COM307000 - Software

Dr. Anca Jurcut
E-mail: anca.jurcut@ucd.ie

School of Computer Science and Informatics
University College Dublin,
Ireland

Future of Malware

Future of Malware

- Recent trends
  - Encrypted, polymorphic, metamorphic malware
  - Fast replication/Warhol worms
  - Flash worms, slow worms
  - Botnets
- The future is bright for malware
  - Good news for the bad guys…
  - …bad news for the good guys
- Future of malware detection?

Encrypted Viruses

- Virus writers know **signature detection** used
- So, how to evade signature detection?
- Encrypting the virus is a good approach
  - Ciphertext looks like random bits
  - Different key, then different "random" bits
  - So, different copies have no common signature
- Encryption often used in viruses today

Encrypted Viruses

- How to detect encrypted viruses?
- Scan for the decryptor code
  - More-or-less standard signature detection
  - But may be more false alarms
- Why not encrypt the decryptor code?
  - Then encrypt the decryptor of the decryptor (and so on…)
- Encryption of limited value to virus writers

Polymorphic Malware

- Polymorphic worm
  - Body of worm is encrypted
  - Decryptor code is "mutated" (or "morphed")
  - Trying to hide decryptor signature
  - Like an encrypted worm on steroids…

Q: How to detect?

A: Emulation — let the code decrypt itself

- Slow, and anti-emulation is possible

Metamorphic Malware

- A metamorphic worm mutates before infecting a new system
  - Sometimes called "body polymorphic"
- Such a worm can, in principle, evade signature-based detection
- Mutated worm must function the same
  - And be "different enough" to avoid detection
- Detection is a difficult research problem

Metamorphic Worm

- One approach to metamorphic replication…
  - The worm is disassembled
  - Worm then stripped to a base form
  - Random variations inserted into code (permute the code, insert dead code, etc., etc.)
  - Assemble the resulting code
- Result is a worm with same functionality as original, but different signature

Warhol Worm

- "In the future everybody will be world-famous for 15 minutes" — Andy Warhol
- Warhol Worm is designed to infect the entire Internet in 15 minutes
- Slammer infected 250,000 in 10 minutes
  - "Burned out" bandwidth
  - Could **not** have infected entire Internet in 15 minutes — too bandwidth intensive
- Can rapid worm do "better" than Slammer?...

A Possible Warhol Worm

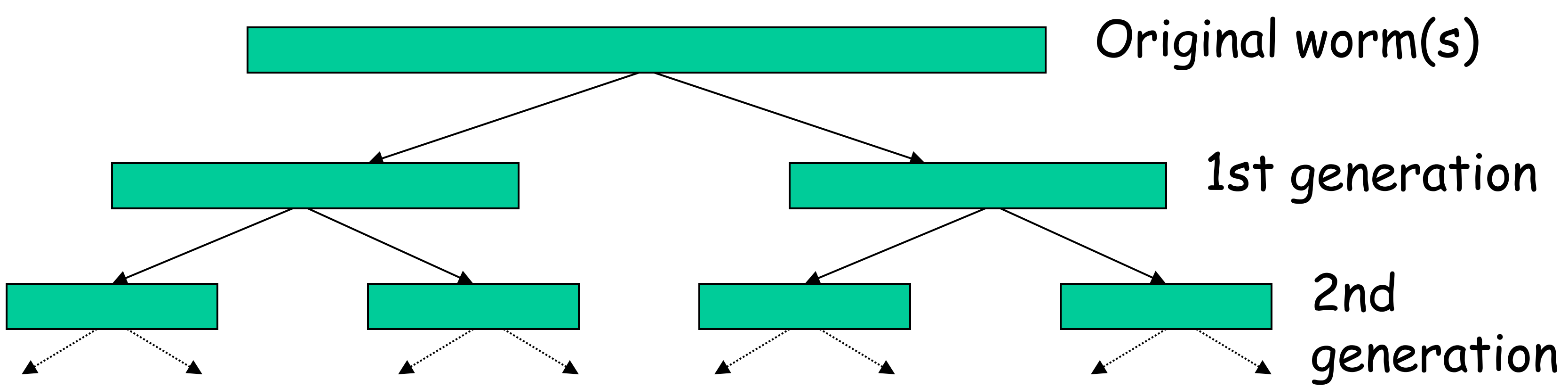
- Seed worm with an initial **hit list** containing a set of vulnerable IP addresses
  - Depends on the particular exploit
  - Tools exist for identifying vulnerable systems
- Each successful initial infection would attack selected part of IP address space
- Could infect entire Internet in 15 minutes!
- No worm this sophisticated has yet been seen in the wild (as of 2011)
  - Slammer generated random IP addresses

Flash Worm

- Can we do "better" than Warhol worm?
- Infect entire Internet in less than 15 minutes?
- Searching for vulnerable IP addresses is the slow part of any worm attack
- Searching might be bandwidth limited
  - Like Slammer
- **Flash worm** designed to infect entire Internet almost instantly

Flash Worm

- Predetermine **all** vulnerable IP addresses
  - Depends on details of the attack
- Embed these addresses in worm(s)
  - Results in huge worm(s)
  - But, the worm replicates, it splits
- No wasted time or bandwidth!



Flash Worm

- Estimated that ideal flash worm could infect the entire Internet in **15 seconds!**
  - Some debate as to actual time it would take
  - Estimates range from 2 seconds to 2 minutes
- In any case…
- …much faster than humans could respond
- So, any defense must be fully automated
- How to defend against such attacks?

Rapid Malware Defenses

- Master IDS watches over network
  - "Infection" proceeds on part of network
  - Determines whether an attack or not
  - If so, IDS saves most of the network
  - If not, only a slight delay
- Beneficial worm
  - Disinfect faster than the worm infects
- Other approaches?

Push vs Pull Malware

- Viruses/worms examples of “push”
- Recently, a lot of “pull” malware
- Scenario
  - A compromised web server
  - Visit a website at compromised server
  - Malware loaded on you machine
- Good paper: Ghost in the Browser

Botnet

- Botnet: a "network" of infected machines
- Infected machines are "bots"
  - Victim is unaware of infection (stealthy)
- **Botmaster controls botnet**
  - Generally, using **IRC (Internet Relay Chat protocol to manage their bots)***
  - P2P botnet architectures exist
- Botnets used for…
  - Spam, DoS attacks, keylogging, ID theft, etc.

* **!!! HOMEWORK:** Read about IRC and its utility for controlling a botnet. Also read how a covert channel can be useful for controlling a botnet.

Botnet Examples

- XtremBot
  - Similar bots: Agobot, Forbot, Phatbot
  - Highly modular, easily modified
  - Source code readily available (GPL license)
- UrXbot
  - Similar bots: SDBot, UrBot, Rbot
  - Less sophisticated than XtremBot type
- GT-Bots and mIRC-based bots
  - mIRC is common IRC client for Windows

More Botnet Examples

- Mariposa
  - Used to steal credit card info
  - Creator arrested in July 2010
- Conficker
  - Estimated 10M infected hosts (2009)
- Kraken
  - Largest as of 2008 (400,000 infections)
- Srizbi
  - For spam, one of largest as of 2008

Computer Infections

- Analogies are made between computer viruses/worms and biological diseases
- There are differences
  - Computer infections are much quicker
  - Ability to intervene in computer outbreak is more limited (vaccination?)
  - Bio disease models often not applicable
  - "Distance" almost meaningless on Internet
- But there are some similarities…

Computer Infections

- Cyber “diseases” vs biological diseases
- One similarity
  - In nature, too few susceptible individuals and disease will die out
  - In the Internet, too few susceptible systems and worm might fail to take hold
- One difference
  - In nature, diseases attack more-or-less at random
  - Cyber attackers select most “desirable” targets
  - Cyber attacks are more focused and damaging
- Mobile devices an interesting hybrid case

Future Malware Detection?

- Malware today far outnumbers "goodware"
  - Metamorphic copies of existing malware
  - Many virus toolkits available
  - Trudy can recycle old viruses, new signatures
- So, may be better to "detect" good code
  - If code not on approved list, assume it's bad
  - That is, use **whitelist** instead of **blacklist**

Example of Miscellaneous Software based Attacks

- Numerous attacks involve software
- We'll discuss a few issues that do not fit into previous categories
  - Salami attack
  - Linearization attack
  - Time bomb
  - Can you ever trust software?

Salami Attack

- What is Salami attack?
  - Programmer "slices off" small amounts of money
  - Slices are hard for victim to detect
- Example
  - Bank calculates interest on accounts
  - Programmer "slices off" any fraction of a cent and puts it in his own account
  - No customer notices missing partial cent
  - Bank may not notice any problem
  - Over time, programmer makes lots of money!

Salami Attack

- Such attacks are possible for insiders
- Do salami attacks actually occur?
  - Or is it just *Office Space* folklore?
- Programmer added a few cents to every employee payroll tax withholding
  - But money credited to programmer's tax
  - Programmer got a big tax refund!
- Rent-a-car franchise in Florida inflated gas tank capacity to overcharge customers

Salami Attacks

- Employee reprogrammed Taco Bell cash register: \$2.99 item registered as \$0.01
  - Employee pocketed \$2.98 on each such item
  - A large “slice” of salami!
- In LA, four men installed computer chip that overstated amount of gas pumped
  - Customers complained when they had to pay for more gas than tank could hold
  - Hard to detect since chip programmed to give correct amount when 5 or 10 gallons purchased
  - Inspector usually asked for 5 or 10 gallons

Linearization Attack

- Program checks for serial number S123N456
- For efficiency, check made one character at a time
- Can attacker take advantage of this?

```
#include <stdio.h>

int main(int argc, const char *argv[])
{
    int i;
    char serial[9]="S123N456\n";

    for(i = 0; i < 8; ++i)
    {
        if(argv[1][i] != serial[i]) break;
    }
    if(i == 8)
    {
        printf("\nSerial number is correct!\n\n");
    }
}
```

Linearization Attack

- Correct number takes longer than incorrect
- Trudy tries all 1st characters
  - Find that S takes longest
- Then she guesses all 2nd characters: S*
  - Finds S1 takes longest
- And so on…
- Trudy can recover one character at a time!
  - Same principle as used in lock picking

Linearization Attack

- What is the advantage to attacking serial number one character at a time?
- Suppose serial number is 8 characters and each has 128 possible values
  - Then $128^8 = 2^{56}$ possible serial numbers
  - Attacker would guess the serial number in about 2^{55} tries — a lot of work!
  - Using the linearization attack, the work is about 8 * (128/2) = 2^9 which is easy

Linearization Attack

- A real-world linearization attack
- TENEX (an ancient timeshare system)
  - Passwords checked one character at a time
  - Careful timing was ***not*** necessary, instead…
  - …could arrange for a "page fault" when next unknown character guessed correctly
  - Page fault register was user accessible
- Attack was very easy in practice

Time Bomb

- In 1986 Donald Gene Burleson told employer to stop withholding taxes from his paycheck
- His company refused
- He planned to sue his company
  - He used company time to prepare legal docs
  - Company found out and fired him
- Burleson had been working on malware…
  - After being fired, his software "time bomb" deleted important company data

Time Bomb

- Company was reluctant to pursue the case
- So Burleson sued company for back pay!
  - Then company finally sued Burleson
- In 1988 Burleson fined $11,800
  - Case took years to prosecute…
  - Cost company thousands of dollars…
  - Resulted in a slap on the wrist for attacker
- One of the first computer crime cases
- Many cases since follow a similar pattern
  - Companies reluctant to prosecute

Trusting Software

- Can you ever trust software?
  - See Reflections on Trusting Trust
- Consider the following thought experiment
- Suppose C compiler has a virus
  - When compiling login program, virus creates backdoor (account with known password)
  - When recompiling the C compiler, virus incorporates itself into new C compiler
- Difficult to get rid of this virus!

Trusting Software

- Suppose you notice something is wrong
- So you start over from scratch
- First, you recompile the C compiler
- Then you recompile the OS
  - Including login program…
  - You have not gotten rid of the problem!
- In the real world
  - Attackers try to hide viruses in virus scanner
  - Imagine damage that would be done by attack on virus signature updates