

# Burglar Alarms for Detecting Intrusions

**Marcus J. Ranum**  
**[mjr@nfr.net](mailto:mjr@nfr.net)**

# Disclaimer

- Originally I expected to provide a bunch of source code with this talk
  - Unfortunately, UNIX isn't very portable anymore
  - Unfortunately, Marcus isn't allowed to touch a keyboard very much anymore
- I don't have any good examples for NT

# Burglar Alarms

- A burglar alarm is a misuse detection system that is carefully targeted
  - You may not care about people port-scanning your firewall from the outside
  - You may care profoundly about people port-scanning your mainframe from the inside
  - Set up a misuse detector to watch for misuses violating site policy

# Burglar Alarms *(cont)*

- Goals:
  - Based on site policy alert administrator to policy violations
  - Detect events that may not be “security” events which may indicate a policy violation
    - New routers
    - New subnets
    - New web servers

# Burglar Alarms *(cont)*

- The ideal burglar alarm will be situated so that it fires when an attacker performs an action that they normally would try once they have successfully broken in
  - Adding a userid
  - Zapping a log file
  - Making a program setuid root

# Burglar Alarms *(cont)*

- Burglar alarms are a big win for the network manager:
  - Leverage local knowledge of the local network layout
  - Leverage knowledge of commonly used attacker tricks
  - Detect successful attacks by detecting second-order effects of a break-in

# Burglar Alarms: Pro

- Reliable
- Predictable
- Easy to implement
- Easy to understand
- Generate next to no false positives
- May detect previously unknown attacks by second-order effects

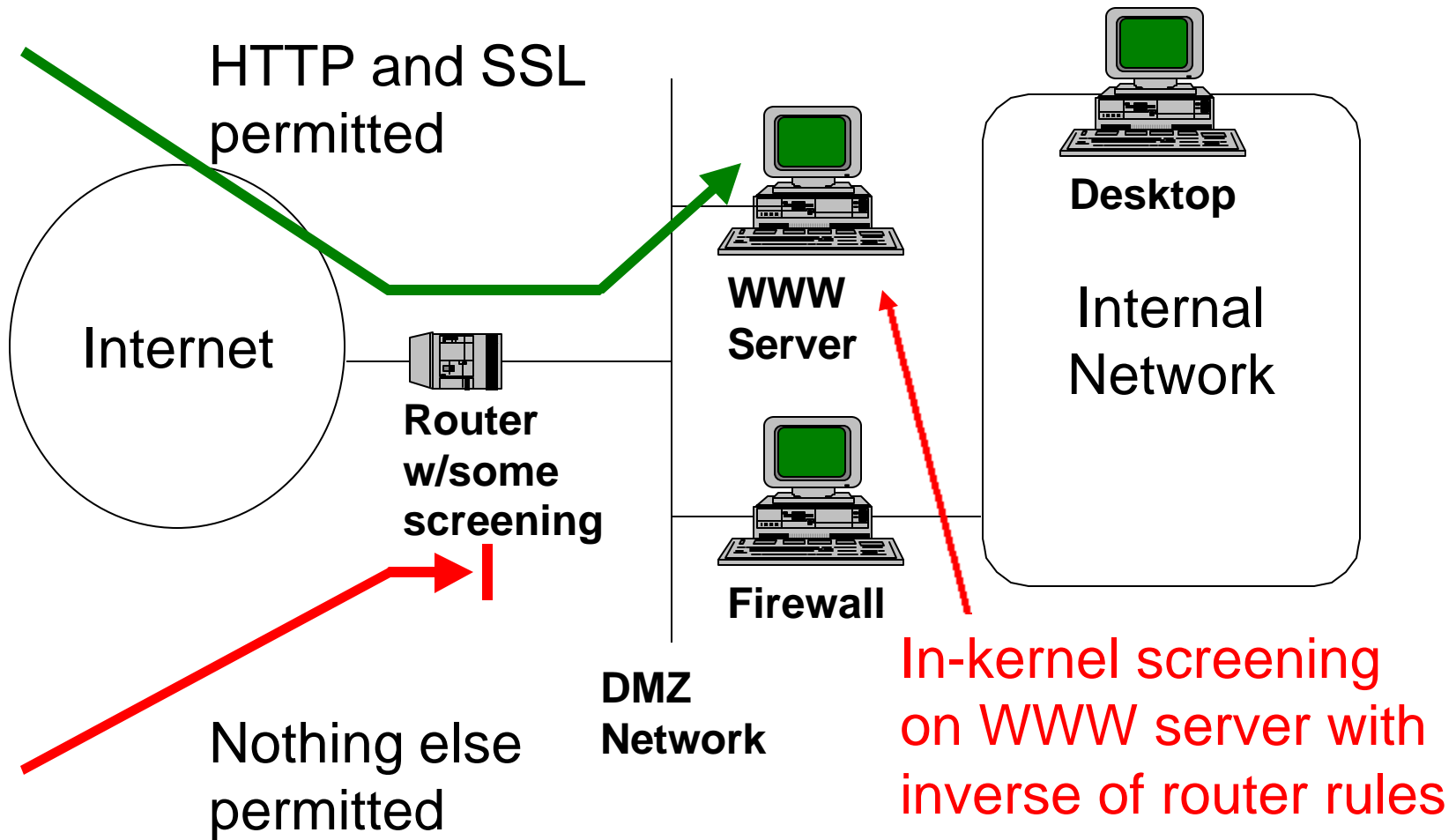
# Burglar Alarms: Con

- Policy-directed
  - Requires knowledge about your network
  - Requires a certain amount of stability within your network
- Requires care not to trigger them yourself

# The Right Thing to Do

- Application writers need to get smarter about not only avoiding errors but logging cases where an unusual condition happens
  - I.e.: don't just truncate huge command lines - truncate them and log a warning that it happened
  - Fix ***and*** detect/notify about buffer overruns!

# Simple Burglar Alarm



# Simple Burglar Alarm *(cont)*

- In-kernel screening can be used to generate alerts easily
- Example is based on ip\_filt screening language
  - Ip\_filt can log packet bodies or events
  - Logs can be post-processed/watched with a simple perl script
  - Remember: this should never happen

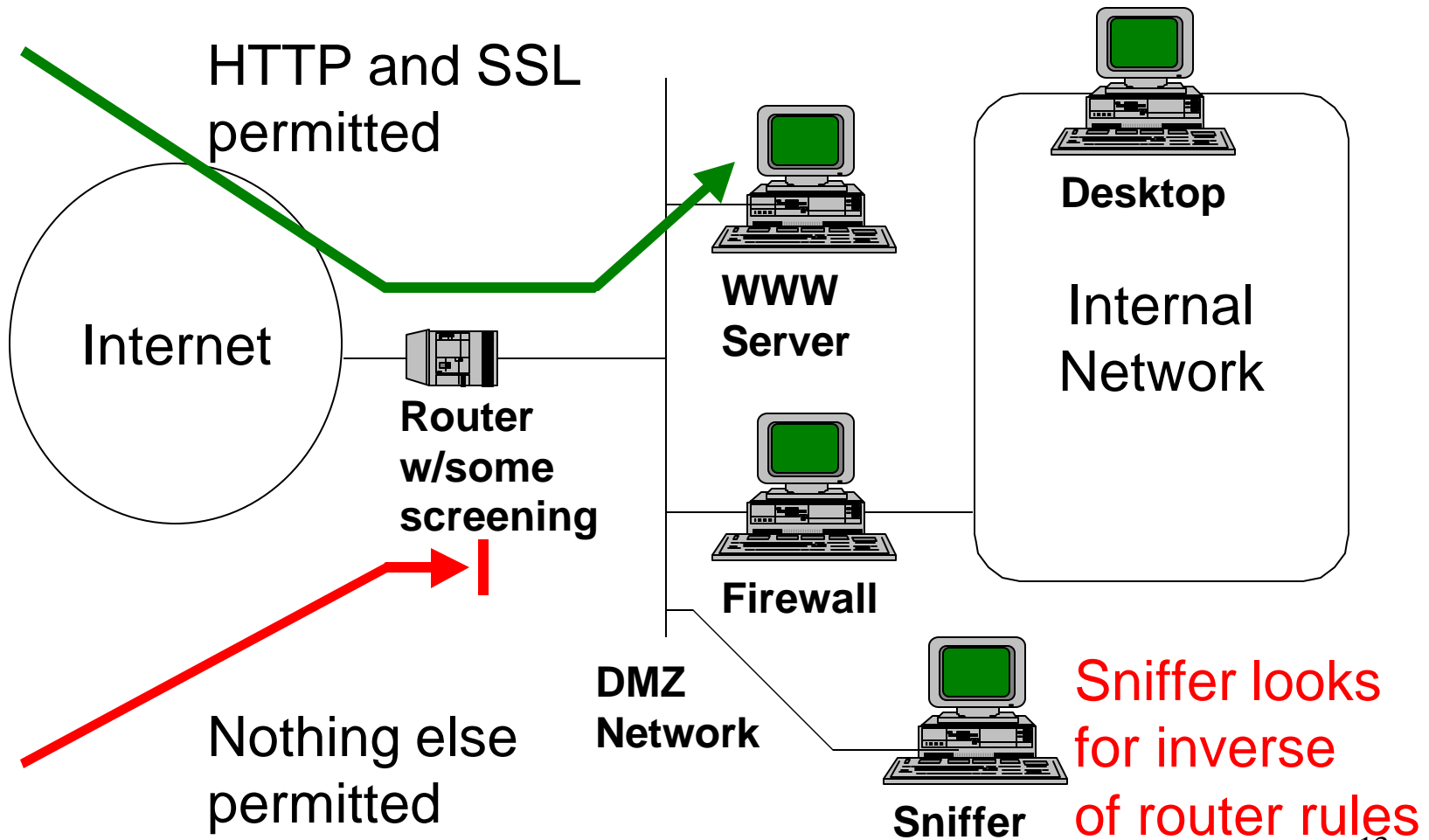
# Simple Burglar Alarm *(cont)*

```
# sample: block all packets by default
block all
```

```
# for example we're assuming outside interface is le0
# drop "localhost" packets coming in from network
block in on le0 log body from localhost to any
```

```
# drop source routed packets
block in quick log body all with opt lsrr
block in quick log body all with opt ssrr
```

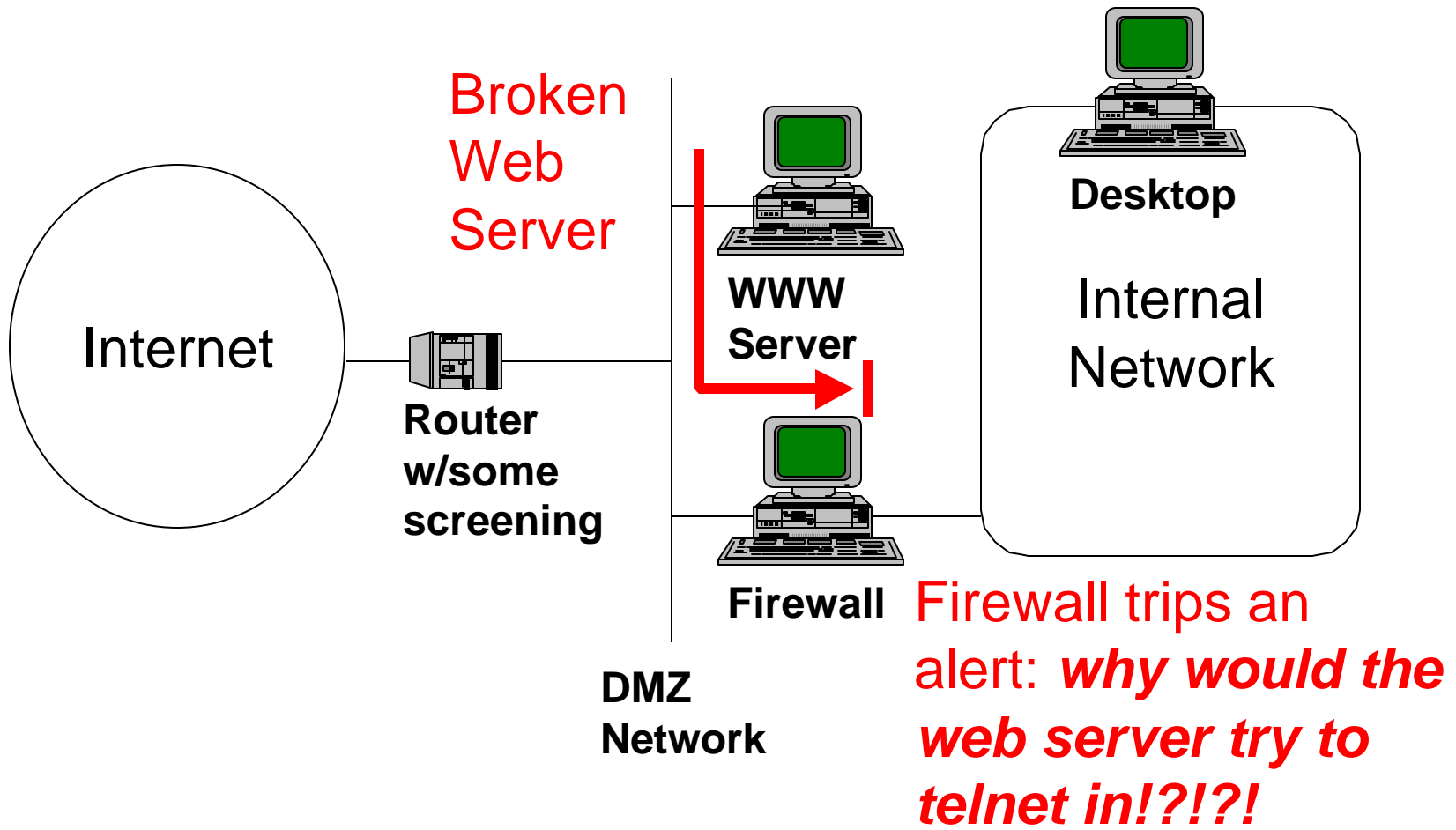
# Simple Burglar Alarm: 2



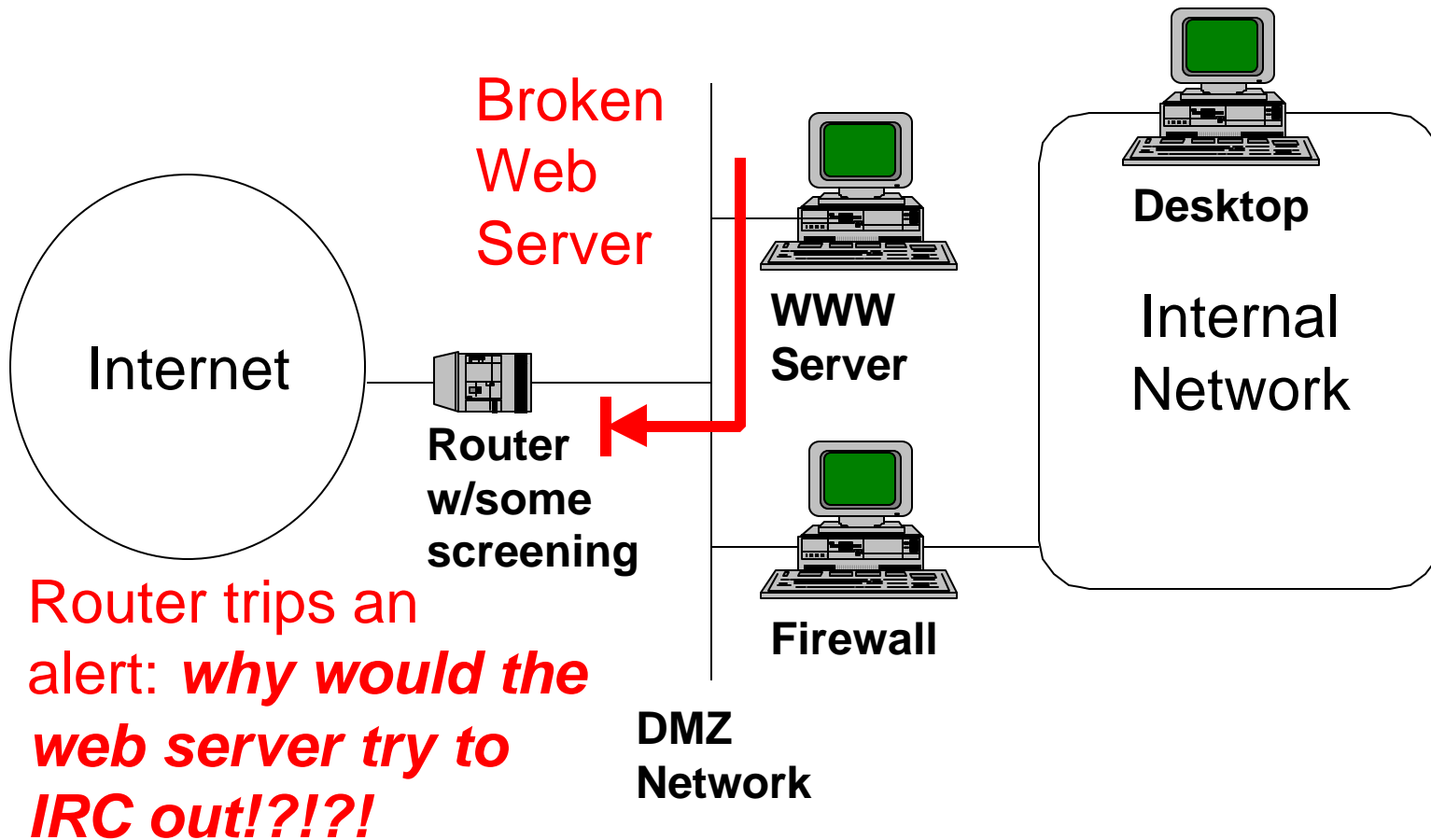
# IDS and firewalls

- Firewalls allow “overlapping” rulesets with different priorities
  - Many firewalls can trigger alerts when traffic to “bad destination” is seen
  - Use this capability to build burglar alarms

# IDS Firewall Alarm



# IDS Firewall Alarm 2



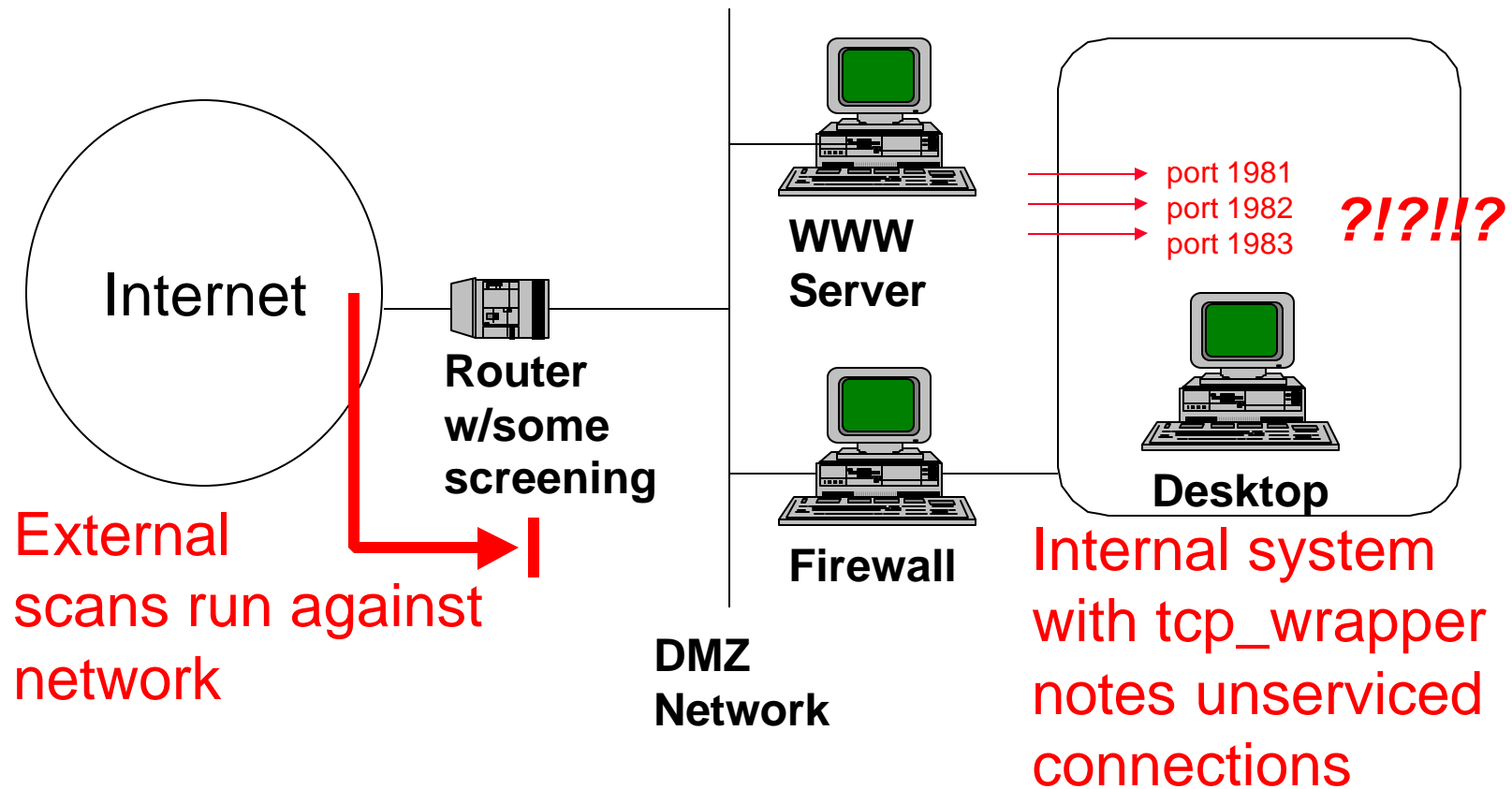
# Building: Burglar alarms

- Burglar alarms are best built using:
  - Sniffers
  - In-kernel packet screens (ip\_filt, ipfilter)
  - Application packet sniffers (tcpdump, NFR, Argus, tcpwatch)
  - Application logs (tcpwrapper, VPN server logs, kernel logs, syslogs)

# Building a Scan Alarm

- Example:
  - Suppose we have router screening in place using “established” keyword
  - Then we should not get connects on certain ports through the firewall router
  - Set up tcp\_wrapper on various port ranges
    - Log occurrence of connections
    - When threshold goes up trigger an alarm

# A Scan Alarm



# Building a Scan Alarm *(cont)*

- Tcp\_wrapper /etc/hosts.deny:

```
bugport9: ALL: (/etc/safe_finger @%h|\n    /usr/ucb/mail -s %d-%h root) &
```

```
bugport10: ALL: (/etc/safe_finger @%h|\n    /usr/ucb/mail -s %d-%h root) &
```

# Building a Scan Alarm *(cont)*

- `/etc/services:`

```
#this line names a service by port
#to watch these ports with tcp_wrapper
bugport9          9/tcp
bugport10         10/tcp
```

# Trapping Actions

- Modifying the top half of system call interfaces used to be easy
  - It's still pretty easy on open source platforms
  - Can be bypassed but it's nice to know who/when/where instances of `connect`, `accept`, `setuid`, `exec` were used

# Trapping Actions *(cont)*

- Wrapping the kernel top half of a system call can be done by changing the syscall jump table
  - Code such as exec wrappers from TIS labs does this
  - Memco's SEOS does this
- Some systems may permit this in a loadable module

# Trapping Actions *(cont)*

- Consider modifying shell(s) to log command lines passed with `-c` that are not attached to a terminal (or even then)

# Chroot-a-nono

- A process that is already chrooted probably should not chroot again
  - If kernel source is available this is easy to do 😊 (vfs\_syscalls.c)
  - Check within chroot system call for root inode != real root and log alarm

```
/* new! */  
if (fdp->fd_rdir != NULL)  
    log(LOG_ERR, "WARNING! chroot when already chrooted!");
```

# ls-o-matic

- Train yourself not to run “ls” as root
- Replace “ls” with a program that mails you or shuts the system down if it is ever run as root
- Use “echo \*” instead of “ls”

... This trick takes a lot of discipline!

# Shared-Library boobytrap

- Systems with shared libraries are a great place to add alarms
- Generate a custom version of the `exec()` library family that logs every command execution that isn't one of a small expected set
  - Good for firewalls or web servers!

# Nit-pick

- Many times when a break-in occurs attackers will set up a sniffer
- If NIT device is not configured they often add it
- Replace NIT device with something that triggers a warning instead
  - /dev/nit or bpf driver can be replaced with a driver that halts the system

# File-change-o

- Very simple cron job can be made to
  - Copy critical files to a hidden directory
    - /etc/passwd, /etc/group, /etc/inetd.conf
    - `find / -user root -print`
  - Diff the files against what's currently installed on the system
    - Bring differences to the administrators' attention
  - Automating tripwire works better for this

# File shrinkener

- Write a program to check if the inode number of `/var/log/messages` has not changed at the same time the file has shrunk
  - Use `ls -i`, and `ls -l` in a shell script
  - Use `stat` in C code
- Embed this within something innocuous (how about `cron`, `sendmail`, or `named`?)

# Stupid Hacker Tricks

- In your FTP area, make a symbolic link:

```
ln -s . stupid_hacker_tricks
```

- Poorly coded software that recursively scans directories may have trouble with it

# Terrify Suzy\*

- May make people think twice about what kind of monitoring is going on in the system

```
# cat > main.c
main()
{
while(1) sleep(30);
} ^D
# cc -o watchdog main.c
# nohup watchdog&
```

\* based on an old story from Boyd Roberts

# Fake Hacktools

- Install fake hack tools
  - Backofficer friendly: pretends to be a back orifice server
    - I want to develop lots more like this; they are tremendously educational and amusing
  - an eggdrop or FSP server that logs everything
  - Stuffing back stack smashes to the remote user is considered unsporting

# Roto-Router

- Redirect incoming traceroute queries to a user-mode process which responds with carefully crafted packets
  - Looks like you go into the network
    - Then to microsoft.com
      - Then to whitehouse.gov
        - Then to playboy.com
        - etc.
  - Louis Mamakos (I think) invented this one

# Scan Slower

- Set up services on a port, that listen and accept connections
  - Set keepalive
  - Never send data
- This could be very nicely implemented in a border device that simulates an entire network or system

# Phat Warez

- Compress a few gigabytes of zeros into a .zip file (it'll get pretty small!)
  - Leave it in your Warez directory

# Redirector

- Set up something (kind of like a dynamic LocalDirector or a firewall with proxy transparency) on the border of your network that takes traffic destined to certain machines
  - Rewrites the destination to be the source
  - Sends it back out
  - “Wow! He’s scanning me back really quickly! He knows all my tricks!”

# Socket Stuffer

- For scanning tools that collect data off the ports and record/parse/log it
  - Have a listener on many man ports
  - Each listener, if connected to, sends back a few USENET postings from talk.bizarre
  - This would be lots of fun against the auditors who like to run ISS scans against you and charge you big \$\$ for the result

# Auditor Biter

- One nice way of catching clueless auditors who send an intern to run ISS against you and charge you big \$\$\$ is to create fake vulnerabilities in your system and wait to see if they appear in the report
  - Measure how much deviance exists between the report and the ISS output

# Rat Poison Files

- Collect a string (a single encrypted password) that is in your shadow password file / customer database / credit card database
  - Have a sniffer watching your system that will scream as soon as it sees that string leave the system

# Noset Executable

- For dedicated service machines, consider removing the ability to set the execute bit in multiuser mode
  - Must also be attached to a terminal
    - *Log whenever it isn't!!!*
  - Log and alert attempts to set execute permission

# No Exec Stack

- Several versions of UNIX (Solaris, some \*BSD variants) can now block attempts to execute code from within the stack
  - Makes buffer overruns a bit harder to implement for attacker
  - Doesn't prevent code to call existing functions -- not a perfect solution

# Fake Holes

- Install a phf.pl script in your CGI directory on your web server
  - Have it generate an alert
  - Make sure your script doesn't have its own holes!

# DumDum Users

- Have a user with a crackable but not obvious password
  - Put something in their .login to alert you when they log in
- If they ever log in, you know someone has gotten hold of your password file
- If they get E-mail you've been sold to spammers

# Summary

- Burglar alarms are a neat idea
- They work
- Some of these examples are simplistic and lame
  - If you have a sick imagination you should be able to do much better than these
- Exploit the home turf advantage