DoS attack packets
- Captured packets can be viewed with a sniffer such as tcpdump or WireShark

Ourmon Event Log

The event log records both probe and back-end events

The goal of the event log is to store significant security-related events as well as important ourmon system events

Event log stores both bot client mesh detection and bot server detection events

Event logs for roughly a week are kept by the system and made available at the bottom of the main web page

DNS and C&C Technology

IRC is built in a fashion that several servers can be inter-linked to form a network of hubs, branches, and leaves

DNS was manifested in two main uses: domain names and multihoming

Both of them were working as facilitators to find the botnet C&C as well as to keep it alive on the Internet, before connecting to the actual C&C server

Reporting, which results in a “takedown” for a DNS record, is often more difficult than a compromised IP address

Several such RRs could be put in place for the same IP address, or different ones, making the C&Cs much more robust

Tricks for Searching the Ourmon Logs

Log information in ourmon exists in two directories:

- Log directory
- Web directory on the back-end graphics system

In the web directory, IRC summarizations are stored in ircreport_today.txt (today) and ircreport.0.txt (yesterday), ircreport.1.txt (day before yesterday), and so on

In the web directory, syndump (all local host) TCP work weight information is stored in syndump.daily.txt (today), syndump.0.txt (yesterday), and so on

In the web directory, normal TCP work weight information is stored in wormsum.all_daily.txt, wormsum.all.0.txt, and so on

Sniffing IRC Messages

Ngrep is a sniffer designed to search for string patterns, primarily in the application layer payloads

It can be used to look at IRC traffic to and from suspicious IP hosts

Ourmon also includes an additional sniffer called the IRC Flight Recorder (ircfr) that can be used to log all IRC data

This allows the security engineer to look up suspicious IRC hosts or channels in border-line anomaly detection cases to determine whether the host or channel is benign or evil

Sandboxes

Sandboxes protect the local system while executing unknown or malicious code

Protection is achieved either by blocking critical operations completely or by performing them in a virtual environment instead of on the real system

Sandboxes can be integrated into a bigger process of automatic malware analysis

Norman Sandbox or CWSandbox both use a database to store malware samples and the resulting analysis reports

CWSandbox is embedded into the Automated Analysis Suite that comes with the CWSandbox software package

CWSandbox

CWSandbox is a tool for automatic behavior analysis of Windows executables

Steps performed by the CWSandbox are:

- The initial malware process is created by the starter application, cwsandbox.exe
- Cwmonitor.dll is injected into each monitored process
- The DLL installs API hooks for all important functions of the Windows API
- If a new process is started by the malware or if an existing one is infected, this process is also monitored
- After a customizable time, all monitored processes are terminated
- A high-level summarized analysis report is created of all the monitored actions
- The network traffic is examined, important web protocols are recognized, and all relevant protocol data is reported

Operations Revealed by CWSandbox

Reading, writing, or locating objects of the local file system, .ini files, or the registry

Finding active local antivirus or security software

Starting new or terminating active applications

Injecting malicious code into running processes

Reading or modifying the virtual memory of running processes

Operations Revealed by CWSandbox (cont'd)

- Installing, starting, or deactivating Windows Services
- Enumerating, creating, or removing local users
- Reading or writing data from or to the Windows Protected Storage
- Enumerating, creating, removing, and modifying Windows network shares
- Loading and unloading dynamic link libraries (DLLs)
- Querying system information, shutting down or rebooting the system, accessing mutexes, or creating threads

Automated Analysis Suite (AAS)

Automated Analysis Suite (AAS) is a tool for automatic collection and analysis of malware

AAS uses a database to store malware samples and the corresponding created analysis reports

AAS integrates the honeypot tool Nepenthes for automatic malware collection

Additionally, malware can be submitted via a PHP-based Web interface

AAS embeds CWSandbox for automatic analysis

Responding to Botnets

- Improve local security policy authentication practices to prevent password-guessing attacks
- Use firewalls and other containment technologies to limit the scope of attacks
- Update all systems and verify that all systems have accepted and installed the patches
- Every windows host needs a virus checker and possibly a spyware or adware checker
- Send abuse e-mail about remote attacks
- Law enforcement may be invoked, especially if the incident is considered serious for legal or financial reasons

Responding to Botnets (cont'd)

- Darknets, honeynets, honeypot tools, and sandboxes are all useful for determining what is going on in botnet-land
- Shadowserver is an all-volunteer group that tracks and reports on botnets and other malware
- All outbound mails have to go through the official mail servers to prevent botclients from spamming directly to the Internet
- Use networking equipment that supports port security to detect DHCP, IP address, and ARP spoofing
- Develop your sources of internal intelligence

Summary

- A botnet consists of at least one bot server or controller and one or more botclients in many thousands
- SDBot typically includes some sort of backdoor that allows an attacker to gain complete access to compromised systems
- Ourmon detects network anomalies based on hosts that are attacking other hosts via denial-of-service (DoS) attacks or by network scanning
- An IRC channel with more than a few clients with high maxworm (work weight) values could be a botnet channel
- Sandboxes protect the local system while executing unknown or malicious code