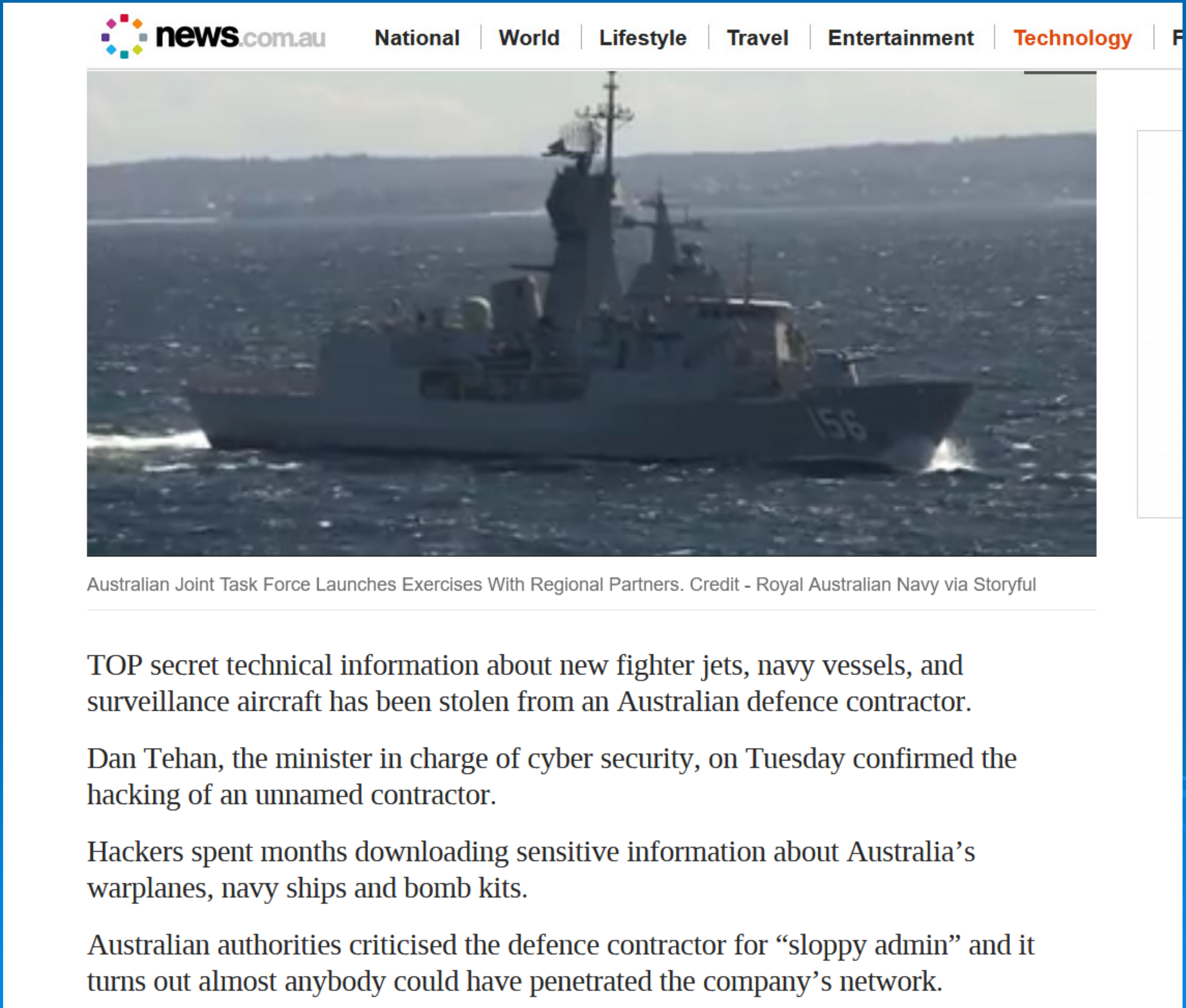
COMS 3000/7003

Week 11

PKI, TLS, Firewalls, Network Security and Monitoring: Logs, Audit Trails and Security Information and Event Management (SIEM)

Australian Defence Contractor Breached



Australian Joint Task Force Launches Exercises With Regional Partners. Credit - Royal Australian Navy via Storyful

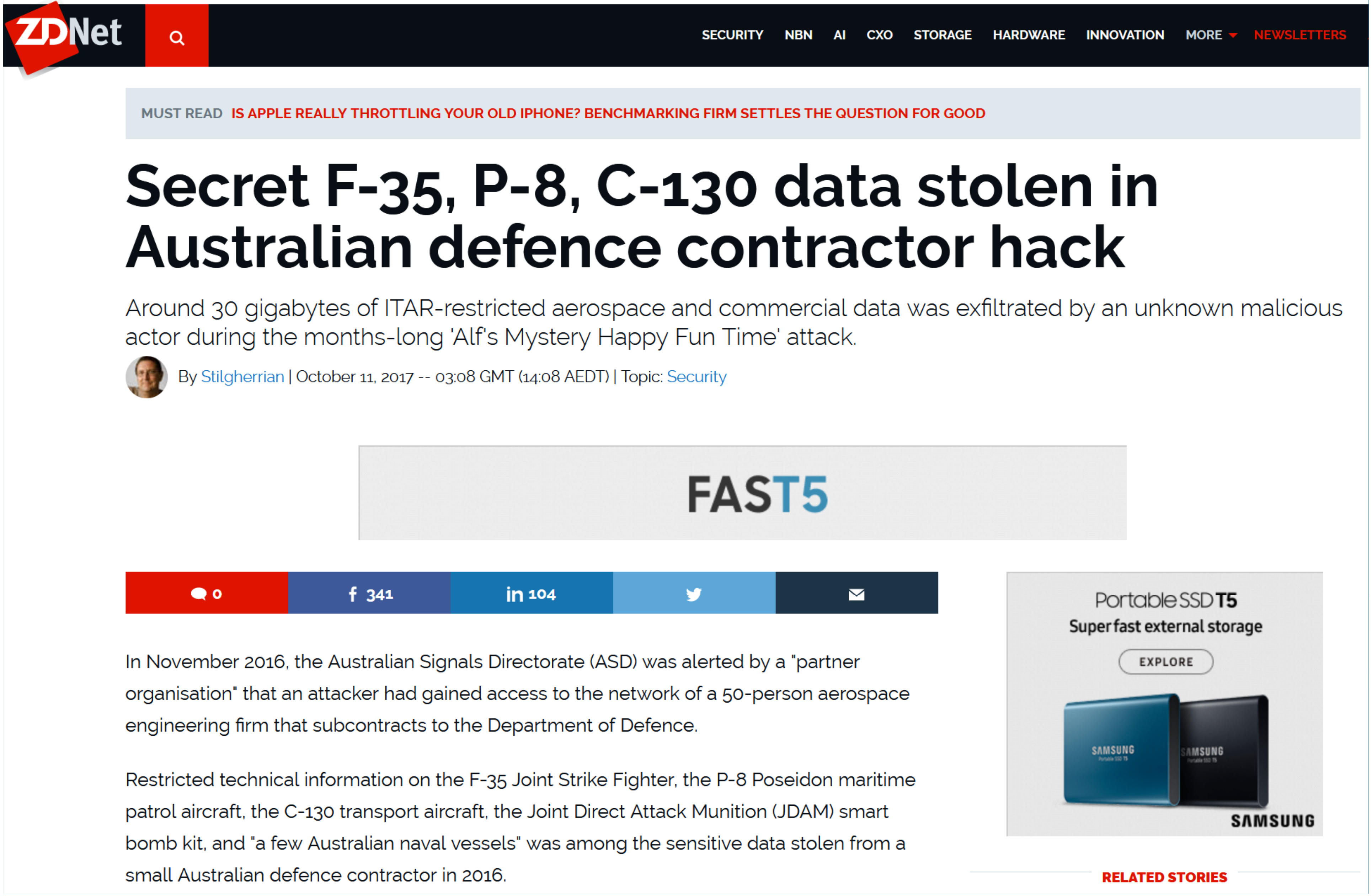
TOP secret technical information about new fighter jets, navy vessels, and surveillance aircraft has been stolen from an Australian defence contractor.

Dan Tehan, the minister in charge of cyber security, on Tuesday confirmed the hacking of an unnamed contractor.

Hackers spent months downloading sensitive information about Australia's warplanes, navy ships and bomb kits.

Australian authorities criticised the defence contractor for "sloppy admin" and it turns out almost anybody could have penetrated the company's network.

http://www.zdnet.com/article/secret-f-35-p-8-c-130-data-stolen-in-australian-defence-contractor-hack/



MUST READ IS APPLE REALLY THROTTLING YOUR OLD IPHONE? BENCHMARKING FIRM SETTLES THE QUESTION FOR GOOD

Secret F-35, P-8, C-130 data stolen in Australian defence contractor hack

Around 30 gigabytes of ITAR-restricted aerospace and commercial data was exfiltrated by an unknown malicious actor during the months-long 'Alf's Mystery Happy Fun Time' attack.

By Stilgherrian | October 11, 2017 -- 03:08 GMT (14:08 AEDT) | Topic: Security

In November 2016, the Australian Signals Directorate (ASD) was alerted by a "partner organisation" that an attacker had gained access to the network of a 50-person aerospace engineering firm that subcontracts to the Department of Defence.

Restricted technical information on the F-35 Joint Strike Fighter, the P-8 Poseidon maritime patrol aircraft, the C-130 transport aircraft, the Joint Direct Attack Munition (JDAM) smart bomb kit, and "a few Australian naval vessels" was among the sensitive data stolen from a small Australian defence contractor in 2016.

RELATED STORIES

Tutorials

(Tutorials back to full length now that the assignment is done.)

Lessons in Tutorial Question 10

There are many ways to do it!

Modulo 33

- 0 – 32
- Can’t be used for numbers 33+ (6 bits)
- Therefore maximum 5 bits in plaintext
- But can produce “32” in ciphertext (6 bits)
- Ciphertext output is 6 bits
- RSA output is bigger than the input

message > modulus

- M = 01100111 (103)
- C = 00010000 (16)
- M = 00000100 (4) X

Digital Certificates

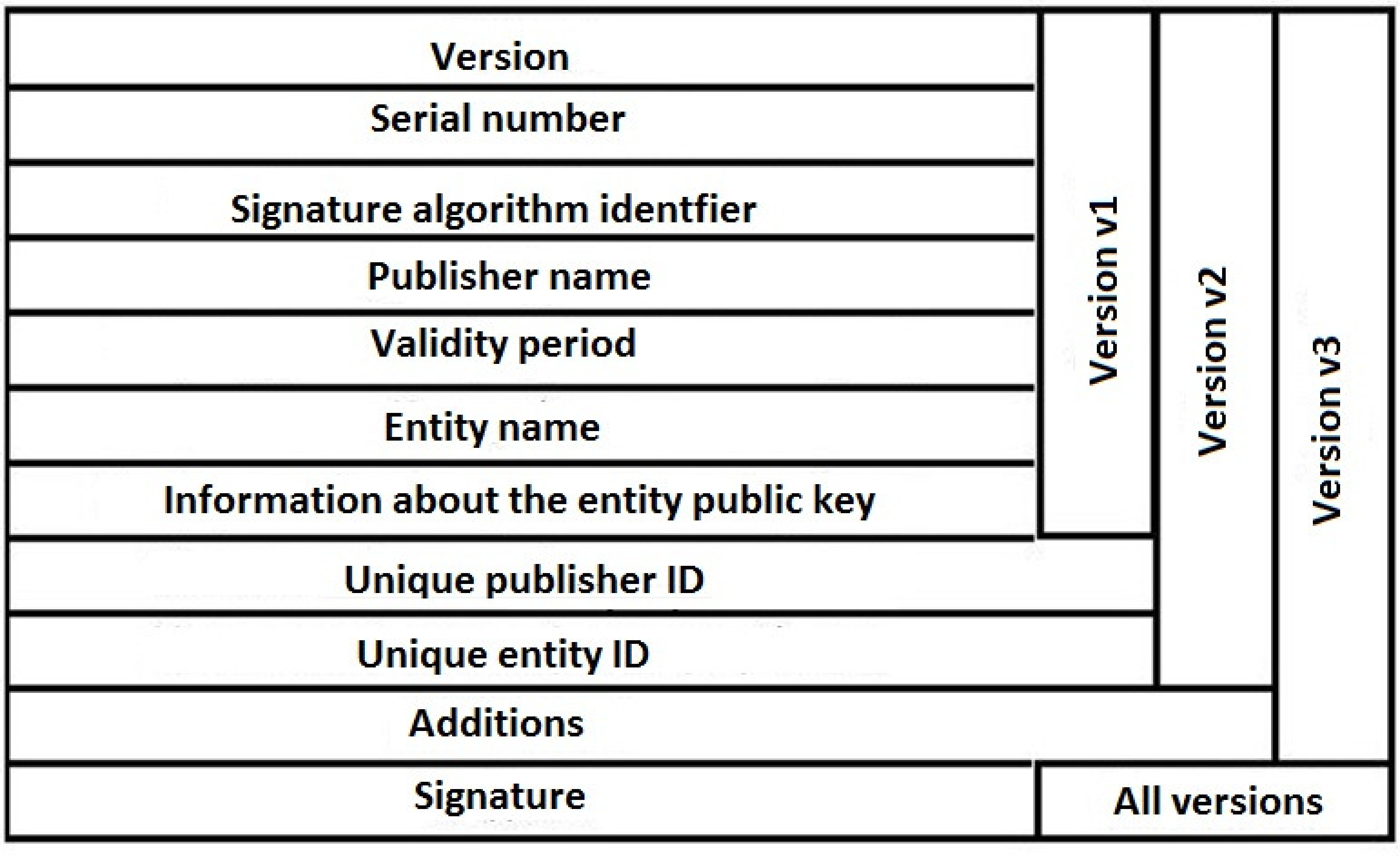
(continued)

X.509 Certificates

- X.509: Public Key Certificate Standard issued by ITU-T
  - Defines format of certificates
  - Most commonly used format of certificates
- More details:
  - http://en.wikipedia.org/wiki/X.509
  - http://tools.ietf.org/html/rfc5280

- Structure of a X.509 v3 certificate
  - Certificate
    - Version
    - Serial Number
    - Algorithm ID
    - Issuer
    - Validity
      - Not Before
      - Not After
    - Subject
    - Subject Public Key Info
      - Public Key Algorithm
      - Subject Public Key (e.g. *n* and *e* for RSA)
    - Issuer Unique Identifier (Optional)
    - Subject Unique Identifier (Optional)
    - Extensions (Optional)
      - …
  - Certificate Signature Algorithm
  - Certificate Signature

X.509 Certificates



Version

Serial number

Signature algorithm identfier

Publisher name

Validity period

Entity name

Information about the entity public key

Unique publisher ID

Unique entity ID

Additions

Signature

Version v1

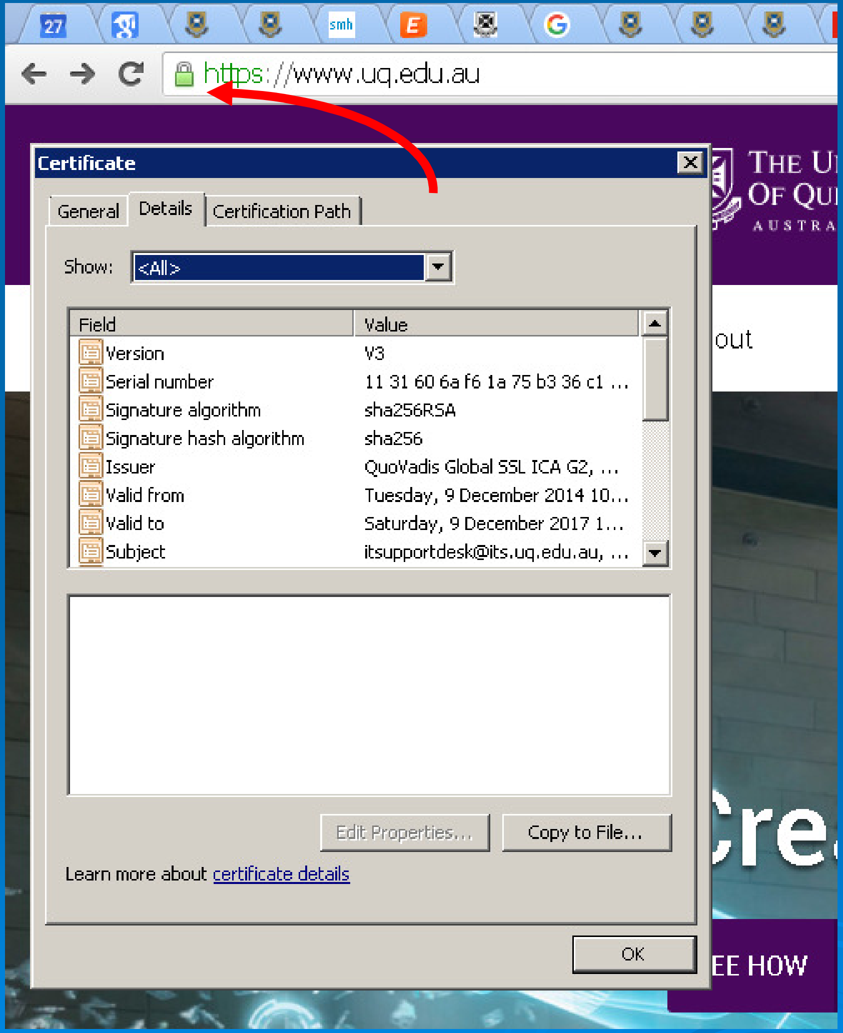
Version v2

Version v3

All versions

Image: http://cryptowiki.net/index.php?title=Asymmetric_key_management

Certificate Example



https://www.uq.edu.au

Certificate

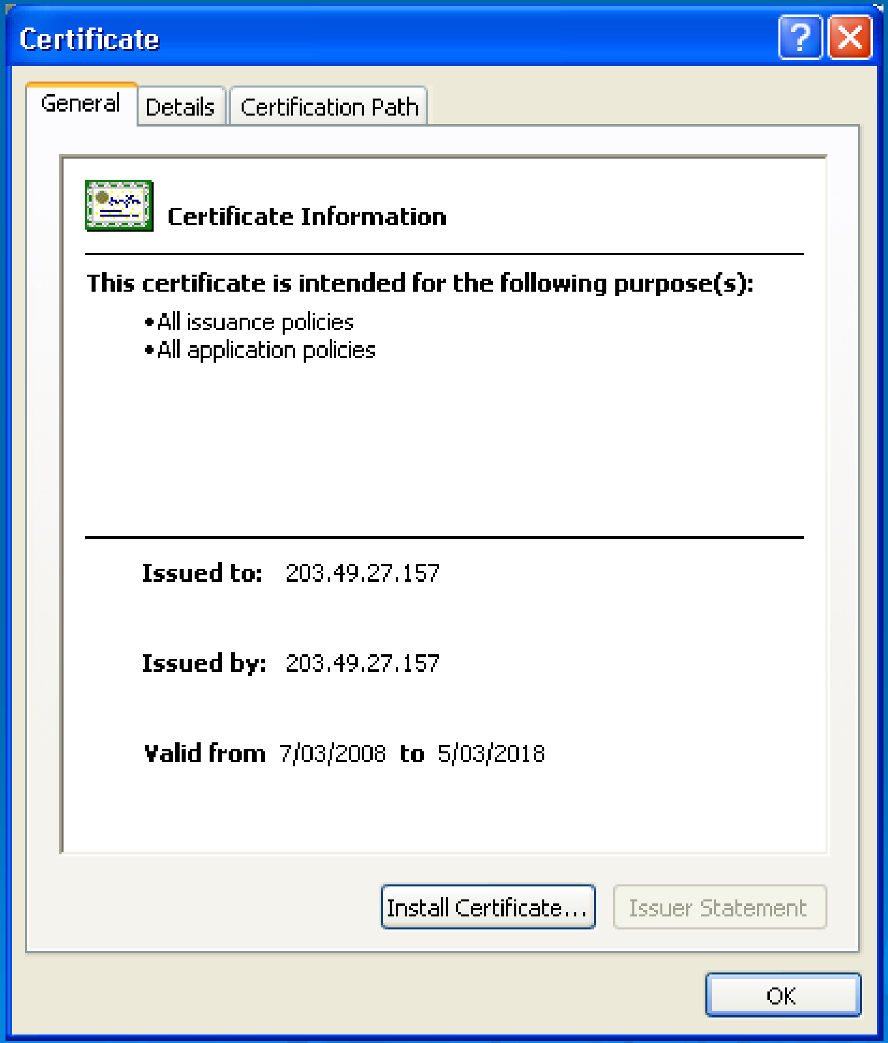
Show: <All>

Field	Value
Version	V3
Serial number	11 31 60 6a f6 1a 75 b3 36 c1 ...
Signature algorithm	sha256RSA
Signature hash algorithm	sha256
Issuer	QuoVadis Global SSL ICA G2, ...
Valid from	Tuesday, 9 December 2014 10...
Valid to	Saturday, 9 December 2017 1...
Subject	itsupportdesk@its.uq.edu.au, ...

Learn more about certificate details

- Certificates are used to establish secure connections in TLS/SSL or HTTPS
  - Discussed later today
- Certification Path shows certification chain

A Root (self-signed) Certificate



Certificate Information

This certificate is intended for the following purpose(s):

- All issuance policies
- All application policies

Issued to: 203.49.27.157

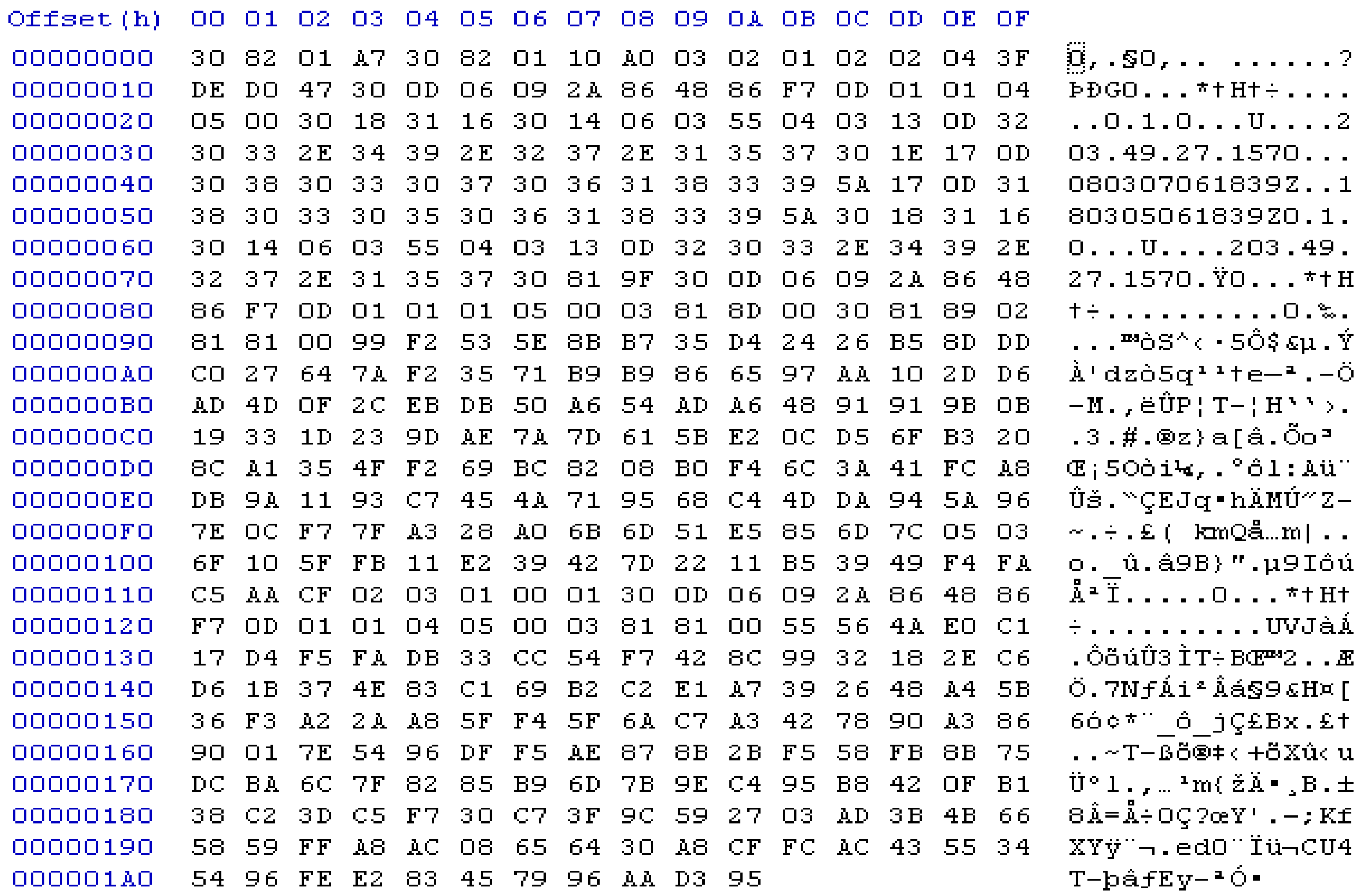
Issued by: 203.49.27.157

Valid from 7/03/2008 **to** 5/03/2018

.PEM or .CER

```
-----BEGIN CERTIFICATE-----
MIIBpzCCARCgAwIBAgIEP97QRzANBgkqhkiG9w0BAQQFADAYMRYwFAYDVQQDEw0y
MDMuNDkuMjcuMTU3MB4XDTA4MDMwNzA2MTgzOVoXDTE4MDMwNTA2MTgzOVowGDEW
MBQGA1UEAxMNMjAzLjQ5LjI3LjE1NzCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkC
gYEAmfJTXou3NdQkJrWN3cAnZHryNXG5uYZll6oQLdatTQ8s69tQplStpkiRkZsL
GTMdI52uen1hW+IM1W+zIIyhNU/yabyCCLDObDpB/KjbmhGTxOVKcZVoxE3alFqW
fgz3f6MooGttUeWFbXwFA28QX/sR4jlCfSIRtTlJ9PrFqs8CAwEAATANBgkqhkiG
9w0BAQQFAAOBgQBVVkrgwRfU9frbM8xU9OKMmTIYLsbWGzdOg8FpssLhpzkmSKRb
NvOiKqhf9F9qx6NCeJCjhpABflSW3/Wuh4sr9Vj7i3Xcumx/goW5bXuexJW4Qg+x
OMI9xfcwxz+cWScDrTtLZlhZ/6isCGVkMKjP/KxDVTRUlv7igOV5lqrTlQ==
-----END CERTIFICATE-----
```

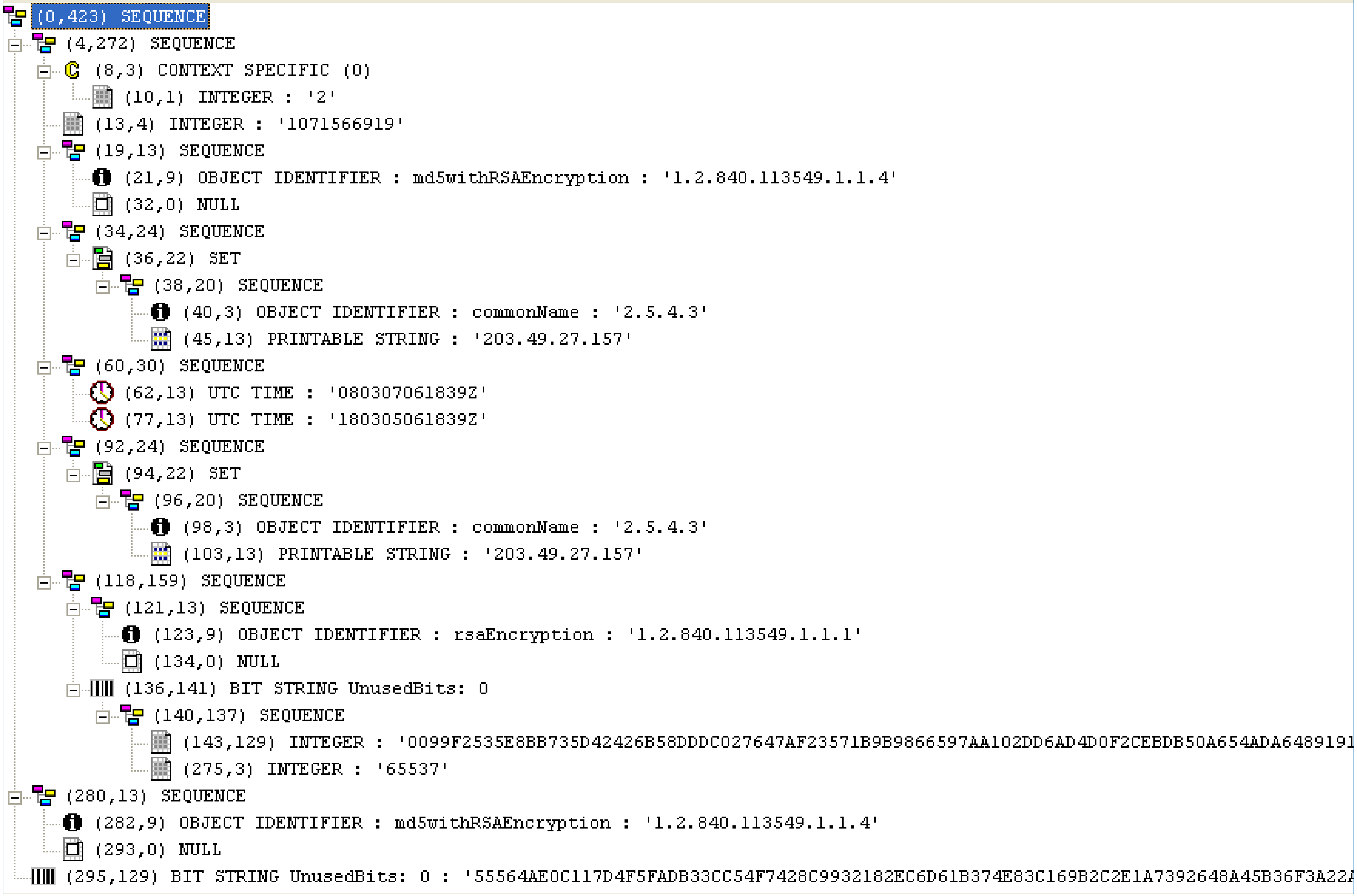
Base64 Decode



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  30 82 01 A7 30 82 01 10 A0 03 02 01 02 02 04 3F  0‚.§0‚.. ......?
00000010  DE D0 47 30 0D 06 09 2A 86 48 86 F7 0D 01 01 04  ÞÐG0...*†H†÷....
00000020  05 00 30 18 31 16 30 14 06 03 55 04 03 13 0D 32  ..0.1.0...U....2
00000030  30 33 2E 34 39 2E 32 37 2E 31 35 37 30 1E 17 0D  03.49.27.1570...
00000040  30 38 30 33 30 37 30 36 31 38 33 39 5A 17 0D 31  080307061839Z..1
00000050  38 30 33 30 35 30 36 31 38 33 39 5A 30 18 31 16  80305061839Z0.1.
00000060  30 14 06 03 55 04 03 13 0D 32 30 33 2E 34 39 2E  0...U....203.49.
00000070  32 37 2E 31 35 37 30 81 9F 30 0D 06 09 2A 86 48  27.1570.Ÿ0...*†H
00000080  86 F7 0D 01 01 01 05 00 03 81 8D 00 30 81 89 02  †÷..........0.‰.
00000090  81 81 00 99 F2 53 5E 8B B7 35 D4 24 26 B5 8D DD  ...™òS^‹·5Ô$&µ.Ý
000000A0  C0 27 64 7A F2 35 71 B9 B9 86 65 97 AA 10 2D D6  À'dzò5q¹¹†e—ª.-Ö
000000B0  AD 4D 0F 2C EB DB 50 A6 54 AD A6 48 91 91 9B 0B  -M.,ëÛP¦T-¦H‘‘›.
000000C0  19 33 1D 23 9D AE 7A 7D 61 5B E2 0C D5 6F B3 20  .3.#.®z}a[â.Õo³
000000D0  8C A1 35 4F F2 69 BC 82 08 B0 F4 6C 3A 41 FC A8  Œ¡5Oòi¼‚.°ôl:Aü¨
000000E0  DB 9A 11 93 C7 45 4A 71 95 68 C4 4D DA 94 5A 96  Ûš.“ÇEJq•hÄMÚ”Z–
000000F0  7E 0C F7 7F A3 28 A0 6B 6D 51 E5 85 6D 7C 05 03  ~.÷.£( kmQå…m|..
00000100  6F 10 5F FB 11 E2 39 42 7D 22 11 B5 39 49 F4 FA  o._û.â9B}".µ9Iôú
00000110  C5 AA CF 02 03 01 00 01 30 0D 06 09 2A 86 48 86  ÅªÏ.....0...*†H†
00000120  F7 0D 01 01 04 05 00 03 81 81 00 55 56 4A E0 C1  ÷..........UVJàÁ
00000130  17 D4 F5 FA DB 33 CC 54 F7 42 8C 99 32 18 2E C6  .ÔõúÛ3ÌT÷BŒ™2..Æ
00000140  D6 1B 37 4E 83 C1 69 B2 C2 E1 A7 39 26 48 A4 5B  Ö.7NƒÁi²Âá§9&H¤[
00000150  36 F3 A2 2A A8 5F F4 5F 6A C7 A3 42 78 90 A3 86  6ó¢*¨_ô_jÇ£Bx.£†
00000160  90 01 7E 54 96 DF F5 AE 87 8B 2B F5 58 FB 8B 75  ..~T–ßõ®‡‹+õXû‹u
00000170  DC BA 6C 7F 82 85 B9 6D 7B 9E C4 95 B8 42 0F B1  Üºl.‚…¹m{žÄ•¸B.±
00000180  38 C2 3D C5 F7 30 C7 3F 9C 59 27 03 AD 3B 4B 66  8Â=Å÷0Ç?œY'.-;Kf
00000190  58 59 FF A8 AC 08 65 64 30 A8 CF FC AC 43 55 34  XYÿ¨¬.ed0¨Ïü¬CU4
000001A0  54 96 FE E2 83 45 79 96 AA D3 95                 T–þâƒEy–ªÓ•
```

ASN.1 DER Decode



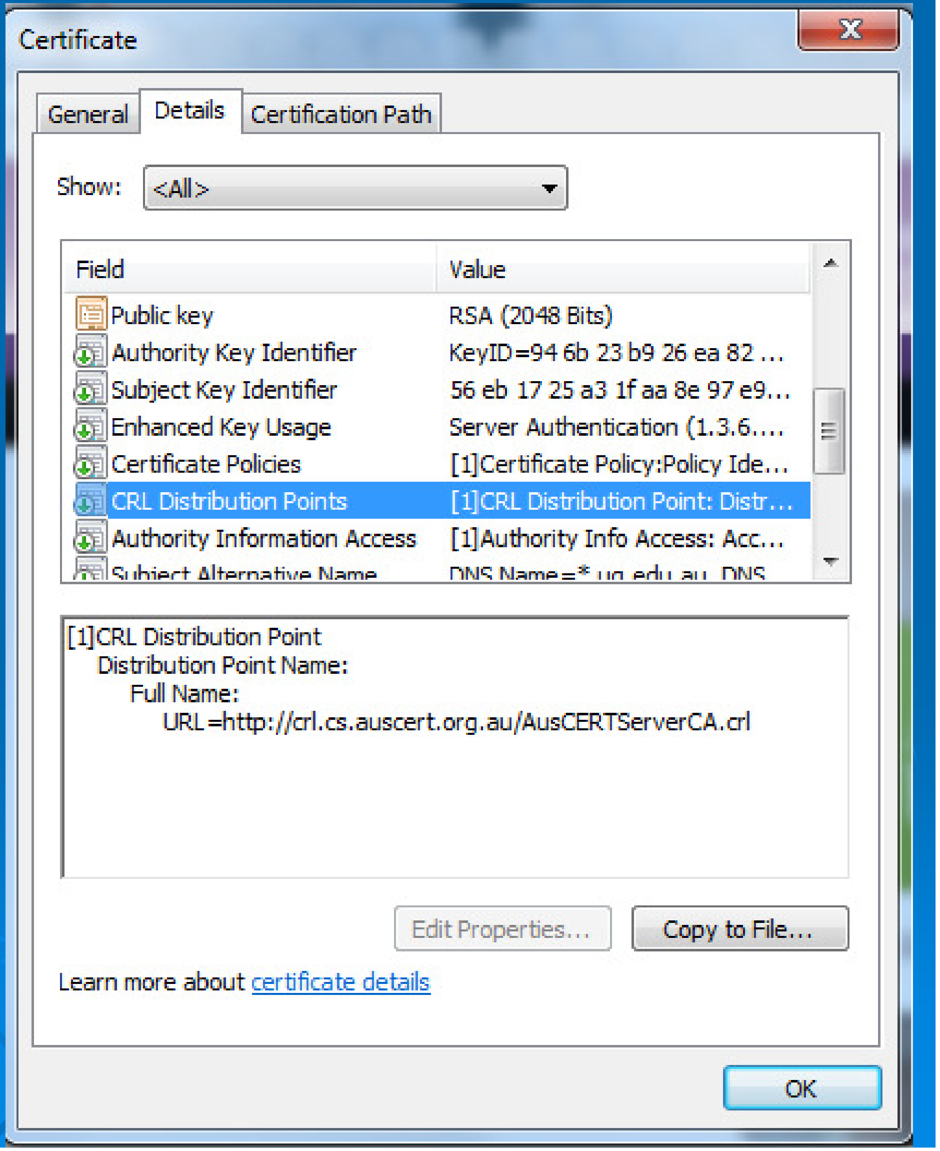
```
(0,423) SEQUENCE
  (4,272) SEQUENCE
    (8,3) CONTEXT SPECIFIC (0)
      (10,1) INTEGER : '2'
    (13,4) INTEGER : '1071566919'
    (19,13) SEQUENCE
      (21,9) OBJECT IDENTIFIER : md5withRSAEncryption : '1.2.840.113549.1.1.4'
      (32,0) NULL
    (34,24) SEQUENCE
      (36,22) SET
        (38,20) SEQUENCE
          (40,3) OBJECT IDENTIFIER : commonName : '2.5.4.3'
          (45,13) PRINTABLE STRING : '203.49.27.157'
    (60,30) SEQUENCE
      (62,13) UTC TIME : '080307061839Z'
      (77,13) UTC TIME : '180305061839Z'
    (92,24) SEQUENCE
      (94,22) SET
        (96,20) SEQUENCE
          (98,3) OBJECT IDENTIFIER : commonName : '2.5.4.3'
          (103,13) PRINTABLE STRING : '203.49.27.157'
    (118,159) SEQUENCE
      (121,13) SEQUENCE
        (123,9) OBJECT IDENTIFIER : rsaEncryption : '1.2.840.113549.1.1.1'
        (134,0) NULL
      (136,141) BIT STRING UnusedBits: 0
        (140,137) SEQUENCE
          (143,129) INTEGER : '0099F2535E8BB735D42426B58DDDC027647AF23571B9B9866597AA102DD6AD4D0F2CEBDB50A654ADA6489191
          (275,3) INTEGER : '65537'
  (280,13) SEQUENCE
    (282,9) OBJECT IDENTIFIER : md5withRSAEncryption : '1.2.840.113549.1.1.4'
    (293,0) NULL
  (295,129) BIT STRING UnusedBits: 0 : '55564AE0C117D4F5FADB33CC54F7428C9932182EC6D61B374E83C169B2C2E1A7392648A45B36F3A22A
```

What if a Private Key is Compromised/Leaked?

- Corresponding public key should not be used any more
- Revocation of corresponding public key certificate
  - ID of certificate is added to a Certificate Revocation List (CRL), published by the responsible CA
- Applications/Protocols should always check current CRLs before accepting a certificate

CRL

- CRL Distribution Point



Show: <All>

Field	Value
Public key	RSA (2048 Bits)
Authority Key Identifier	KeyID=94 6b 23 b9 26 ea 82 ...
Subject Key Identifier	56 eb 17 25 a3 1f aa 8e 97 e9...
Enhanced Key Usage	Server Authentication (1.3.6....
Certificate Policies	[1]Certificate Policy:Policy Ide...
CRL Distribution Points	[1]CRL Distribution Point: Distr...
Authority Information Access	[1]Authority Info Access: Acc...
Subject Alternative Name	DNS Name=*.uq.edu.au, DNS

```
[1]CRL Distribution Point
     Distribution Point Name:
          Full Name:
               URL=http://crl.cs.auscert.org.au/AusCERTServerCA.crl
```

Learn more about certificate details

Public Key Infrastructure (PKI)

- What is a PKI?
  - All the things you need to make public key cryptography work (and secure)
  - “A public-key infrastructure (PKI) is a set of hardware, software, people, policies, and procedures needed to create, manage, distribute, use, store, and revoke digital certificates. ...”
    - http://en.wikipedia.org/wiki/Public_key_infrastructure
- Key components
  - Certificates
    - Binds identity to public key, typically X.509
  - Certification Authorities (CAs)
    - Issue Certificates, with digital signature
  - Certificate Revocation Lists (CRLs)
    - List of certificates (serial numbers) that should no longer be trusted
- Certificates are used in TLS/SSL to authenticate the server, as we will see later today.

Quick detour back to Symmetric Key Cryptography:

Message Authentication Codes (MAC)

Efficient Authentication/Integrity

- Assume we want to provide authentication and integrity for packets in a secure network protocol, e.g. SSL/TLS.
- How can we do this?
  - Digital Signature, e.g. SHA-2 + RSA
- Problem
  - Using Public Key cryptography is still relatively expensive (even if it is only on a hash), especially if it has to be done on a packet per packet basis
- Can we do this more efficiently using a secret-key algorithm?
  - Yes we can.

Authentication/Integrity with Secret-key cryptography

- How can we provide authentication and integrity using only a secret key and a cryptographic one-way hash function?
  - Assume Alice and Bob share a secret key K
  - Alice sends a message *m* to Bob
    - We want Authentication and Integrity
  - Basic Idea
    - Alice computes a cryptographic checksum or Message Authentication Code (MAC)
    - MAC = h(K || m)
  - If Trudy alters *m*, she cannot compute a valid MAC without knowing K
    - → Integrity
  - Knowing K, Bob can verify the MAC. Only someone knowing K (i.e. Alice) would have been able to compute a valid MAC
    - → Authenticity

Trudy, the evil, active attacker

HMAC

- Problem:
  - Simple MAC = h(K || m) is not secure for a hash functions such as SHA-1, SHA-2 or md5, based on the so-called *Merkle–Damgård* construction
  - Susceptible to so called ‘length extension attack’
- A more complex, nested version is used to make it secure, called *HMAC*
  - HMAC: Keyed hashing for Message Authentication
  - Most widely used MAC in the Internet, IETF Standard RFC2104
- $HMAC(K,m) = H((K' \oplus opad) \parallel H((K' \oplus ipad) \parallel m))$
  - K’: key hashed or padded to blocksize
  - opad: outer padding, constant 0x5c
  - ipad: inner padding, constant 0x36
  - ⊕: XOR
  - ‖: concatenation
- HMAC does not rely on collision resistance of hash function, so is secure even with ‘weak’ hash functions such as md5 or SHA-1!
- New SHA-3 (Keccak) is NOT vulnerable to length extension attack.
- Therefore, simple MAC = SHA-3(K || m) is secure!
- http://en.wikipedia.org/wiki/Hash-based_message_authentication_code

TLS/SSL

Transport Layer Security (TLS)

Secure Socket Layer (SSL)

SSL/TLS - History

- ‘Secure Socket Layer’ (SSL)
  - Developed by Netscape in 1994
  - Versions 1.0 -> 3.0
- Goal:
  - Provide Authentication, Integrity and Confidentiality of communication between web browser and server
  - Design is generic, SSL/TLS can be used with any TCP-based application

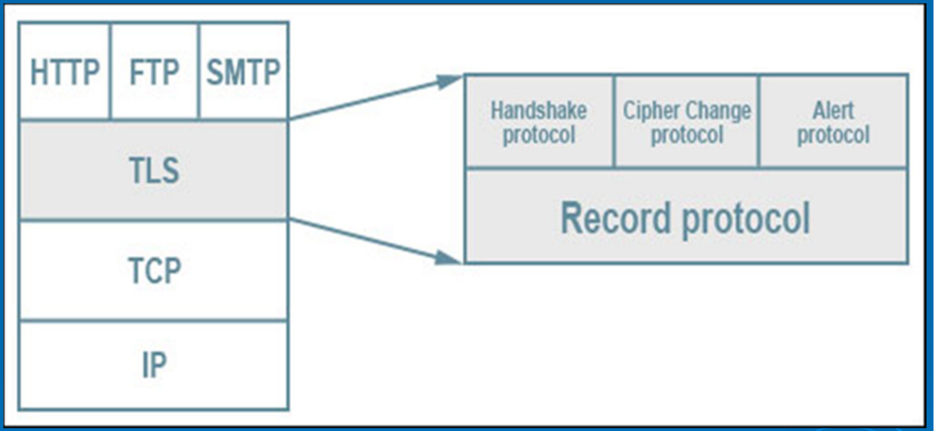
- SSL has been adopted by the IETF (Internet Engineering Task Force) as a standard
  - With very minor modifications
  - → ‘Transport Layer Security’ (TLS)

  - TLS 1.0 based on SSL 3.0 (but not interoperable), RFC2246, 1999
  - TLS 1.1 RFC4346, 2006
    - Fixed a few weakness
  - TLS 1.2 RFC 5246, 2008
    - Added more secure hash functions, e.g. SHA-256
    - http://tools.ietf.org/html/rfc5246

- Names “TLS” and “SSL” are often used interchangeably
- Most widely used security protocol

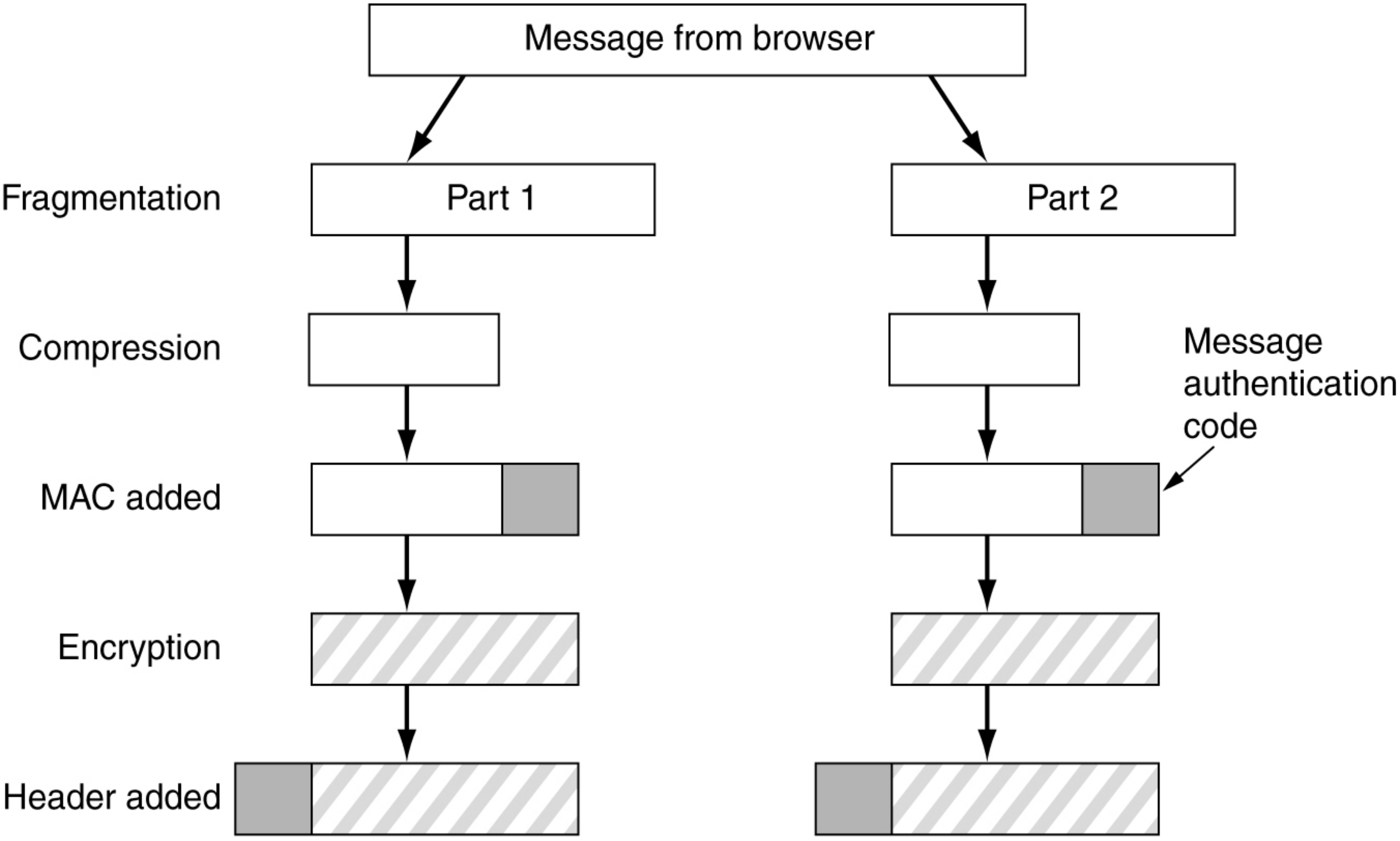
TLS

- TLS sits between the application and the (reliable) transport layer, i.e. TCP
- While TLS is most commonly used to secure web traffic (HTTP), it can be used for any application
  - HTTP over TLS is HTTPS
- Most programming languages provide support for SSL/TLS
  - e.g. Java SSLSocket()



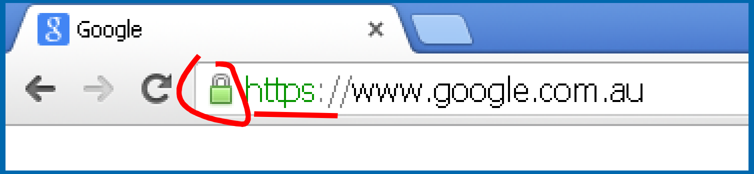
- TLS consists of the following parts
  - Handshake protocol
    - Establishes shared secret key, negotiates cipher suite
  - Cipher Change protocol
    - Enables cipher change
  - Alert Protocol
    - Reports errors
  - Record Protocol
    - Main part, provides secure transport

TLS – Record Protocol

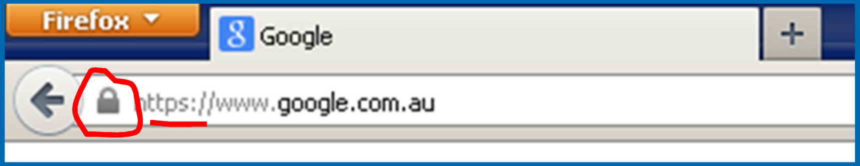


TLS in Web Browsers

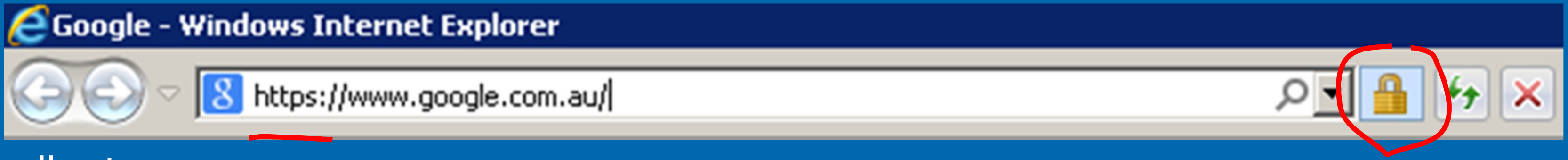
- How can you tell if TLS is active in your web browser?



https://www.google.com.au



https://www.google.com.au



https://www.google.com.au/

- Indicators
  - Padlock
  - *https* instead of *http*
    - *https = 'secure http' or 'http over SSL/TLS'*

- Some websites can be accessed via both HTTP and HTTPS, but increasingly, http:// requests are redirected to the https:// address, e.g. Google.
  - Random example:
    - **A**: http://www.anz.com.au
    - **B**: https://www.anz.com.au

- How does the browser treat these two URLs differently?
  - In A, the browser connects to the web server on port **80**, and request the data via the HTTP protocol
  - In B, the browser connects to the web server on port 443 and sets up a secure SSL/TLS session. Then HTTP is used to request the content over the secured connection.

SSL/TLS

- SSL/TLS provides:
  - Key establishment
  - Authentication
  - Confidentiality
  - Integrity

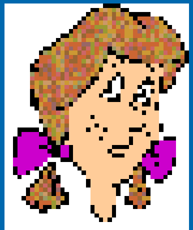
- TLS uses cryptographic hash functions, secret-key ciphers and public-key ciphers
  - A “cipher suite” is a combination specific algorithms to be used in a TLS session
  - Examples (see RFCs for complete list):
    - TLS_RSA_WITH_DES_CBC_SHA
    - TLS_DH_anon_WITH_RC4_128_MD5
    - (Key establishment, cipher, cryptographic hash for HMAC)

TLS Handshake

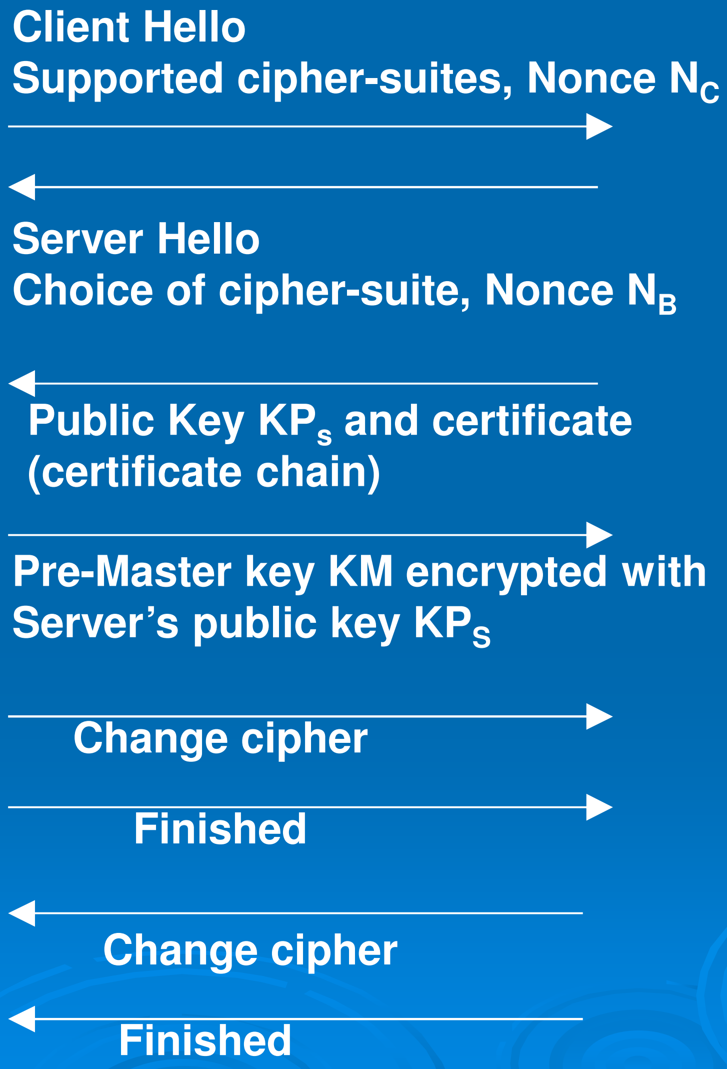
- “Handshake”: Initial phase of a TLS session
- Purpose:
  - Negotiate cipher suite to be used
  - Mutual Authentication of server (and client)
    - Authentication of server is mandatory
    - Client authentication to server is optional
    - In most cases, this is done via public key cryptography, and via the exchange of X.509 certificates or certificate chains
  - Establish shared secret keys for encryption and authentication, and MAC
  - After the handshake, all data sent via the TLS connection is encrypted and integrity is provided via HMAC

SSL/TLS Handshake

Client



Server



Client Hello
Supported cipher-suites, Nonce N_C

Server Hello
Choice of cipher-suite, Nonce N_B

Public Key KP_s and certificate (certificate chain)

Pre-Master key KM encrypted with Server's public key KP_S

Change cipher

Finished

Change cipher

Finished

Client chooses a random 384-bit "pre-master key" KM

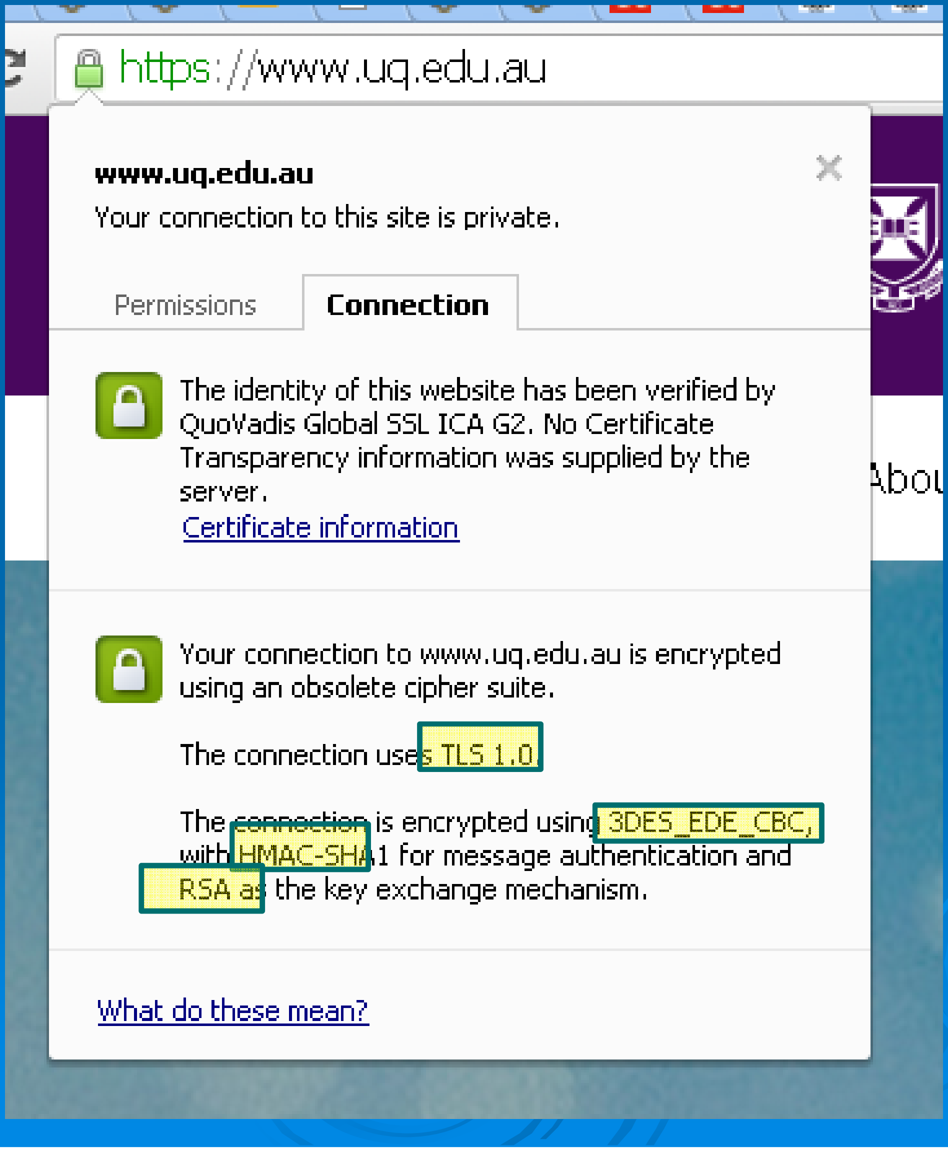
Optionally, the server can request a client certificate here

Client computes shared secret key from KM and Nonces

Server computes shared secret key from KM and Nonces

Secure session is established. All data sent is now protected.

Example



https://www.uq.edu.au

www.uq.edu.au

Your connection to this site is private.

The identity of this website has been verified by QuoVadis Global SSL ICA G2. No Certificate Transparency information was supplied by the server.

Certificate information

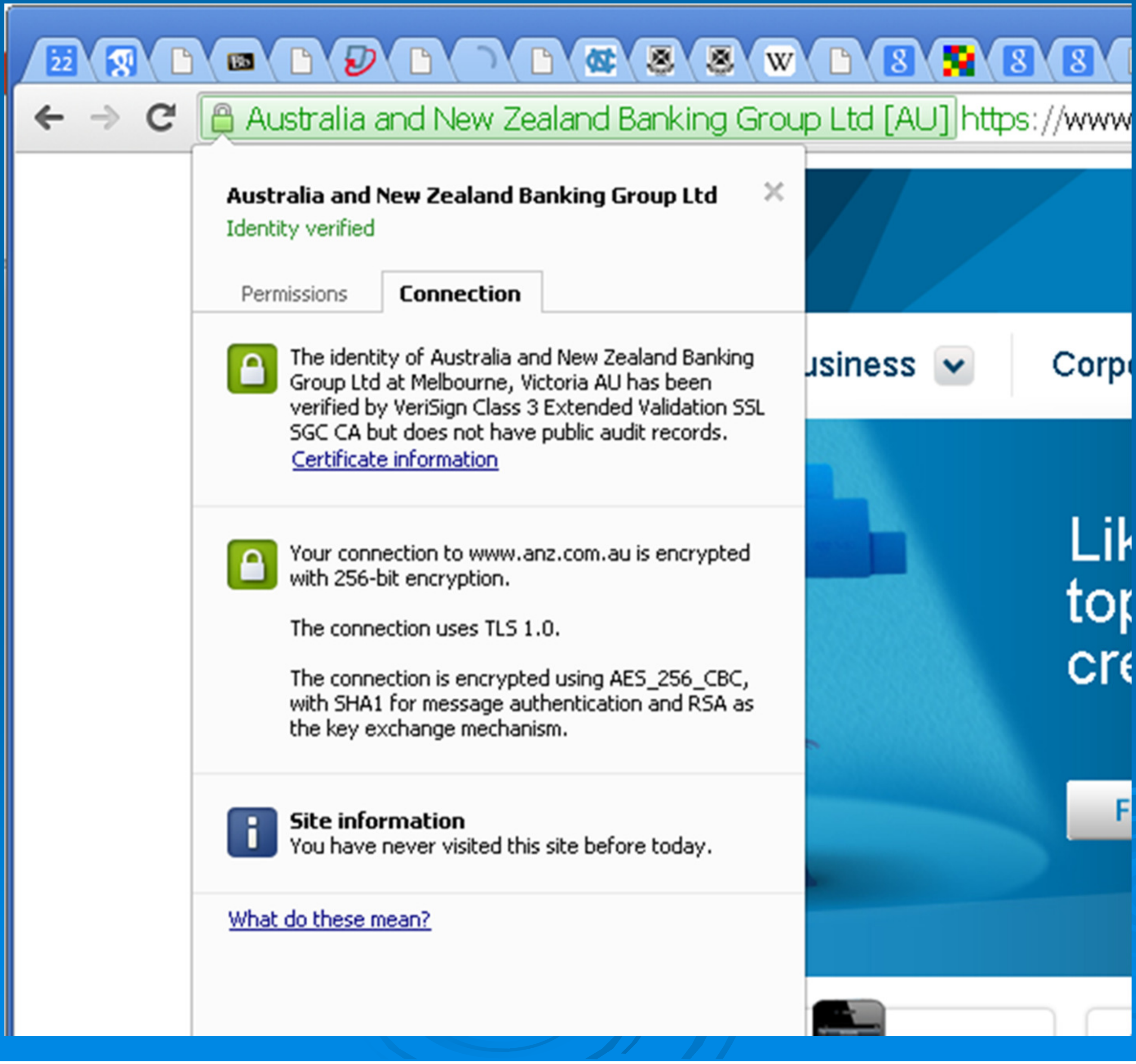
Your connection to www.uq.edu.au is encrypted using an obsolete cipher suite.

The connection uses TLS 1.0

The connection is encrypted using 3DES_EDE_CBC, with HMAC-SHA1 for message authentication and RSA as the key exchange mechanism.

What do these mean?

Another Example



Australia and New Zealand Banking Group Ltd [AU] https://www
usiness
Corp
Lik
top
cre
F

Australia and New Zealand Banking Group Ltd

Identity verified

The identity of Australia and New Zealand Banking Group Ltd at Melbourne, Victoria AU has been verified by VeriSign Class 3 Extended Validation SSL SGC CA but does not have public audit records.

Certificate information

Your connection to www.anz.com.au is encrypted with 256-bit encryption.

The connection uses TLS 1.0.

The connection is encrypted using AES_256_CBC, with SHA1 for message authentication and RSA as the key exchange mechanism.

Site information

You have never visited this site before today.

What do these mean?

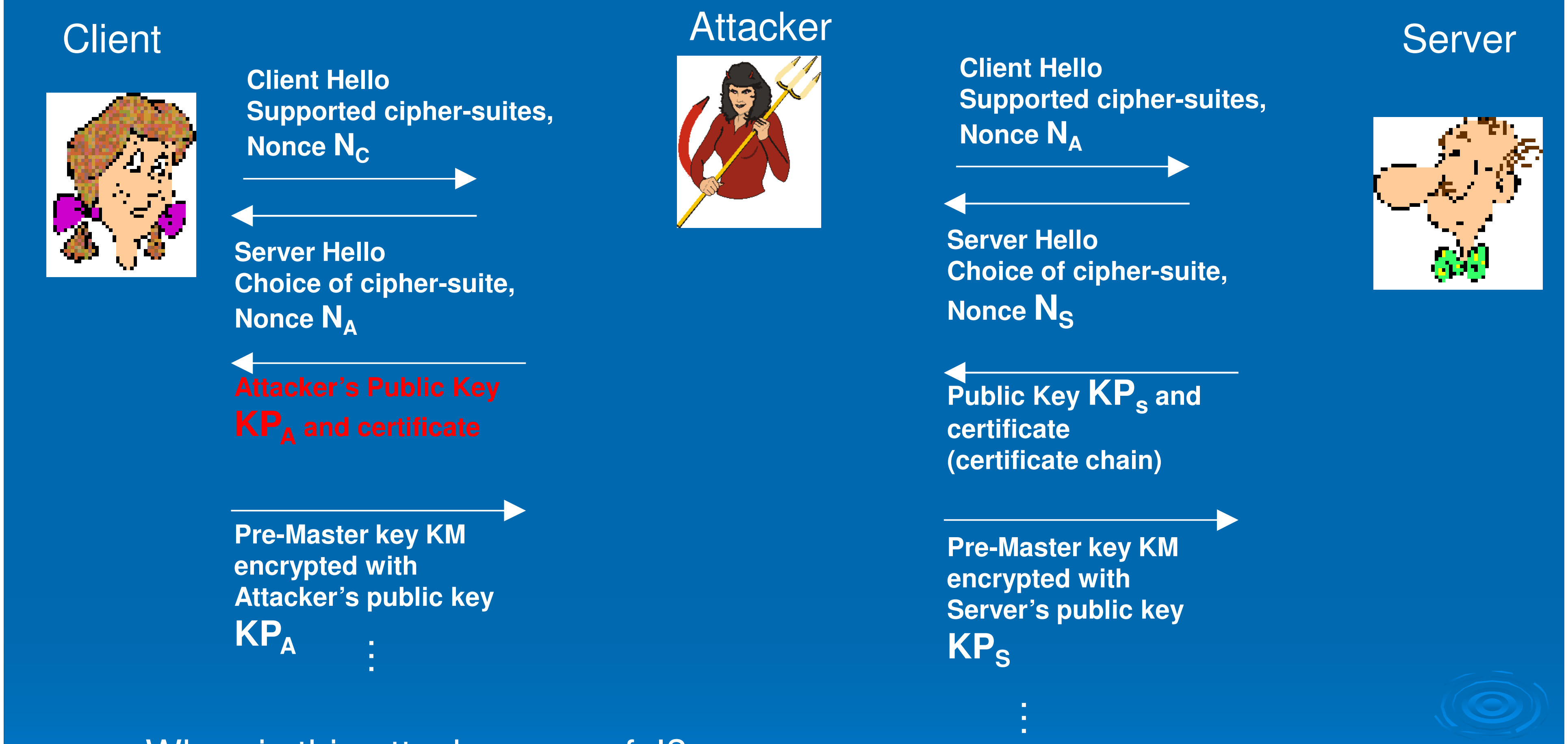
Client Authentication

- For example, in a Web banking scenario using SSL
  - How does the Server authenticate to the client?
    - Via Server Certificate
  - How does the Client/user authenticate to the server?
    - Authentication via certificate is optional in TLS, and is not used for web browsing
    - Via login/password
    - Sent over secure SSL connection

SSL - Man in the middle attack?

- Assume an attacker can redirect a web browsers request to a proxy that he/she controls
  - DNS poisoning
  - ARP spoofing
- How is it possible to launch a Man-in-the-middle attack against SSL/TLS, i.e. the handshake?
  - If DH_anon (plain Diffie-Hellman) is used for key establishment
  - What if RSA Public Key certificates are used in the handshake?

SSL MITM Attack



Client

Attacker

Server

Client Hello
Supported cipher-suites,
Nonce N_C

Server Hello
Choice of cipher-suite,
Nonce N_A

Attacker's Public Key
KP_A and certificate

Pre-Master key KM
encrypted with
Attacker's public key
KP_A

⋮

Client Hello
Supported cipher-suites,
Nonce N_A

Server Hello
Choice of cipher-suite,
Nonce N_S

Public Key KP_S and
certificate
(certificate chain)

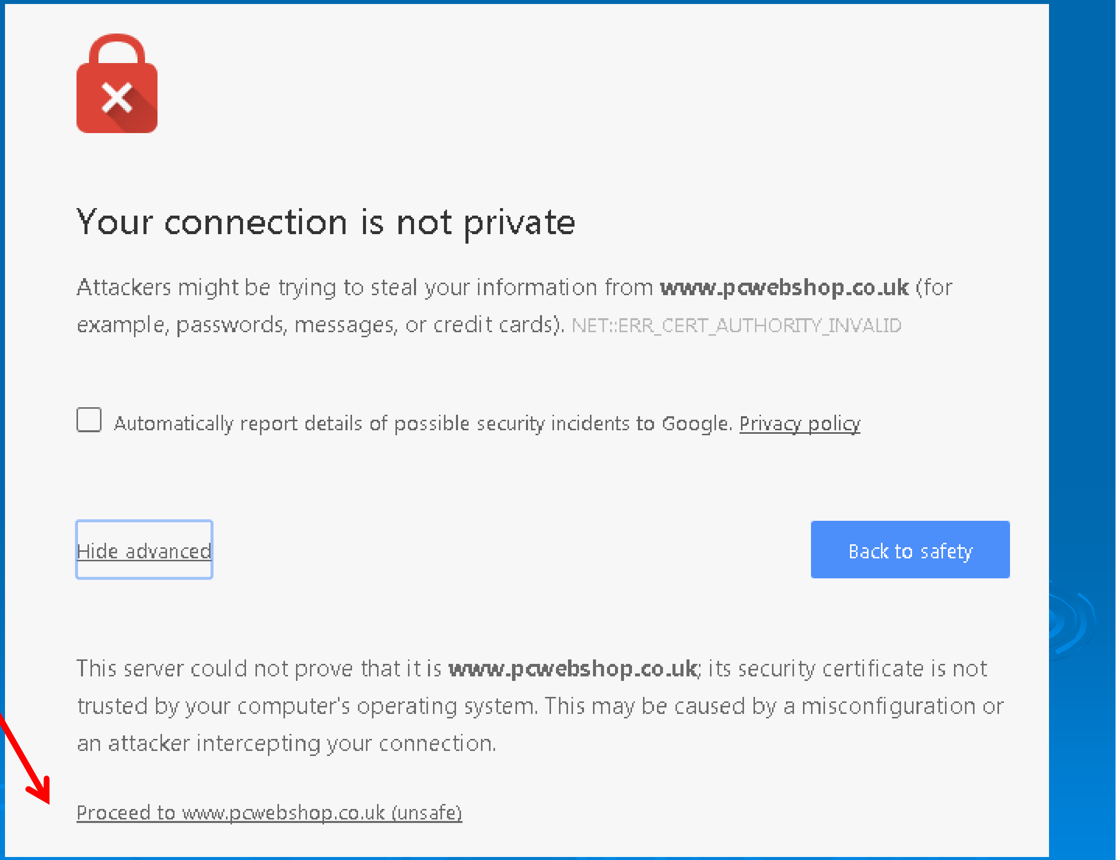
Pre-Master key KM
encrypted with
Server's public key
KP_S

⋮

- When is this attack successful?
  - Attacker needs a way to redirect client's request
    - DNS poisoning, ARP spoofing, …
  - Attack is successful if client accepts attacker's Server certificate
    - Issued by trusted CA?
    - Does name on certificate match the name of the company we want to communicate with?

MITM Attack

- Successful if
  - user clicks here



Your connection is not private

Attackers might be trying to steal your information from **www.pcwebshop.co.uk** (for example, passwords, messages, or credit cards). NET::ERR_CERT_AUTHORITY_INVALID

☐ Automatically report details of possible security incidents to Google. Privacy policy

This server could not prove that it is **www.pcwebshop.co.uk**; its security certificate is not trusted by your computer's operating system. This may be caused by a misconfiguration or an attacker intercepting your connection.

Proceed to www.pcwebshop.co.uk (unsafe)

http://en.wikipedia.org/wiki/Man-in-the-middle_attack

Recently detected Vulnerabilities in SSL/TLS

- Heartbleed
  - Implementation bug in OpenSSL
    - Missing bounds check in TLS heartbeat extension
  - http://heartbleed.com/
  - http://xkcd.com/1354/
- POODLE
  - Design bug in SSL3.0
  - Problem with padding used with CBC cipher mode
    - Problem: Even though not many browsers use SSL3.0, they happily downgrade during handshake (Man-in-the-middle 'version rollback attack')
  - http://arstechnica.com/security/2014/10/ssl-broken-again-in-poodle-attack/

- …
- Good survey of TLS attacks
  - http://en.wikipedia.org/wiki/Transport_Layer_Security

TLS Security

- If you are not concerned about TLS security yet, there is some more:
  - http://www.youtube.com/watch?v=ibF36Yyeehw
  - (This link is not examined material)

Firewalls

(a PCI DSS view)

Firewalls

- But first a brief intro to network security:
- https://www.youtube.com/watch?v=Dat4eUuiWag

- Very old but basic intro to network security and firewalls for non-IT students

Monitoring, Logs, Audit Trails, and Security Information and Event Management (SIEM)

Simon O’Brien and Amanda Lugton

Splunk

(See Echo recording)