Operating Systems

23. Security

Paul Krzyzanowski

Rutgers University

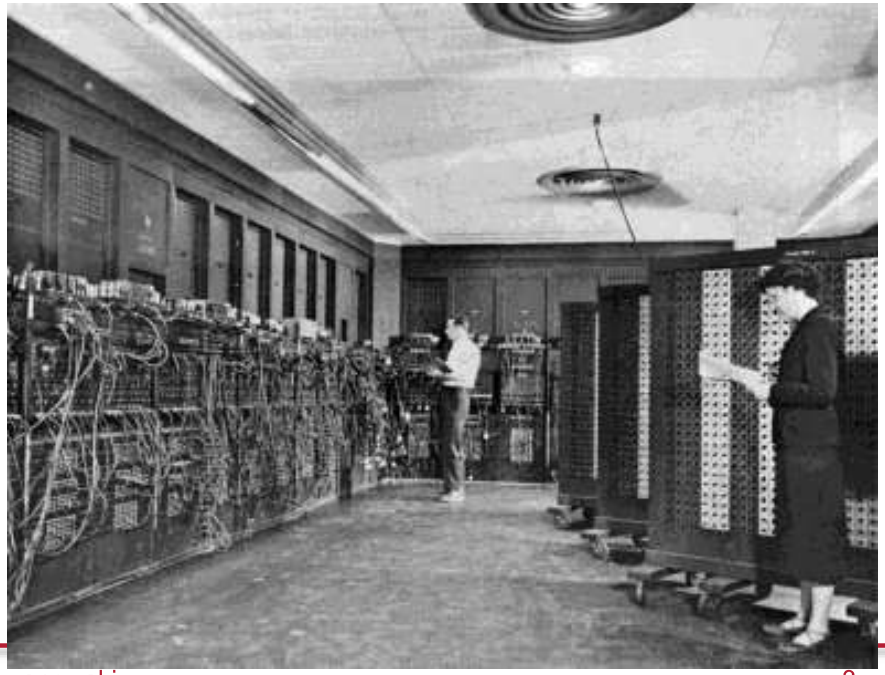
Spring 2015

Threats

Computer security… then

Issue from the dawn of computing:

- Colossus at Bletchley Park: breaking codes
- ENIAC at Moore School: ballistic firing tables
- single-user, single-process systems
- data security needed
- physical security



Computer security… now

- Sensitive data of different users lives on the same file servers
- Multiple processes on same machine
- Authentication and transactions over network
  - open for snooping
- We might want to run other people's code in our process space
  - Device drivers, media managers
  - Java applets, Flash code
  - Downloaded software
  - … not just from trusted organizations
    (also, do you trust a trusted organization?)

Systems are easier to attack

Automation

- Data gathering
- Mass mailings

Distance

- Attack from your own home

Sharing techniques

- Virus kits, virus obfuscation kits (*crypting* services)
- Hacking tools

Penetration

Guess a password

- system defaults, brute force, dictionary attack

Crack a password

- Online vs. offline
- Precomputed hashes (see rainbow tables)
  - Defense: Salt

Penetration: Guess/get a password

To access the Web-based Utility of the Router:

- Launch a web browser, such as Internet Explorer or Mozilla Firefox, and enter the Router's default IP address, **192.168.1.1**, in the *Address* field. Press the **Enter** key.
- A screen will appear asking you for your User name and Password. Enter **admin** in the *User Name* field, and enter your password (default password is **admin**) in the *Password* field. Then click the **OK** button.

Figure 6-1: Router's IP Address

Page 29 of the
Linksys Wireless-N Gigabit Security Router with VPN
user guide

Figure 6-2: Login Screen for Web-based Utility

Penetration

Social engineering

- people have a tendency to trust others
- identify corporate/school organizational structure
- facebook, twitter, blogs, personal home pages
- look through dumpsters for information
- impersonate a user
- Phishing: impersonate a company/service

Penetration

Trojan horse

- program masquerades as another
- Get the user to click on something, run something, enter data

```
All CS iLab and Graduate machine hostnames have changed
from XXX.rutgers.edu to XXX.cs.rutgers.edu.

_____________________________________________________________________________

Use your Rutgers University username and password to log into any
CS iLab and Graduate domain machine.

_____________________________________________________________________________

If you can not log in please verify that you have acknowledged the Department
of Computer Science Academic Integrity Policy at the following address.

You must acknowledge this policy once every academic calendar year.

http://www.cs.rutgers.edu/policies/academicintegrity/index.php?page=4

To confirm that you have submitted a record, log onto the above web
page.  A message stating "Acknowledgment received on ..." should
appear at the bottom.  If you see a message stating "To acknowledge
this policy, you must be logged in." you need to first log in.  If you
  see a message stating "You have no entry as of ...", we have no record
of your acknowledgement and you must submit a record to regain access
to your account.

_____________________________________________________________________________
pxk@ls.cs.rutgers.edu's password:
Permission denied, please try again.
pxk@ls.cs.rutgers.edu's password:
```

Phishing

Masqueraded e-mail

Subject: Attn: Web/E-mail Account Holder,
Date: April 20, 2014

Attn: Web/E-mail Account Holder,

This message is from the University Webmail Messaging Center to all email account owners.

We are currently carrying out scheduled maintenance,upgrade of our web mail service and we are changing our mail host server,as a result your original password will be reset.

We are sorry for any inconvenience caused.

To complete your webmail email account upgrade, you must reply to this email immediately and provide the information requested below.
****
CONFIRM YOUR EMAIL IDENTITY NOW
E-mail Address:
User Name/ID:
Password:
Re-type Password:
****
Failure to do this will immediately render your email address deactivated from the University Webmail.
****
This E-mail is confidential and privileged. If you are not the intended Recipient please accept our apologies; Please do not Disclose, Copy or Distribute Information in this E-mail or take any action in Reliance on its contents: to do so is strictly prohibited and may be Unlawful.

Please inform us that this Message has gone astray before deleting it.

Thank you for your Co-operation.

Malicious Files and Attachments

Take advantage of:

- Programs that automatically open attachments
- Interfaces that hide extensions yet use them to execute a program
  - trick the user

`love-letter.txt.vbs` *looks like* `love-letter.txt`

`resume.doc.scr` *looks like* `resume.doc`

Exploiting bugs

Exploit software bugs

- Most (all) software is buggy
- Big programs have lots of bugs
  - *sendmail*, *wu-ftp*
- some big programs are *setuid* programs
  - *lpr, uucp, sendmail, mount, mkdir, eject*

Common bugs

- buffer overflow
  (blindly read data into buffer)
  - e.g., *gets*
- back doors and undocumented options

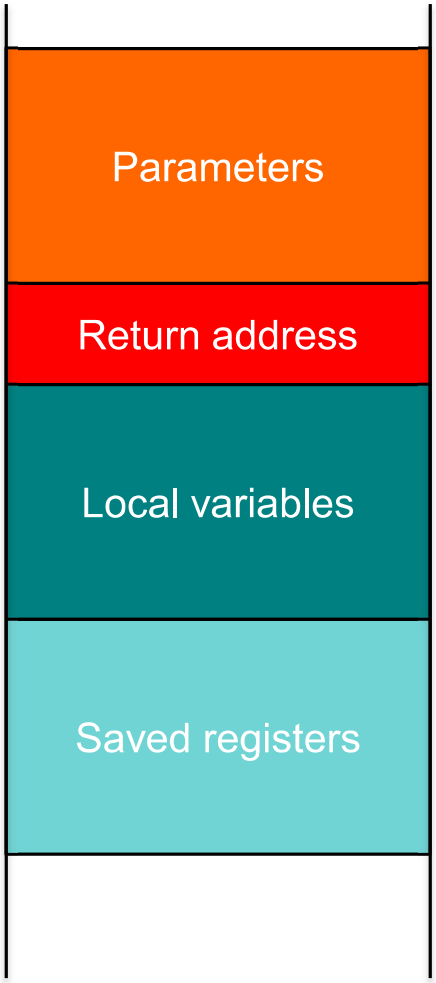
The classic buffer overflow bug

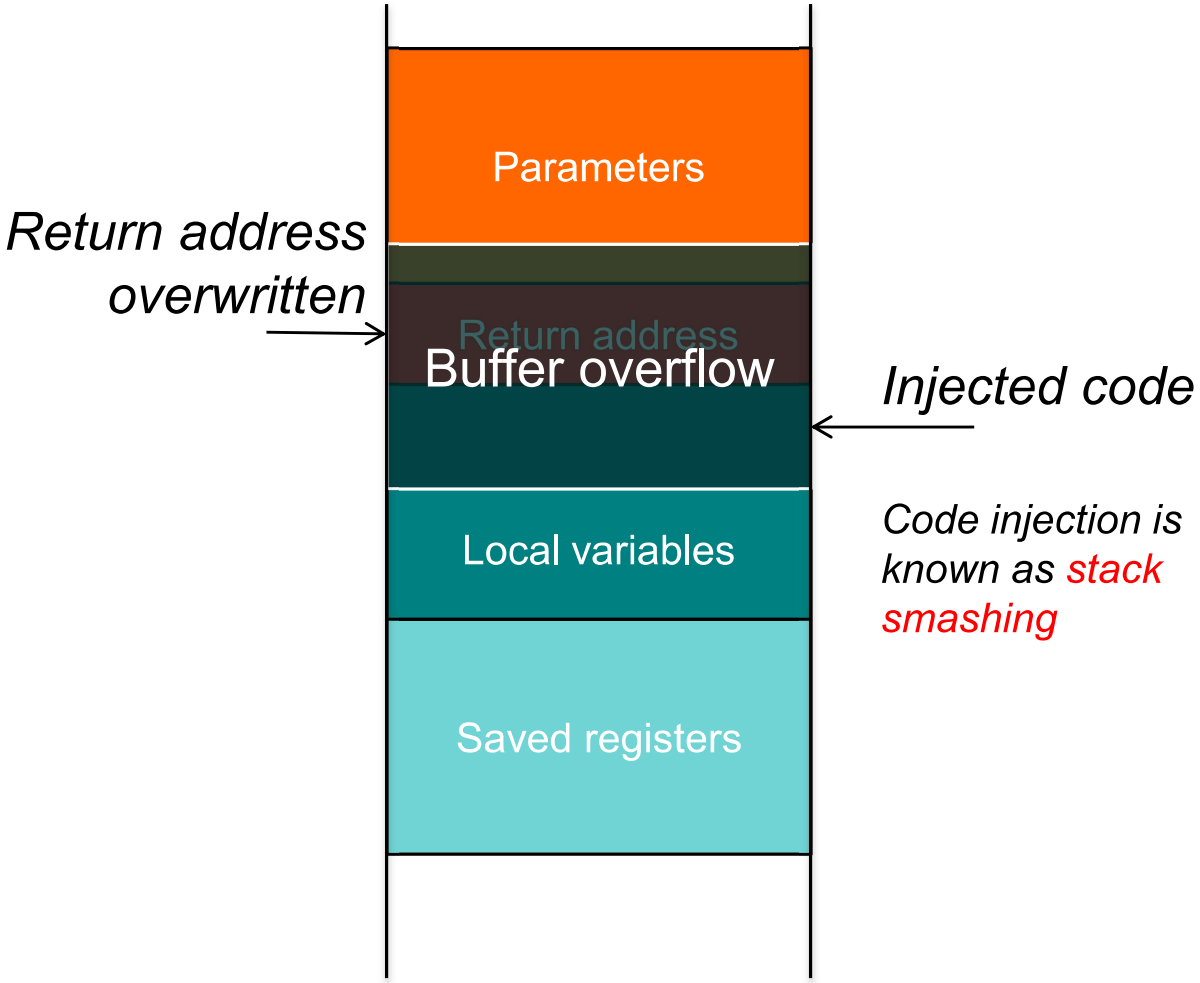
gets.c from OS X: © 1990,1992 The Regents of the University of California.

```
gets(buf)
char *buf;
   register char *s;
   static int warned;
   static char w[] = "warning: this program uses gets(), which is unsafe.\r\n";

   if (!warned) {
      (void) write(STDERR_FILENO, w, sizeof(w) - 1);
      warned = 1;
   }
   for (s = buf; (c = getchar()) != '\n';)
      if (c == EOF)
           if (s == buf)
                return (NULL);
           else
                break;
      else
           *s++ = c;
   *s = 0;
   return (buf);
}
```

Buffer overflow





More data was input than the programmer expected, causing the local array that was allocated for the data to overflow. The overflow overwrites the return address on the stack. Now, when the function returns, the return address is under the control of the attacker.

Dealing with buffer overflows: No Execute

- Executable space protection
  - Disallow code execution on the stack or heap
  - Set MMU per-page execute permissions to no-execute
  - Intel and AMD added this support in 2004

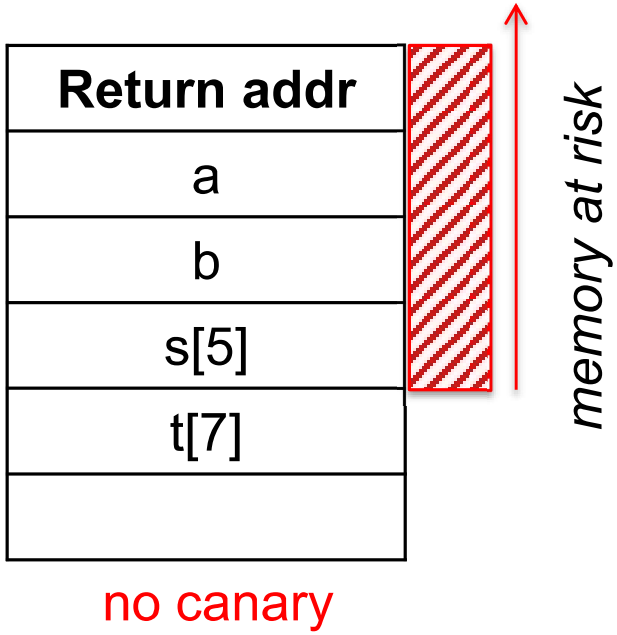
  - Examples
    - Microsoft DEP (Data Execution Prevention) (since XP SP2)
    - Linux PaX patches
    - OS X ≥10.5

Dealing with buffer overflows: Canaries

- Stack canaries
  - Place a random integer before the return address on the stack
  - Before a return, check that the integer is there and not overwritten: a buffer overflow attack will likely overwrite it

```
int a, b=999;
char s[5], t[7];

gets(s);
```

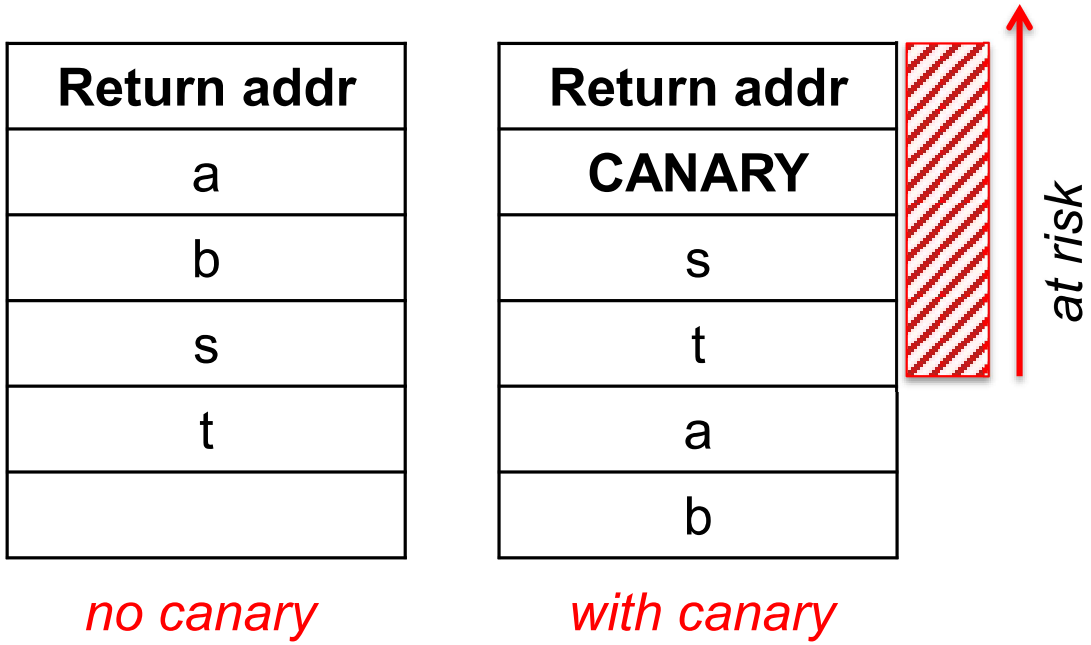


Dealing with buffer overflows: Canaries

- Stack canaries
  - Place a random integer before the return address on the stack
  - Before a return, check that the integer is there and not overwritten: a buffer overflow attack will likely overwrite it
  - Allocate arrays into higher memory in the stack so they won't clobber other automatic (stack-based) variables

```
int a, b=999;
char s[5], t[7];

gets(s);
```



Virus

- Does not run as a self-contained process
- Code is attached onto another program or script

- File infector
  - primarily a problem on systems without adequate protection mechanisms
- Boot-sector
- Macro (most common form: visual basic scripting)
- Hypervisor
  - intercept traps and privileged instructions from OS

Virus scanning

- Search for a "signature"
  - Bunch of bytes present in a virus that (we hope!) is unique to the virus and not any legitimate code
  - *NOT a cryptographic signature!*
- Some viruses are encrypted
  - Signature is either the code that does the decryption or the scanner must be smart enough to decrypt the virus
  - Crypting service: obfuscates malware to escape virus detectors
- Some viruses mutate to change their code every time they infect another system
  - Run the code through an emulator to detect the mutation

Key loggers

- Record every keystroke
- Windows *hook* mechanism
  - Procedure to intercept message traffic before it reaches a target windows procedure
  - Can be chained
  - Installed via *SetWindowsHookEx*
  - WH_KEYBOARD and WH_MOUSE
    - Capture key up, down events and mouse events
- Hardware loggers

Rootkits

- Replacement commands (or standard shared libraries or OS components) to hide the presence of an intruder
  - *ps, ls, who, netstat, …*
- Hide the presence of a user or additional software (backdoors, key loggers, sniffers)
- Now the OS can no longer be trusted!
- Examples
  - Lenovo Superfish adware (2014)
    - Preinstalled self-signed root certificate
    - Allows anyone on your network to silently intercept HTTPS communications
  - Sony BMG DRM rootkit (October 2005)
    - Creates hidden directory; installs several of its own device drivers; reroutes Windows system calls to its own routines
    - Intercepts kernel-level APIs and disguises its presence with cloaking (hides sys files)
  - Carrier IQ (December 2011)
    - Software for cell phone analytics – designed to be undetectable
    - Installed on Sprint, HTC, Apple (iPhone ≤4), Samsung, BlackBerry, ….