Cracking into embedded devices
And beyond! - by Adrian Pastor
Most devices have web interfaces enabled by default
This applies to consumer and corporate appliances
• The devices are ownable via their web interface
• Not just info theft is possible but also gaining root/admin privileges
Why “and beyond”? 

- Attack doesn’t end after owning the embedded device 
- If device not properly segmented, we can probe the *internal network*
• Internet -> target device -> LAN
• Target device: stepping stone / bouncing point
• Not many companies consider DMZing “miscellaneous” devices

Why “and beyond”? (2)
Most of what we need to probe the LAN already on device
i.e.: Axis camera with shell scripting (mish) and PHP support

Why “and beyond”? (3)
• Who’s paying attention to printers, cameras, etc? Anyone?
• After all they’re just primitive devices
• Not taking into account as seriously as app / web servers security-wise

Why “and beyond”? (4)
Focus on remotely exploitable web bugs

- Can be exploited reliably
- Can be hard to detect by IDS
- No need to develop platform-specific shellcode
• Devices’ web interfaces often developed without parameter filtering in mind
  ◦ Real example: Linksys WAG54GS [1]
    • Tons of persistent XSS
• Lots of possibilities / attack scenarios
The juicy bugs!

- Auth bypass
- File retrieval / directory traversal
- XSS - reflected and persistent!
- CSRF - most devices are affected
- Privilege escalation
• Any admin setting can be changed
• Ideal when web int. NOT enabled on WAN
• Payload is launched when admin tricked to visit 3rd-party evil page
• Evil page makes browser send forged request to vulnerable device
• Web server password-protected but enabled on WAN interface
• Attacker *doesn’t* need to be authenticated
• Malformed request to web server injects malicious payload on logs page

**Personal Fav. #2:** Persistent XSS on logs page
- Admin browses vulnerable page while logged in
- Device is compromised – ie: new admin account is added
- Example: Axis 2100 IP cameras [2]
Personal Fav. #2: Persistent XSS on logs page (cont)

• Ironic: security-conscious admins get owned
No interaction required from victim admin
Usually simple to exploit. i.e.:
- knowledge of “authenticated” URL
- Replay request that changes admin setting
- No need to rely on 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

**Personal Fav. #4:**
**Preauth leak + XSS on preauth URL**
- Steal session IDs
- Overwrite login form’s ‘action’ attribute
- Phishing heaven!
- Real example: Pers. XSS on Aruba 800 Mobility Controller's login page [3] – by Jan Fry
  ◦ You own the controller you own all the WAPs – sweet! 😊

**Personal Fav. #4:** Pers. XSS on admin login page
• Because not needing to rely on cracking a weak password is great
• Let’s see review a few real examples
Auth bypass type 1: unprotected URLs

- Password prompt returned when accessing http://victim.foo/
- If creds correct, then redirect to “authed” URL
- Problem is no auth data (ie: password/session ID) is transmitted
- Simply knowing the admin URLs does the job! - ie: http://victim.foo/admin-settings.cgi
- Real example: 3COM APXXXXX (vuln not published yet)
• Resources (URLs) password protected
• However, assumed to be accessed via a certain method – ie: GET
• Requesting resource as POST gives the goodies!
• Real example: BT Voyager 2091 Wireless ADSL [4]
Auth bypass type 2: unchecked HTTP methods (cont)

- Get config file without password:

  ```
  POST /psiBackupInfo HTTP/1.1
  Host: 192.168.1.1
  Connection: close
  Content-Length: 0
  <CRLF>
  <CRLF>
  ```
- Admin URLs password-protected correctly
- However, admin requests are NOT
- Real example: Linksys WRT54GS [5] – by Ginsu Rabbit
- Settings URLs requires password: 
  `GET /wireless.htm`

- Submitting admin request does NOT:
  `POST /Security.tri`
  `Content-Length: 24`
  `SecurityMode=0&layout=en`
Auth bypass type 4: URL fuzzing

- Web server OKs multiple representations of URL
- i.e.: the following URLs could all be valid:
  - http://victim.foo/path/
  - http://victim.foo\path\n  - http://victim.foo/path?
  - http://victim.foo/path?anyparameter=anyvalue
  - http://victim.foo/path/
  - http://victim.foo/path//
• Real example: BT Home Hub and Thomson/Alcatel Speedtouch 7G [6]
• i.e.: the following URL gives you the config file without supplying creds:
  ◦ http://192.168.1.254/cgi/b/backup/user.ini//
• No open tcp/udp ports on WAN interface by default
• Requirement: attack must be remote
• Most people would give up at this point
• Possible attack vectors, anyone?
• OK, WAN is not an option
• How about the **LAN interface**?
• “Didn’t you say it must be a remote attack?” you must be thinking 😊
• Think **client side**!
• Victim user’s **browser** his worst enemy
• If you can’t attack via WAN, let the internal user do it via **LAN**
• The aikido way: blend in, take advantage of already-established channels
• The recipe:
  ◦ CSRF
  ◦ Auth bypass
• The weapon:
  ◦ Simple form retrieved via hidden `iframe`
The attack:
- Any user in Home Hub’s LAN visits a malicious web page
- Web page causes user’s browser to submit an interesting request to Home Hub, i.e.: enable remote assistance

BT Home Hub hacking challenge (cont)
Demo time!
[1] Persistent XSS and CSRF on Linksys WAG54GS router
http://www.gnucitizen.org/blog/persistent-xss-and-csrf-on-
wireless-g-adsl-gateway-with-speedbooster-wag54gs

[2] Persistent XSS on Aruba 800 Mobility Controller's login
page
http://www.securityfocus.com/bid/26465

[3] Multiple vulnerabilities on Axis 2100 IP cameras
http://www.procheckup.com/Vulnerability_Axis_2100_rese
arch.pdf
http://www.securityfocus.com/archive/1/440405
http://www.securityfocus.com/bid/19057/discuss

[5] Linksys WRT54GS POST Request Configuration Change Authentication Bypass Vulnerability
http://www.securityfocus.com/archive/1/442452/30/0/threaded
http://www.securityfocus.com/bid/19347
[6] BT Home Flub: Pwnin the BT Home Hub
http://www.gnucitizen.org/blog/bt-home-flub-pwnin-the-bt-home-hub
http://www.gnucitizen.org/blog/bt-home-flub-pwnin-the-bt-home-hub-3
http://www.gnucitizen.org/blog/bt-home-flub-pwnin-the-bt-home-hub-4