RIoT – Raiding Internet of Things

Jacob Holcomb, Security Analyst @ ISE

Speaker Information

- **Who?** Jacob Holcomb
  - » I specialize in application and network security, and exploit research and development.
  - » I have responsibly disclosed ~100 vulnerabilities since 2012.
- **What?** Security Analyst @ ISE
- **Why?** Security is **IMPORTANT!**

About ISE

- **We are:**
  - Ethical Hackers
  - Computer Scientists
  - Individuals

- **Our Customers are:**
  - Anyone in need of protecting important assets
- **Our perspective is:**
  - Whitebox (vs. Blackbox)

Why IoT Security is Important

- **100%** of IoT systems evaluated were vulnerable to exploitation.
- Routers and storage systems are not the only embedded devices with egregious deficiencies.
- These systems **CAN** and **ARE** being mass exploited.
- Everyday devices are becoming connected and transmitting data.

“Gartner Says the Internet of Things Installed Base Will Grow to 26 Billion Units By 2020”

*Source: https://www.gartner.com/newsroom/id/2636073

ISE IoT Research

Independent Security Evaluators

- Exploiting SOHO Routers
- Exploiting SOHO Router Services
- Exploiting N.A.S.ty Systems (Network Storage Exploitation)
- Hospital Systems and Infrastrucutre
- **https://securityevaluators.com/knowledge/**

#SOHOpelessly Broken

HACK ROUTERS AND GET PAID

https://sohopelesslybroken.com

DEFCON, DerbyCon, BSIDES, ToorCon

Topics

- Inherent Risks of Networking Equipment
- Threat Modeling and Testing Methodology
- Remediation and Security Design Principles
- IoT Integration

Security Risks

- **Large** attack surface
- Insecure by default
- Assumption of security on the (wireless) LAN
- Poor security design and implementation

Threat Modeling

- Threat Modeling allows engineers to identify and remediate software issues in early development stages. Cost-effective
- Thorough understanding of system and assets
- Should be a core component of any SDLC process

Threat Modeling Empowers You

- Define Project Scope
- System Modeling
- Identify Assets
- Identify Threats
- Identify Vulnerabilities
- Quantify Risk
- Security Response

Testing Methodology

- Information Gathering
- Scanning and Enumeration
- Gaining Access
- Maintaining Access

Information Gathering

- **Administration Settings**
  - Default credentials
  - Management interface(s)
- **WLAN/LAN Settings**
  - SSID and wireless encryption
  - Layer 3 Subnet (IP Range)
- **Network Service Settings**
  - DHCP, DNS, SNMP, UPnP, SMB, FTP, etc.

Scanning and Enumeration

- Identifying active hosts
- Identifying open TCP/UDP ports
- Identifying running services and versions

Scanning and Enumeration Cont.

```
root@Hak42:/# nmap -sS -Pn -sV -p T:1-65535 192.168.1.1

Starting Nmap 6.25 ( http://nmap.org ) at 2013-07-28 18:25 EDT
Nmap scan report for Wireless_Broadband_Router.InfoSec42 (192.168.1.1)
Host is up (0.0053s latency).
Not shown: 65524 closed ports
PORT     STATE SERVICE        VERSION
23/tcp   open  tcpwrapped
80/tcp   open  http           Verizon FIOS Actiontec http config
234/tcp  open  tcpwrapped
443/tcp  open  ssl/http       Verizon FIOS Actiontec http config
992/tcp  open  ssl/tcpwrapped
2555/tcp open  unknown
2556/tcp open  unknown
4567/tcp open  http           Actiontec TR069 remote access
8023/tcp open  tcpwrapped
8080/tcp open  http           Verizon FIOS Actiontec http config
8443/tcp open  ssl/http       Verizon FIOS Actiontec http config
```

Port Scan

TCP: nmap –sS –Pn –sV –p T:1-65535 X.X.X.X

UDP: nmap –sU –Pn –p U:1-65535 X.X.X.X

Banner Grab

Netcat: nc –nv <X.X.X.X> <port>

```
root@Hak42:/# nc -nv 192.168.1.1 8080
(UNKNOWN) [192.168.1.1] 8080 (http-alt) open
GET / HTTP/1.1

HTTP/1.1 200 OK
Content-Type: text/html
Set-Cookie: rg_cookie_session_id=1476875494; path=/;
Cache-Control: no-cache,no-store
Pragma: no-cache
Expires: Sun, 28 Jul 2013 22:33:39 GMT
Date: Sun, 28 Jul 2013 22:33:39 GMT
Accept-Ranges: bytes
Connection: close

<!--- Page(9074)=[Login] ---><HTML><HEAD><META HTTP-E
TENT="NO-CACHE"><META HTTP-EQUIV="PRAGMA" CONTENT="NO
ground-image: url('images/gradientstrip.gif'); backgr
TD, INPUT, OPTION, SELECT {font-size: 11px}
```

Gaining Access

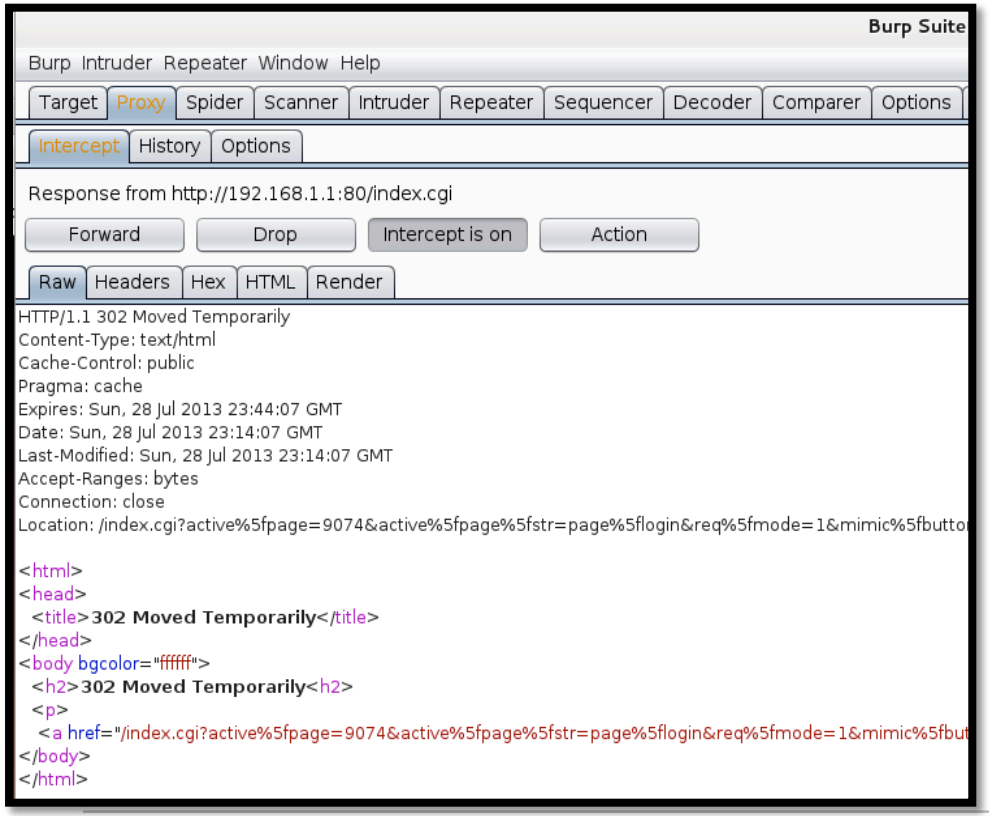
- **Service Investigation**
  - Analyze web applications
  - Analyze servers (e.g., FTP, SMTP, SMB, HTTP)
  - Source Code Review (Static Code Analysis)
  - Fuzz Network Services (Dynamic Analysis)

Analyzing Web Applications

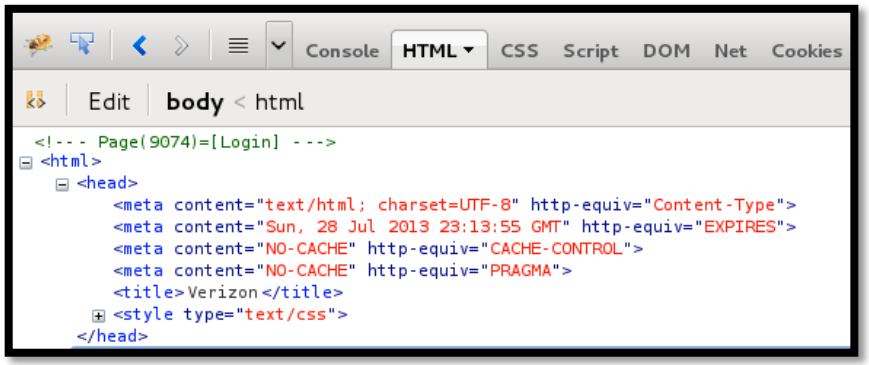
- **Understand the application**
  - Programming languages used
    - Server side
    - Client side
  - Protocols and APIs used (e.g., SOAP, REST)
  - Internet Media Type/MIME (e.g., JavaScript, HTML)
- **Tools**
  - Web proxy (e.g., Burpsuite)
  - Firebug (JavaScript debugger, HTML inspection)
  - Web Crawler
  - Python!

Analyzing Web Applications Cont.

Burpsuite



Firebug



Analyzing Network Servers

- **Authentication**
  - Type (e.g., Password, Key Pair)
  - Anonymous access/Weak or no credentials
  - Misconfigurations (e.g., Directory listing, permissions)
- **Protocol**
  - Standard? Custom? Can you speak it?
- **Encryption**
  - SSL/TLS?
  - SSH (AES, 3DES)?

Static Code Analysis

- **If source code is available, GET IT!**

- **Things to look for:**
  - Logic flaws (e.g., authentication, authorization)
  - Functions not performing bounds-checking
  - Backdoors

Static Code – Vulnerable Code

```
char ttybuf[16], buf[256];
FILE *ppp_fp;
int i;

        system("mkdir -p /tmp/ppp");
        sprintf(buf, "echo '%s * %s *'>/tmp/ppp/pap-secrets", nvram_safe_get("wan_pptp_username"), nvram_safe_get("wan_pptp_passwd"));
        system(buf);
        sprintf(buf, "echo '%s * %s *'>/tmp/ppp/chap-secrets", nvram_safe_get("wan_pptp_username"), nvram_safe_get("wan_pptp_passwd"));
        system(buf);
```

overflow

cmd inject

Static Code – More Vulnerable Code

```
int ej_apps_action(int eid, webs_t wp, int argc, char **argv){
    char *apps_action = websGetVar(wp, "apps_action", "");
    char *apps_name = websGetVar(wp, "apps_name", "");
    char *apps_flag = websGetVar(wp, "apps_flag", "");
    char command[128];

    if(strlen(apps_action) <= 0)
        return 0;

    nvram_set("apps_state_action", apps_action);

    memset(command, 0, sizeof(command));

    if(!strcmp(apps_action, "install")){
        if(strlen(apps_name) <= 0 || strlen(apps_flag) <= 0)
            return 0;

        sprintf(command, "start_apps_install %s %s", apps_name, apps_flag);
```

overflow

*Code from the ASUS RT-N56U

Fuzzing (Dynamic Analysis)

- **What happens if peculiar input is introduced?**
  - A{-G42!BBB}}}}}}/\/\/}}}}}}+=-_-1234d`~~((.)_(.))$
  - BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
- **Fuzzers**
  - **SPIKE:** generic_send_tcp X.X.X.X 21 ftp.spk 0 0
  - **BED:** ./bed.pl -s HTTP -t X.X.X.X -p 80
  - **Peach Framework**
  - **Metasploit Framework**
  - **Python!**

Analyze Fuzzing Results

- **Toolz**
  - Debugger (i.e., GDB)
  - System Call Tracer (i.e., strace)

```
gdb) i r
         zero       at       v0       v1       a0       a1       a2       a3
R0   00000000 00000000 00000000 1dcd0000 7fff69c0 00000000 00000000 00000000
           t0       t1       t2       t3       t4       t5       t6       t7
R8   00000000 0000fc00 00000000 802de000 00000000 00000004 7f82ed18 00000000
           s0       s1       s2       s3       s4       s5       s6       s7
R16  42424242 42424242 42424242 42424242 42424242 00425008 7fff6c50 00410000
           t8       t9       k0       k1       gp       sp       s8       ra
R24  00000000 7fff6b50 00000000 00000000 42424242 7fff6b60 00410000 7fff6b58
       status       lo       hi badvaddr    cause       pc
     0100fc13 02625a00 00000000 2ab59358 00000024 7fff6b64
         fcsr      fir      hi1      lo1      hi2      lo2      hi3      lo3
     00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
       dspctl  restart
     00000000 00000000
gdb) x/21i $sp
  0x7fff6b60:  andi    at,k1,0x4132
=> 0x7fff6b64:  lui     t0,0x6e6c
```

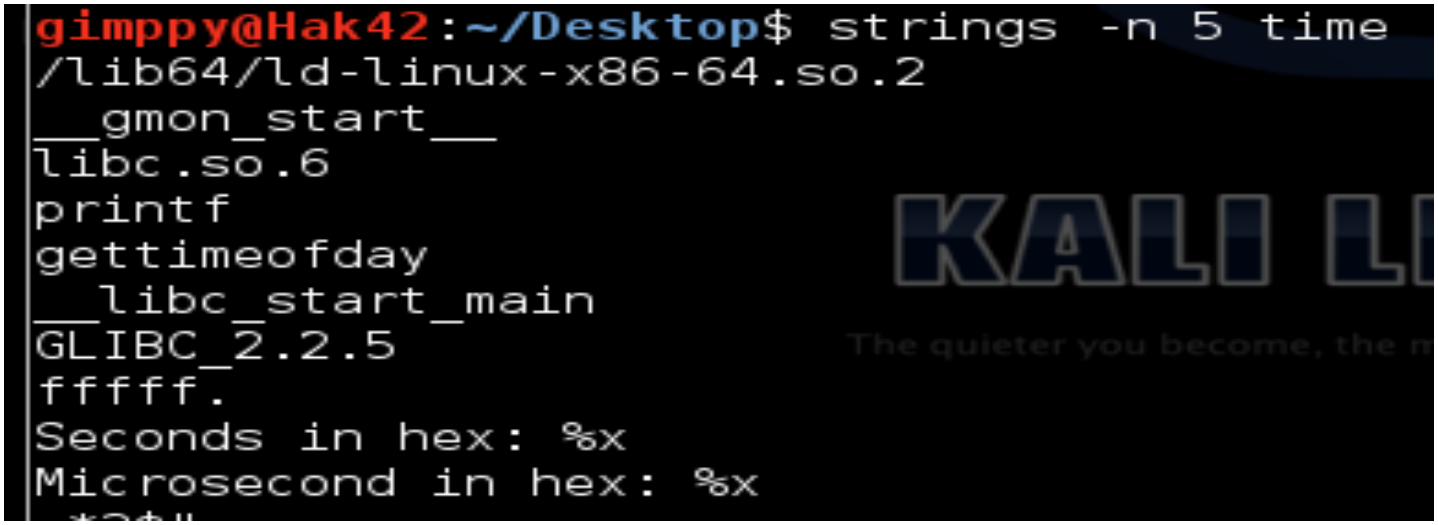
*Debugging ASUS RT-AC66U mips exploit

Gaining Access

- **Reverse Engineering**
  - System Binaries
- **Simple RE Toolz and Techniques**
  - Strings
  - Hexdump
  - Grep
  - GDB, IDA
  - Quemu
  - Open source? Perform static analysis!
- **Exploit Development**

Reverse Engineering Toolz and Techniques

- **Strings:** strings –n <INT> <FILE>



Reverse Engineering Toolz and Techniques

- **Grep:** grep –R <string> *

```
irmware$ grep -R backdoor *
DRU_v1.0.8.0/src/router/mipsel-uclibc/install/httpd/usr/sbin/httpd matches
/src/router/shared/broadcom.c://Tom.Hung 2012-6-27, Add backdoor feature
/src/router/shared/broadcom.c:static int backdoor(webs_t wp, char_t *urlPrefix, char_t *webDir, int arg,
/src/router/shared/broadcom.c:static void do_backdoor_asp(char *url, FILE *stream)
/src/router/shared/broadcom.c:      backdoor(stream, NULL, NULL, 0, url, path, query);
/src/router/shared/broadcom.c:    { "backdoor*", "text/html", no_cache, NULL, do_backdoor_asp, do_auth },
```

?!?!?

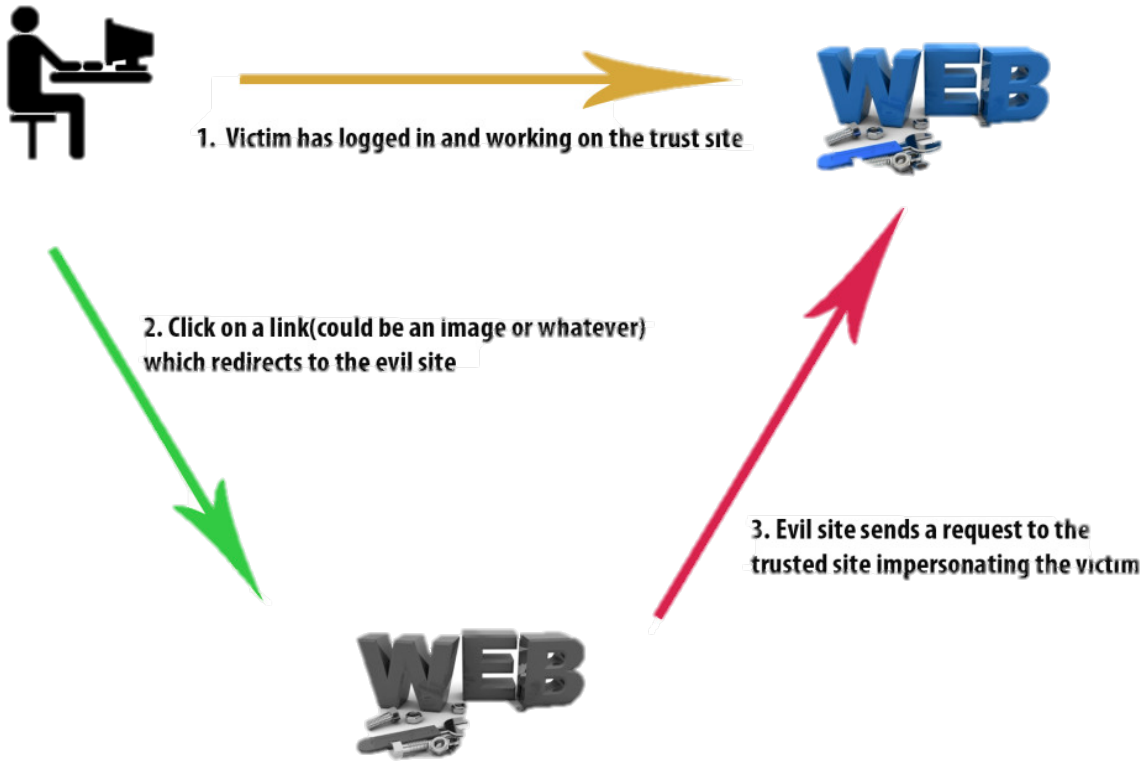
*Code from the TRENDnet TEW-812DRU

Frequently Discovered Vulnerabilities

- **Command Injection**
- **Cross-Site Request Forgery**
- **Memory Corruption (e.g., Buffer Overflow)**
- **Missing Function Level Access Control**
  - Authentication Bypass
  - Authorization Failure
- **Information Disclosure**
- **Backdoor**
- **Poor Session Management**
  - Deterministic Cookie Generation
- **Directory Traversal**
  - Arbitrary File Upload and Download

Cross-Site Request Forgery

#define: CSRF is an attack that forces an unsuspecting victim into executing web commands that perform unwanted actions on a web application.



Testing for Cross-Site Request Forgery

- **Anti-CSRF Tokens?**
- **HTTP referrer checking?**

```
<h1>Password Reset Configuration</h1>
<h3>Choose one of the questions in the list for each question, then provide an answer. You will have to answ
password.</h3>
<h2>Challenge Questions</h2>
<form id="Form1" method="POST" name="PasswordQuestions" style="margin:0" action="">
    <input type="hidden" value="18z2q5m5j7m5v4iufkfsyioh0e3bycnytr6wdq7dsnns4hfvro" name="1k8lin552kl9o0tc">
    <input type="hidden" value="submit" name="submitted">
    <input type="hidden" value="false" name="isSimpleResetEnabled">
```

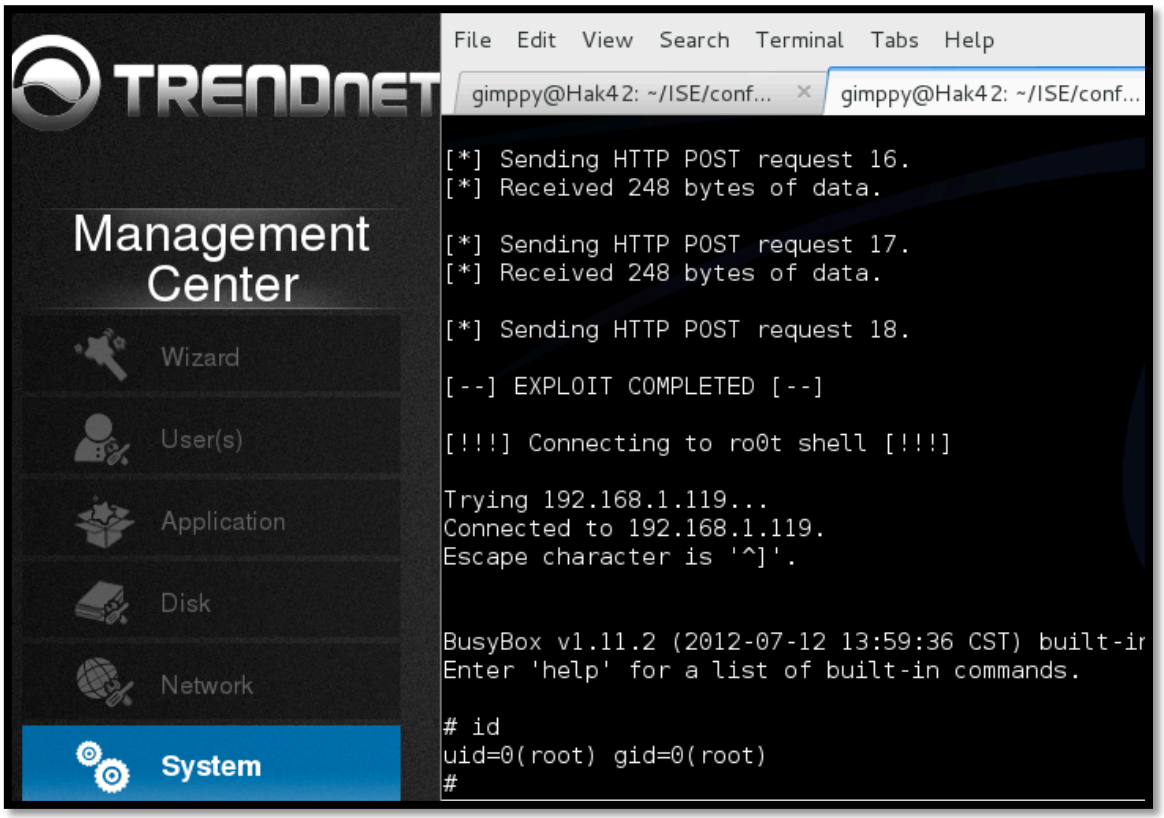
Cross-Site Request Forgery Countermeasures

- **Users**
  - Logout of web applications
  - Do NOT save credentials in your browser
- **Developers**
  - Implement Anti-CSRF tokens **AND** HTTP referrer checking
  - Feeling ambitious? Require users to authenticate before performing a state change

Command Injection

#define:

Command Injection is a form of attack where operating system specific commands are injected into a vulnerable application for execution.



Testing for Command Injection

- **Survey the application**
  - Look for application features that could call underlying system functionality(e.g., ping, traceroute)
  - Source code? Static analysis!

- **Test Examples**
  - ifconfig ; cat /etc/passwd ← Linux
  - dir | ipconfig ← Windows/Linux
  - ls /var/www/`<cmd>` or $(<cmd>) ← Linux*

*Command substitution

Command Injection Countermeasures

- **Developers**
  - Avoid calling shell commands when possible
  - If an API does not exist, sanitize user input before passing it to a function that executes system commands.

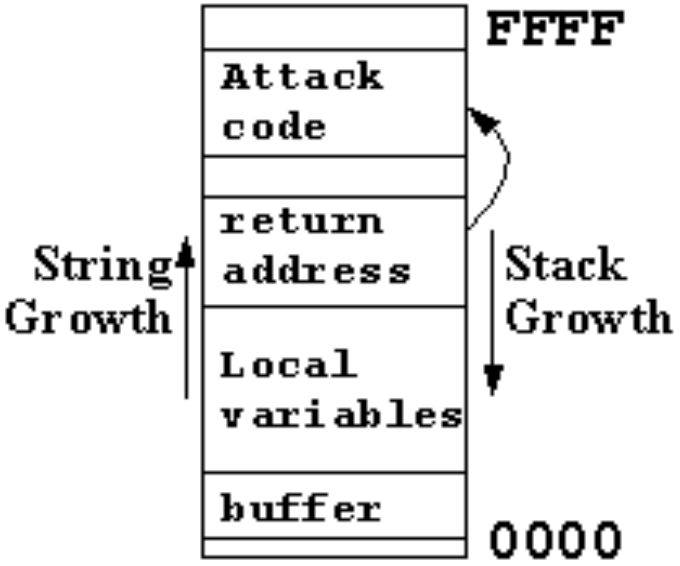
- **Python Example**
  - **BAD:** os.system(‘ls ‘ + dir)
  - **GOOD:** os.listdir(dir)

Buffer Overflow

#define: Buffer Overflows occur when a program attempts to write data that exceeds the capacity of a fixed length buffer, and consequently, overwrites adjacent memory.

```
gimppy@Hak42: ~
File  Edit  View  Search  Terminal  Tabs  Help
root@Hak42: /home/gimppy/ISE/SO...   gimppy@Hak42: ~
Reading symbols from /lib/libgcc_s.so.1...(no debugging symbols fou
Loaded symbols for /lib/libgcc_s.so.1
warning: Unable to find libthread_db matching inferior's thread lib
 be available.
0x40b4fbb0 in msgrcv () from /lib/libc.so.6
1: x/i $pc
=> 0x40b4fbb0 <msgrcv+80>:      mov     r7, r0
(gdb) c
Continuing.

Program received signal SIGSEGV, Segmentation fault.
0x44444444 in ?? ()
1: x/i $pc
=> 0x44444444:  <error: Cannot access memory at address 0x44444444>
```



Stack Based Buffer Overflow (x86)

Testing for Buffer Overflows

- **Testing for overflows**
  - Dynamic Analysis
  - Static Analysis

Buffer Overflow Countermeasures

- **Developers**
  - Don't use unsafe functions
  - Perform bounds checking
  - Compile/Link with overflow prevention techniques
    - Canary/Stack Cookie
      - gcc –fstack-protector
    - ASLR
      - gcc –fPIE || ld -pie
    - DEP/NX
      - gcc marks the stack non-executable by default

Buffer Overflow DEMO

ASUS RT-N56U (ASUSWRT firmware)

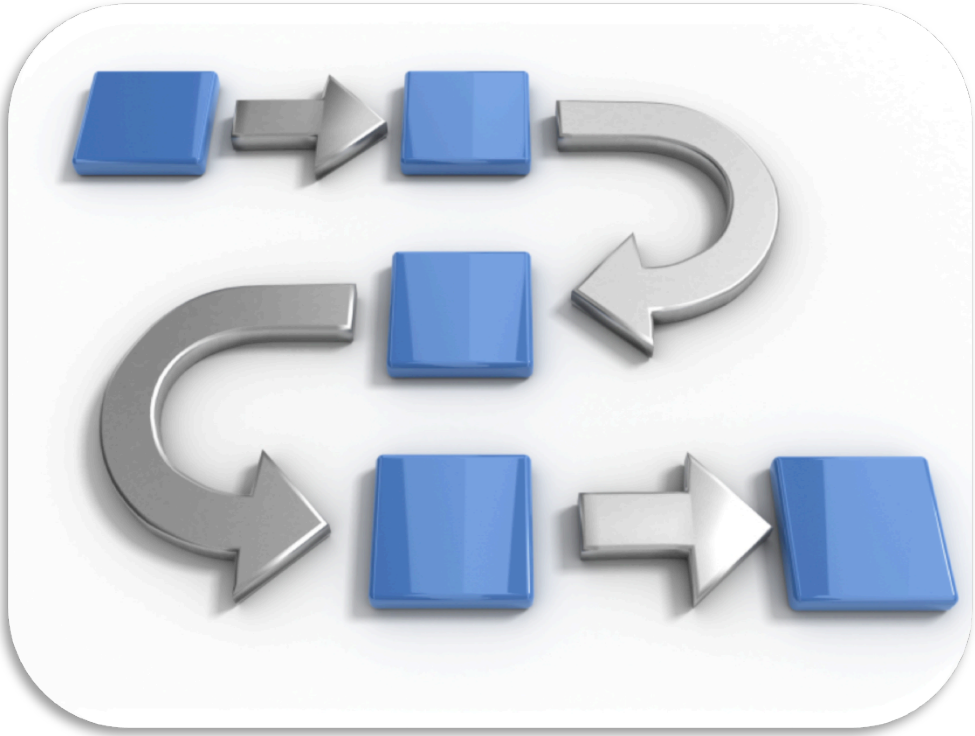
- HTTPD Stack Based Buffer Overflow

If that terrifies you…Mass Exploitation

- **N.A.S.ty Worm**
  - **D-LINK DNS-345**
  - **TRENDnet TN-200/TN-2011T1**
  - **Western Digital MyCloud EX4**
- **Similar Occurrences?**

N.A.S.ty Worm Operation

1. Scan IP Range for TCP/80
2. Fingerprint TCP/80
3. Exploit Target
4. Download and Run
5. Rinse and Repeat



Similar Occurrences

- **Linksys Moon Worm**
- **Qnap ShellShock Worm**
- **Synology Ransomware**

Security Design Principles

- **Least Privilege**
- **Secure by Default**
- **Failsafe**
- **Defense in Depth**
- **Audit**
- **Trust Model**

Least Privilege

- Least privilege refers to the principle that a task should be accomplished with the absolute lowest level of privilege required.

Secure by Default

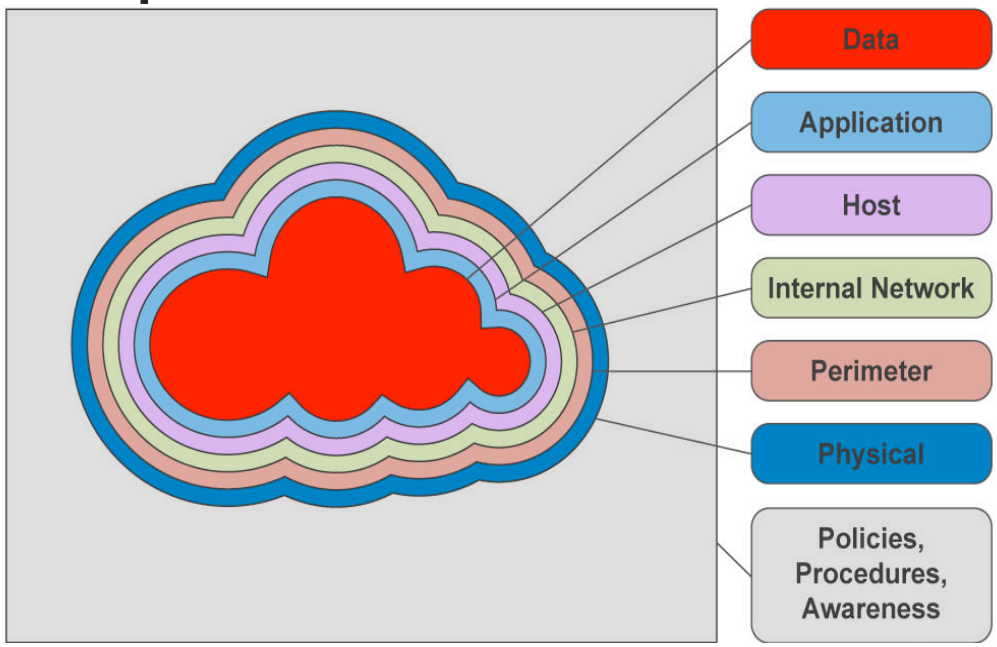
- A system is secure by default if the out-of-the-box settings put the system in a secure state.

Failsafe

- Failing safe refers to the tendency for a failure of the system to leave it in a secure state as opposed to an insecure state.

Defense in Depth

- Layers of defensive measures such that the failure of any one component will not lead to total system compromise.



Audit

- Audit is a critical part of the system that assists in recovery, investigation, and deterrence.

Trust Model

- A trust model clearly defines the trust assumptions made by the system and may highlight improper assumptions on the part of the designers.

IoT Integration

- **Threat Modeling**
- **Security Design Principles**

YIKES! What can we do?

- **Consumers**
  - Harden your IoT devices.
  - Demand that vendors put more emphasis into securing IoT networking equipment.
- **Vendors**
  - Abide by Security Design Principles
  - Follow coding best practices
  - Patch management

Presenter Information

Name: Jacob Holcomb
Twitter: @rootHak42
Blog: http://infosec42.blogspot.com
Github: https://github.com/Gimppy042