# P2L2 Modern Malware

**Past Malware**
In the past malware was for fun and fame. It was used to damage websites or cause denial of service (DoS) attacks.

**Modern Malware**
Now, often for profit and political gains.

Now that money and politics are involved:
- malware is technically sophisticated and is based on the latest technologies.
- malware now is designed for efficiency, robustness, and evasiveness.

**Botnet**
Botnets are the most prevalent form of malware. Most attacks and frauds are due to botnet.

Bot - also known as a zombie - is  compromised computer under the control of an attacker. Bot Code (malware) on the compromised computer communicates with the attacker's server and carries out malicious activities per the attackers instructions.

Botnet - a network of bots controlled by an attacker to perform coordinated malicious activities. The combined power of the botnet means the attacker controls a very large and powerful computer platform. Which allows the attacker to launch a wide variety of malicious activities.

Spamming = infected machines send out unsolicited emails.
Click Fraud = used by botmasters to fraudulently increase revenue from advertisers
Phishing = used to gather valuable financial information

**Attacks and Frauds by Botnets**
Botnets usually have one of two goals: monetary profit or political activism.
Botnets are responsible for:

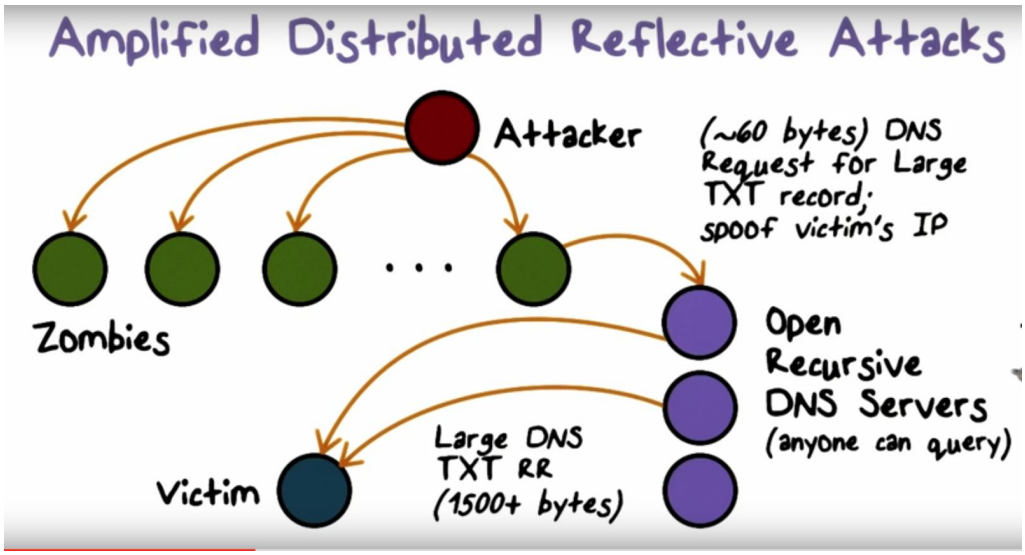
Spam -- nearly all spam is from botnets
DDOS (distributed denial of service) attacks
Click fraud
Phishing and pharming
Keylogging and Data/ID theft
Key/Password Cracking
Anonymized terrorist and criminal communication
Cheat in online games and polls

**DDoS Using Botnets**

1. The attacker chooses a victim
2. The attacker commandeers a botnet. The zombies have the botcode and communicate with the botmaster regularly.
3. The attacker sends a command to all the bots in the botnet to attack at the same time. For example: a request for connection
4. The victim is then overloaded with requests and must deny service.

**Amplified Distributed Reflective Attacks**

To counter the attack, victims can increase their number of servers, but the attacker can amplify the attack.




Open recursive DNS Servers are servers that any machine can query. A common query is to look up the IP address of a domain name. They also answer queries about large TXT records, these are 1500 or more bytes.

The attackers use the DNS servers:

1. the botmaster commands the bots to query the open recursive DNS servers about the TXT records.
2. The address the query is coming from is spoofed - it is the victim's address.
3. When all the bots request the same information be sent to the victim.

**DDos**

- the attacker does not have to use his own computer in the attack
- there are so many computers involved in the attack, it is difficult to distinguish legitimate from malicious traffic.

**Botnet Command and Control**
Botnet is a network of compromised computers that the botmaster uses for malicious purposes. The botmaster needs to control the bots, so control and communication is required.

**Botnet C&C Problem**
How can the botmaster know which computers have been infected and how to communicate with them.

Method 1:

- The victim computers can contact the botmaster.
    - Create malware (vx)
    - Download vx code; fiddle with it; compile it
    - Uses email propagation/social engineering
- Spreading is the easy part, now how do we use the compromised computers (victims)?

The problems with this method:

- The address of the botmaster must be hardcoded into the malware.
- When a sysadmin discovers the bot, they will be able to trace it

This method is not stealthy.

- There is only one rally point for control
- Once the email account is known, it is easy to ban. Thereby cutting off the botmaster from the bot.

This method is not robust

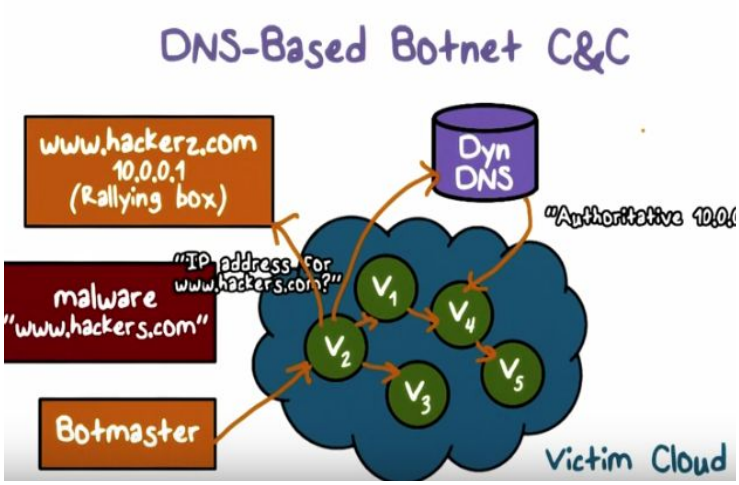
**Botnet C&C Design**
How can bots contact their master safely?

Simple Naive Approach does not work. This method is for script-kiddies, first time malware authors.
Utility and safety are important to the bot masters.
Design considerations:

- Must be efficient and reliable. It must be able to reach a sizeable
- Stealthy - hard to detect (i.e. it must blend with normal/regular traffic)
- Resilient - it should be hard to disable or block C&C traffic

**C&C Design Quiz**
Bots have more sophisticated communication capabilities than worms and viruses.
Bots do NOT require direct communication with the C&C server before beginning an attack.
A bot should not use custom communication protocols because that is not stealthy.

**DNS-Based Botnet C&C**




In this illustration the botmaster releases malware. The domain name of the C&C server is hardcoded.

When it is time for the bot C&C: the bot will ask the DNS for the address of the domain name. Then the bots will communicate with the DNC server.

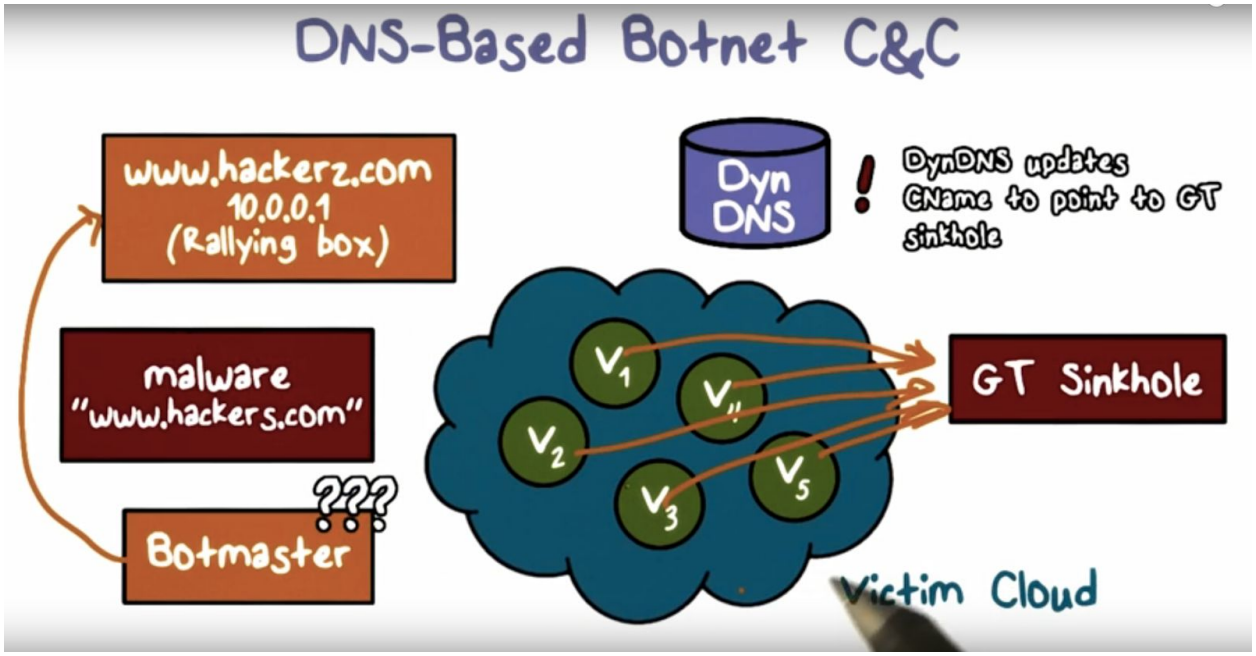
Botmasters prefer dynamic DNS servers because of the frequent change between domain name and IP address. So the botmaster can change the machine used for C&C. He can just change the DNS mapping quickly.

Using DNS for C&C is a good idea because DNS is used anytime a computer needs to talk to another computer. DNS stores the mapping between the domain name and the IP address. DNS is always allowed on a network.

**Anomaly detection:** the way bots look up a domain suggest a domain suggest the domain is most likely used for C&C.

For example: if a domain name is referenced by hundreds of machine across the internet, but the domain is not known to Google search, there is a good chance it is a bot.

Once the domain name has been identified as a botmaster and used for C&C, a number of responses are possible:




- DynDNS can map the domain name to a sinkhole. When a bot asked for the dynamic address of the domain, the DNS will give the address of the sinkhole instead of the botmaster.
- The advantage of a sinkhole: it allows researchers to discover where the bots are in the net.

**Botnet C&C Quiz**

A C&C scheme must provide:

Efficient/reliable communications
Stealth communications
Resilient communications

**Advanced Persistent Threat (APT)**

- Tend to target specific organizations.

Advanced:

- Use special malware. This malware is usually common malware that has been adapted for special operations and operators. The advantage of using common malware is, if a sysadmin detects it, it will not be recognized as APT.
- It is used for high value theft - such as stealing the plans of a new airplane.

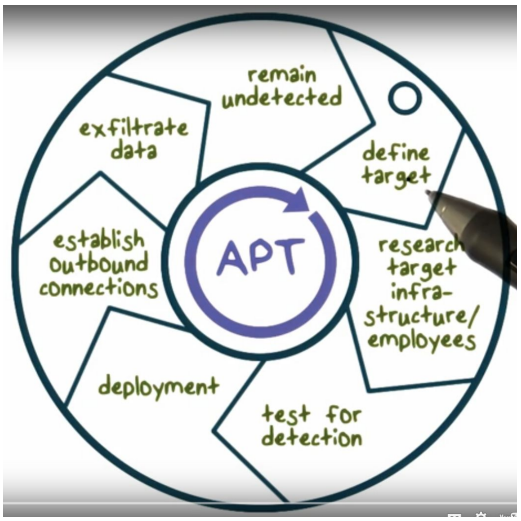
Persistent:

- Long-term presence, multi-step, low-and-slow -
- Once the malware gets into an organization, it will be there for a long time
- It will also take many small steps that will not be detected

Threat:

- The data targeted is high value

**APT Lifecycle**




-It begins by defining at target.
-Then research the target and determine the vulnerabilities of its network services.
-test for detection and use the knowledge of the organization's infrastructure. For example, once the attacker has the name of a high level officer in the company, he can send an email with embedded malware to this person.
-Now the APT has a foothold in the organization. The malware can now establish outbound connections and begin to gather information and pass it to the botmasters.
-The APT will be careful to avoid detection, by keeping its footprint as small as possible.

**APT Characteristics**

Zero-day exploit or a specially crafted malware.
A zero-day exploit takes advantage of a previously unknown weakness or vulnerability in a system. There is no patch or fix for the system or prevention for the attack.

A zero-day exploit will often go undetected. It is usually designed to detect the signatures and behavior patterns in a system.

**Social Engineering -** APTs are designed to trick even the most sophisticated users.
For example: An APT will first compromise core internal network control elements such as routers and web servers to learn about valuable targets.
Once the APT learns who emails who and what attachments and topics are discussed, it can forge emails from users, this is called Spear Phishing because it is targeted against a specific individual.

APTs can also play man-in-the-middle (MITM) on the compromised routers/server to make social engineering attacks very convincing. For example APTs can forge answers to challenges or inquiry by suspecting users.

APTs are designed in a low-and-slow fashion to completely blend in with normal activities.
For this reason it is very hard to detect anomalies by existing approaches.

APTs are a persistent operation that involves multiple deliberate steps over time, rather than a single attack.

**APT Attacks and Characteristics**

Boy in the Middle - covertly changes a computer's network routing
Clickjacking - web users unknowingly click on something that is not as it is portrayed.
Man in the Browser - modifies web pages covertly
Man in the Middle - eavesdrops
Keyloggers - covertly records keystrokes

**APT Example**

A CEO gets an email with a PDF document attached. When he opens the document, there is attack data embedded in it. It breaks out the plug-in sandbox in the browser and compromised the browser by adding an extension.
From this point on, the browser extension will embed the malware into the PDF document. Thus spreading it all over the company. Eventually finding the sysadmin user who has authorization over the server. It can grant itself access over data on the server, allowing to steal valuable data off the servers.
The significant characteristics of this APT:

1. The sysadmin and users do not realize their system has been compromised.
2. The APT activities blend in with normal user activity.
3. The APT moves incrementally and takes time to get to the key individuals.

**Malware Analysis**

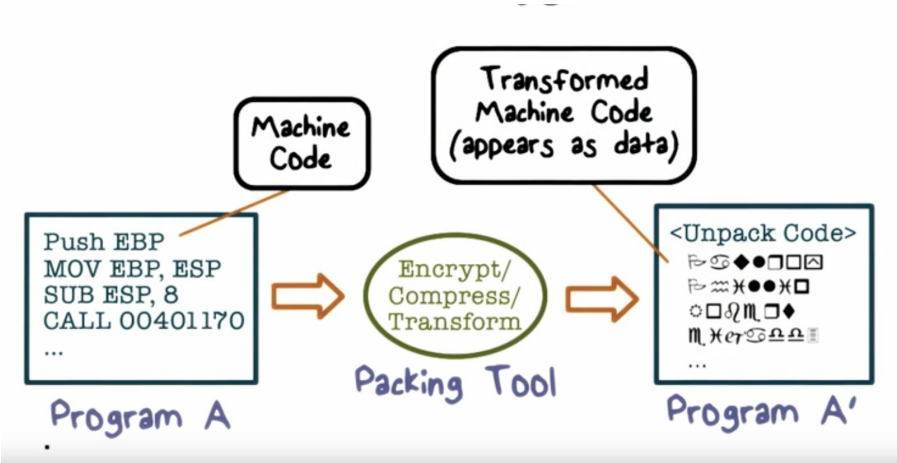
Analysis that is performed for detection/response
Method of Analysis:

- Static Analysis: attempts to understand what a malware instance would do if executed. This is done without actually running the malware.
- Dynamic Analysis: attempts to understand what a program does when executed. Different granularities of analysis:
- Fine-grained (eg automated unpacking) - looking at instruction by instruction
- Coarse-grained (system call tracing) - looking at function calls

Dynamic analysis downfalls: only reveal behaviour of the program during a specific runs. On any particular run, the malware may be waiting for specific conditions to be right before executing some of its code.

**Malware Obfuscation**

Packing: a technique whereby parts or all of an executable file are compressed, encrypted, or transformed in some fashion. This makes part of the program data instead of code. The code that reverses the pre-runtime transformation is included in the executable and is called unpacking.




After using the packing tool the program looks like it contains random data. Each time the packing is performed, it is different. So a signature approach will not work on detecting the malware.
Even though the program contains the code to unpack -- this cannot be used as a signature since a number of legitimate programs use packing/unpacking.

**Unpacking**
Many modern malware programs use packing.

- Leading to thousands of packers, countless ways to obfuscate code.
- Volume of malware samples makes manual unpacking untenable.

So there is a need for automated unpacking that does not require prior knowledge of the code. It also needs to be fine-grained tracing-based universal automated unpacking algorithms.

One method - detect the execution of code not in the static code model.

- Run the malware.
- Determine the code that was not in the program before unpacking. These must be instructions that were unpacked just before execution.
- The other techniques can be used to identify the logic of the malware code.

**Malware Analysis Quiz**
These approaches can be used to detect the malicious browser extension malware.

- a network monitor that analyzes traffic to detect anomalies or known bad traffic.
- a host monitor that examines operating systems activities
- a malware analysis system that identifies malicious logic