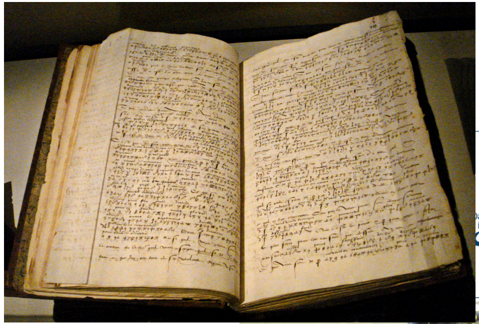
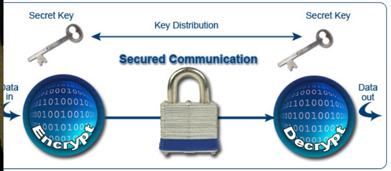
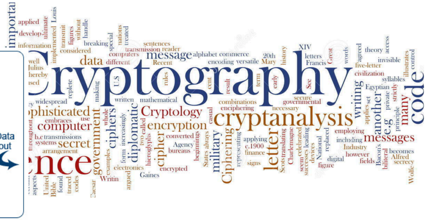
ECDSA

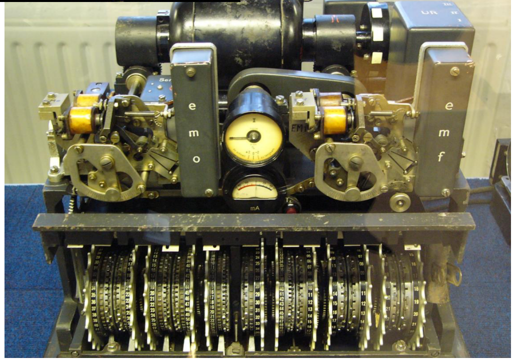
Geoff Huston
APNIC

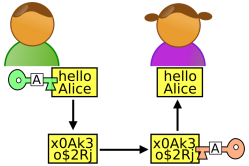
Its all about cryptography











The Basic Challenge

Pick a pair of keys such that:

- Messages encoded with one key can only be decoded with the other key
- Knowledge of the value of one key does not infer the value of the other key

The Power of Primes

$$(m^e)^d \equiv m \pmod{n}$$

As long as *d* and *n* are relatively large, and *n* is the product of two large prime numbers, then finding the value of *d* when you already know the values of *e* and *n* is computationally expensive

The Power of Primes

$$(m^e)^d \equiv m \pmod{n}$$

As long as *d* and *n* are relatively large, and *n* is the
product of two large prime numbers, then
finding the value of *d* when you already know
the values of *e* and *n* is computationally
expensive

But computers get larger and faster - what was infeasible yesterday may be possible tomorrow

The Power of Primes

$$(m^e)^d \equiv m \pmod{n}$$

As long as *d* and *n* are relatively large, and *n* is the product of two large prime numbers, then finding the value of *d* when you already know the values of *e* and *n* is computationally expensive

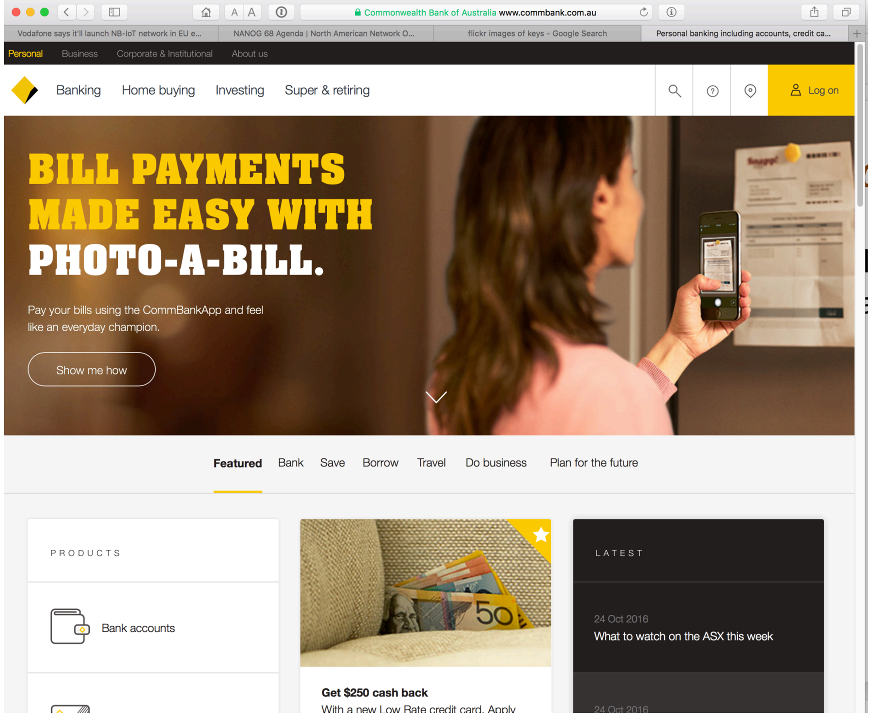
But computers get larger and faster - what was infeasible yesterday may be possible tomorrow

The way to stay ahead is to make the value of n larger and larger

Why is this important?

Because much of the foundation of internet Security rests upon this relationship

My Bank...(I hope!)



BILL PAYMENTS MADE EASY WITH PHOTO-A-BILL.

Pay your bills using the CommBankApp and feel like an everyday champion.

Get $250 cash back

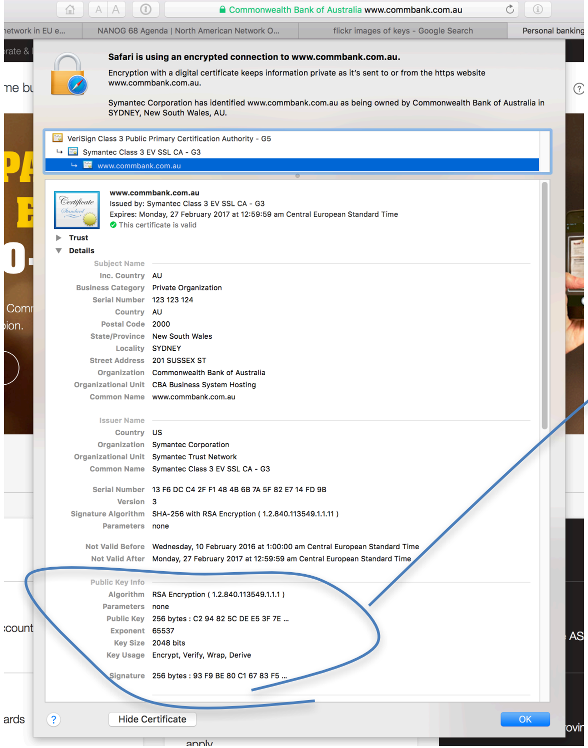
With a new Low Rate credit card. Apply

24 Oct 2016

What to watch on the ASX this week

24 Oct 2016

My Bank's Digital Signature



Safari is using an encrypted connection to www.commbank.com.au.

Encryption with a digital certificate keeps information private as it's sent to or from the https website www.commbank.com.au.

Symantec Corporation has identified www.commbank.com.au as being owned by Commonwealth Bank of Australia in SYDNEY, New South Wales, AU.

VeriSign Class 3 Public Primary Certification Authority - G5
- Symantec Class 3 EV SSL CA - G3
  - www.commbank.com.au

www.commbank.com.au
Issued by: Symantec Class 3 EV SSL CA - G3
Expires: Monday, 27 February 2017 at 12:59:59 am Central European Standard Time
✓ This certificate is valid

▶ **Trust**

▼ **Details**

Subject Name	
Inc. Country	AU
Business Category	Private Organization
Serial Number	123 123 124
Country	AU
Postal Code	2000
State/Province	New South Wales
Locality	SYDNEY
Street Address	201 SUSSEX ST
Organization	Commonwealth Bank of Australia
Organizational Unit	CBA Business System Hosting
Common Name	www.commbank.com.au
Issuer Name	
Country	US
Organization	Symantec Corporation
Organizational Unit	Symantec Trust Network
Common Name	Symantec Class 3 EV SSL CA - G3
Serial Number	13 F6 DC C4 2F F1 48 4B 6B 7A 5F 82 E7 14 FD 9B
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters	none
Not Valid Before	Wednesday, 10 February 2016 at 1:00:00 am Central European Standard Time
Not Valid After	Monday, 27 February 2017 at 12:59:59 am Central European Standard Time
Public Key Info	
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	none
Public Key	256 bytes : C2 94 82 5C DE E5 3F 7E ...
Exponent	65537
Key Size	2048 bits
Key Usage	Encrypt, Verify, Wrap, Derive
Signature	256 bytes : 93 F9 BE 80 C1 67 83 F5 ...

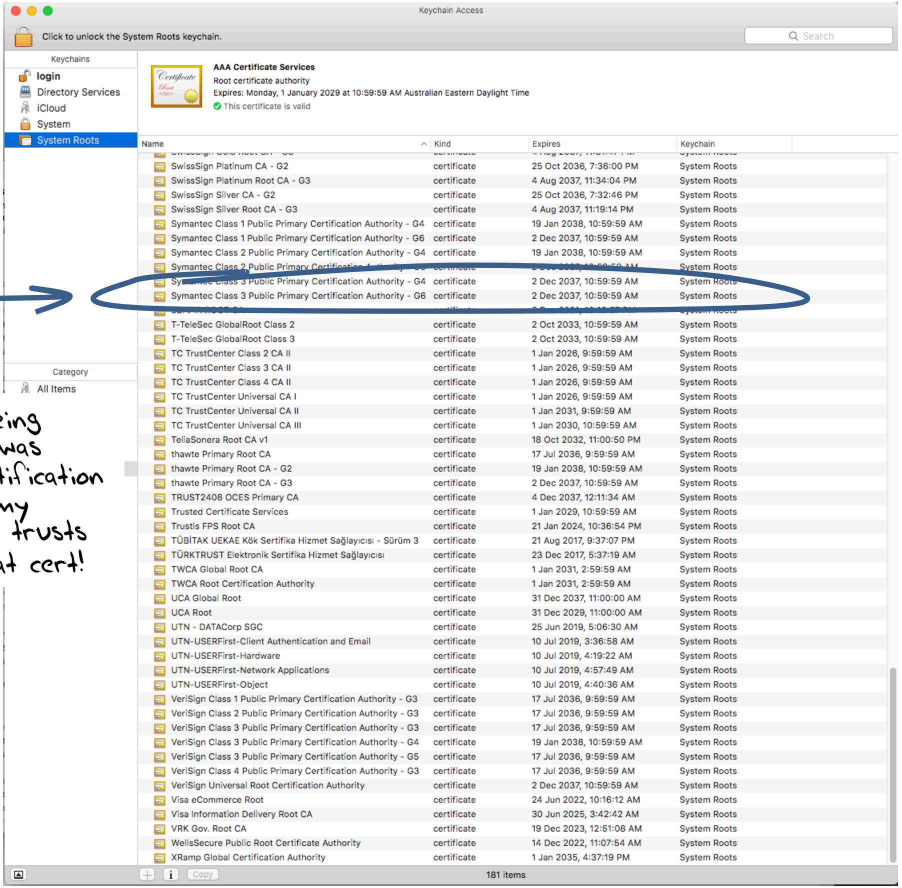
Yes, that says RSA using a 2048-bit Public key

Domain Name Certification

- The Commonwealth Bank of Australia has generated a key pair
- And they passed a certificate signing request to a company called “Symantec”
- Who is willing to vouch (in a certificate) that the entity who goes by the domain name of www.commbank.com.au also has a certain public key value
- So if I can associate this public key with a connection then I have a high degree of confidence that I’ve connected to www.commbank.com.au, as long as I am prepared to trust Symantec and the certificates that they issue

Why should i trust them?

Local Trust



Click to unlock the System Roots keychain.

AAA Certificate Services
Root certificate authority
Expires: Monday, 1 January 2029 at 10:59:59 AM Australian Eastern Daylight Time
✓ This certificate is valid

Name	Kind	Expires	Keychain
SwissSign Platinum CA - G2	certificate	25 Oct 2036, 7:36:00 PM	System Roots
SwissSign Platinum Root CA - G3	certificate	4 Aug 2037, 11:34:04 PM	System Roots
SwissSign Silver CA - G2	certificate	25 Oct 2036, 7:32:46 PM	System Roots
SwissSign Silver Root CA - G3	certificate	4 Aug 2037, 11:19:14 PM	System Roots
Symantec Class 1 Public Primary Certification Authority - G4	certificate	19 Jan 2038, 10:59:59 AM	System Roots
Symantec Class 1 Public Primary Certification Authority - G6	certificate	2 Dec 2037, 10:59:59 AM	System Roots
Symantec Class 2 Public Primary Certification Authority - G4	certificate	19 Jan 2038, 10:59:59 AM	System Roots
Symantec Class 2 Public Primary Certification Authority - G6	certificate	2 Dec 2037, 10:59:59 AM	System Roots
Symantec Class 3 Public Primary Certification Authority - G4	certificate	2 Dec 2037, 10:59:59 AM	System Roots
Symantec Class 3 Public Primary Certification Authority - G6	certificate	2 Dec 2037, 10:59:59 AM	System Roots
[illegible]	certificate	[illegible]	System Roots
T-TeleSec GlobalRoot Class 2	certificate	2 Oct 2033, 10:59:59 AM	System Roots
T-TeleSec GlobalRoot Class 3	certificate	2 Oct 2033, 10:59:59 AM	System Roots
TC TrustCenter Class 2 CA II	certificate	1 Jan 2026, 9:59:59 AM	System Roots
TC TrustCenter Class 3 CA II	certificate	1 Jan 2026, 9:59:59 AM	System Roots
TC TrustCenter Class 4 CA II	certificate	1 Jan 2026, 9:59:59 AM	System Roots
TC TrustCenter Universal CA I	certificate	1 Jan 2026, 9:59:59 AM	System Roots
TC TrustCenter Universal CA II	certificate	1 Jan 2031, 9:59:59 AM	System Roots
TC TrustCenter Universal CA III	certificate	1 Jan 2030, 10:59:59 AM	System Roots
TeliaSonera Root CA v1	certificate	18 Oct 2032, 11:00:50 PM	System Roots
thawte Primary Root CA	certificate	17 Jul 2036, 9:59:59 AM	System Roots
thawte Primary Root CA - G2	certificate	19 Jan 2038, 10:59:59 AM	System Roots
thawte Primary Root CA - G3	certificate	2 Dec 2037, 10:59:59 AM	System Roots
TRUST2408 OCES Primary CA	certificate	4 Dec 2037, 12:11:34 AM	System Roots
Trusted Certificate Services	certificate	1 Jan 2029, 10:59:59 AM	System Roots
Trustis FPS Root CA	certificate	21 Jan 2024, 10:36:54 PM	System Roots
TÜBİTAK UEKAE Kök Sertifika Hizmet Sağlayıcısı - Sürüm 3	certificate	21 Aug 2017, 9:37:07 PM	System Roots
TÜRKTRUST Elektronik Sertifika Hizmet Sağlayıcısı	certificate	23 Dec 2017, 5:37:19 AM	System Roots
TWCA Global Root CA	certificate	1 Jan 2031, 2:59:59 AM	System Roots
TWCA Root Certification Authority	certificate	1 Jan 2031, 2:59:59 AM	System Roots
UCA Global Root	certificate	31 Dec 2037, 11:00:00 AM	System Roots
UCA Root	certificate	31 Dec 2029, 11:00:00 AM	System Roots
UTN - DATACorp SGC	certificate	25 Jun 2019, 5:06:30 AM	System Roots
UTN-USERFirst-Client Authentication and Email	certificate	10 Jul 2019, 3:36:58 AM	System Roots
UTN-USERFirst-Hardware	certificate	10 Jul 2019, 4:19:22 AM	System Roots
UTN-USERFirst-Network Applications	certificate	10 Jul 2019, 4:57:49 AM	System Roots
UTN-USERFirst-Object	certificate	10 Jul 2019, 4:40:36 AM	System Roots
VeriSign Class 1 Public Primary Certification Authority - G3	certificate	17 Jul 2036, 9:59:59 AM	System Roots
VeriSign Class 2 Public Primary Certification Authority - G3	certificate	17 Jul 2036, 9:59:59 AM	System Roots
VeriSign Class 3 Public Primary Certification Authority - G3	certificate	17 Jul 2036, 9:59:59 AM	System Roots
VeriSign Class 3 Public Primary Certification Authority - G4	certificate	19 Jan 2038, 10:59:59 AM	System Roots
VeriSign Class 3 Public Primary Certification Authority - G5	certificate	17 Jul 2036, 9:59:59 AM	System Roots
VeriSign Class 4 Public Primary Certification Authority - G3	certificate	17 Jul 2036, 9:59:59 AM	System Roots
VeriSign Universal Root Certification Authority	certificate	2 Dec 2037, 10:59:59 AM	System Roots
Visa eCommerce Root	certificate	24 Jun 2022, 10:16:12 AM	System Roots
Visa Information Delivery Root CA	certificate	30 Jun 2025, 3:42:42 AM	System Roots
VRK Gov. Root CA	certificate	19 Dec 2023, 12:51:08 AM	System Roots
WellsSecure Public Root Certificate Authority	certificate	14 Dec 2022, 11:07:54 AM	System Roots
XRamp Global Certification Authority	certificate	1 Jan 2035, 4:37:19 PM	System Roots

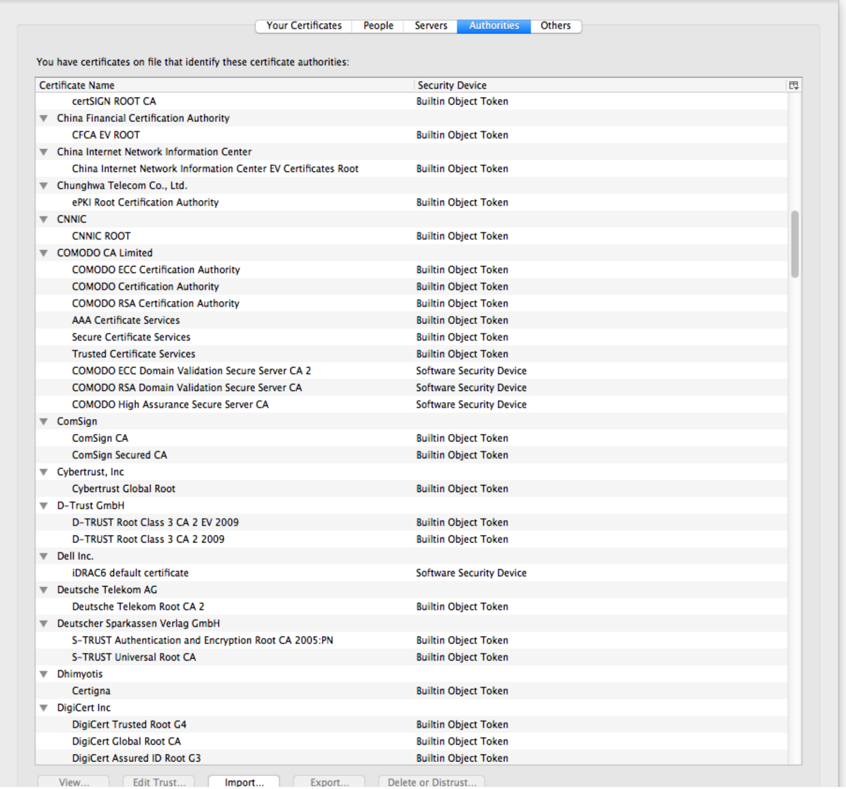
181 items

The cert i'm being
asked to trust was
issued by a certification
authority that my
browser already trusts
– so i trust that cert!

Local Trust

That’s a big list of people to Trust

Are they all trustable?



You have certificates on file that identify these certificate authorities:

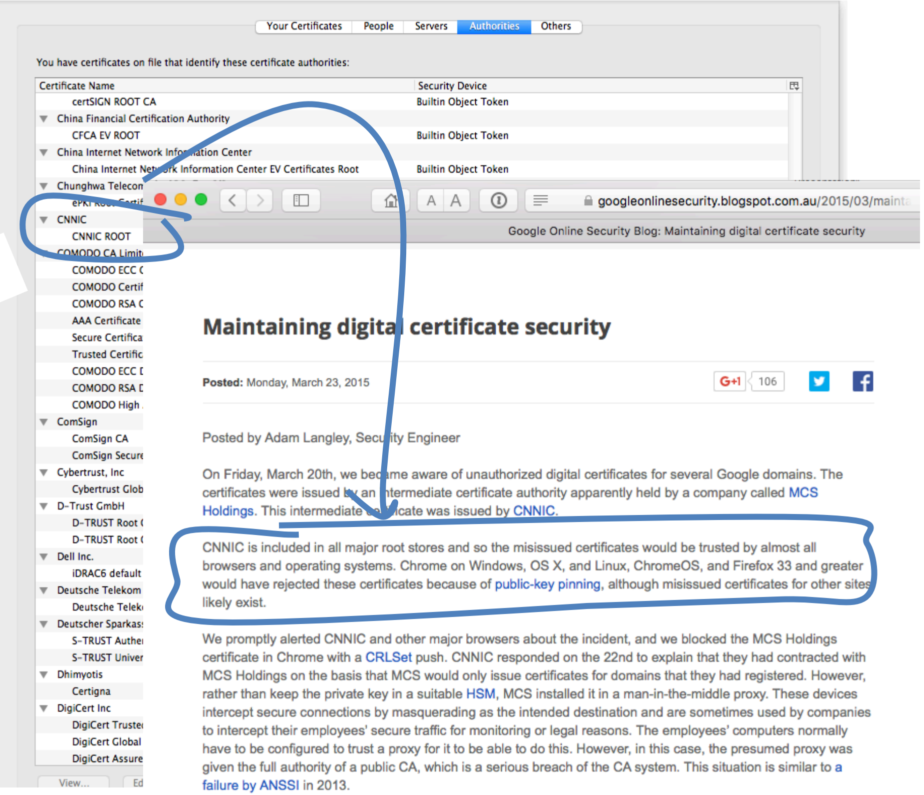
Certificate Name	Security Device
certSIGN ROOT CA	Builtin Object Token
▼ China Financial Certification Authority	
CFCA EV ROOT	Builtin Object Token
▼ China Internet Network Information Center	
China Internet Network Information Center EV Certificates Root	Builtin Object Token
▼ Chunghwa Telecom Co., Ltd.	
ePKI Root Certification Authority	Builtin Object Token
▼ CNNIC	
CNNIC ROOT	Builtin Object Token
▼ COMODO CA Limited	
COMODO ECC Certification Authority	Builtin Object Token
COMODO Certification Authority	Builtin Object Token
COMODO RSA Certification Authority	Builtin Object Token
AAA Certificate Services	Builtin Object Token
Secure Certificate Services	Builtin Object Token
Trusted Certificate Services	Builtin Object Token
COMODO ECC Domain Validation Secure Server CA 2	Software Security Device
COMODO RSA Domain Validation Secure Server CA	Software Security Device
COMODO High Assurance Secure Server CA	Software Security Device
▼ ComSign	
ComSign CA	Builtin Object Token
ComSign Secured CA	Builtin Object Token
▼ Cybertrust, Inc	
Cybertrust Global Root	Builtin Object Token
▼ D-Trust GmbH	
D-TRUST Root Class 3 CA 2 EV 2009	Builtin Object Token
D-TRUST Root Class 3 CA 2 2009	Builtin Object Token
▼ Dell Inc.	
iDRAC6 default certificate	Software Security Device
▼ Deutsche Telekom AG	
Deutsche Telekom Root CA 2	Builtin Object Token
▼ Deutscher Sparkassen Verlag GmbH	
S-TRUST Authentication and Encryption Root CA 2005:PN	Builtin Object Token
S-TRUST Universal Root CA	Builtin Object Token
▼ Dhimyotis	
Certigna	Builtin Object Token
▼ DigiCert Inc	
DigiCert Trusted Root G4	Builtin Object Token
DigiCert Global Root CA	Builtin Object Token
DigiCert Assured ID Root G3	Builtin Object Token

Local Trust

That’s a big list of people to Trust

Are they all trustable?

Evidently Not!



You have certificates on file that identify these certificate authorities:

Certificate Name	Security Device
certSIGN ROOT CA	Builtin Object Token
China Financial Certification Authority	
CFCA EV ROOT	Builtin Object Token
China Internet Network Information Center	
China Internet Network Information Center EV Certificates Root	Builtin Object Token
Chunghwa Telecom	
ePKI Root Certif	
CNNIC	
CNNIC ROOT	
COMODO CA Limit	
COMODO ECC C	
COMODO Certif	
COMODO RSA C	
AAA Certificate	
Secure Certifica	
Trusted Certific	
COMODO ECC D	
COMODO RSA D	
COMODO High	
ComSign	
ComSign CA	
ComSign Secure	
Cybertrust, Inc	
Cybertrust Glob	
D-Trust GmbH	
D-TRUST Root C	
D-TRUST Root C	
Dell Inc.	
iDRAC6 default	
Deutsche Telekom	
Deutsche Telek	
Deutscher Sparkas	
S-TRUST Authen	
S-TRUST Univer	
Dhimyotis	
Certigna	
DigiCert Inc	
DigiCert Truste	
DigiCert Global	
DigiCert Assure	

googleonlinesecurity.blogspot.com.au/2015/03/mainta

Google Online Security Blog: Maintaining digital certificate security

Maintaining digital certificate security

Posted: Monday, March 23, 2015

G+1 106

Posted by Adam Langley, Security Engineer

On Friday, March 20th, we became aware of unauthorized digital certificates for several Google domains. The certificates were issued by an intermediate certificate authority apparently held by a company called MCS Holdings. This intermediate certificate was issued by CNNIC.

CNNIC is included in all major root stores and so the misissued certificates would be trusted by almost all browsers and operating systems. Chrome on Windows, OS X, and Linux, ChromeOS, and Firefox 33 and greater would have rejected these certificates because of public-key pinning, although misissued certificates for other sites likely exist.

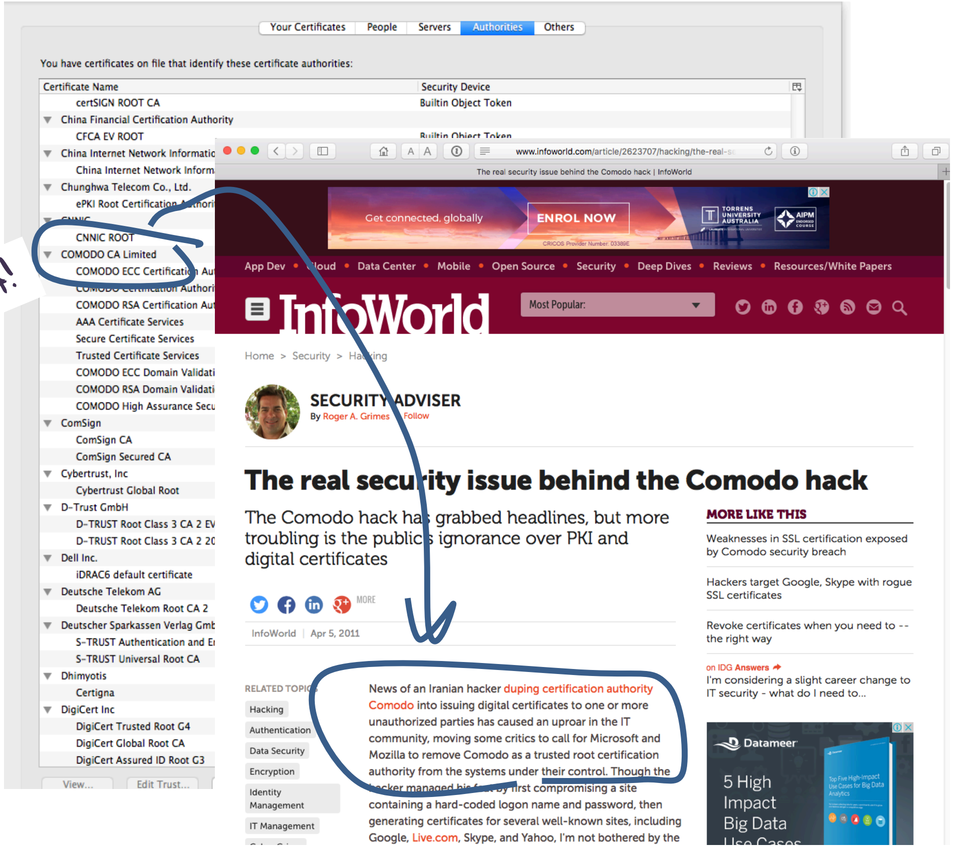
We promptly alerted CNNIC and other major browsers about the incident, and we blocked the MCS Holdings certificate in Chrome with a CRLSet push. CNNIC responded on the 22nd to explain that they had contracted with MCS Holdings on the basis that MCS would only issue certificates for domains that they had registered. However, rather than keep the private key in a suitable HSM, MCS installed it in a man-in-the-middle proxy. These devices intercept secure connections by masquerading as the intended destination and are sometimes used by companies to intercept their employees’ secure traffic for monitoring or legal reasons. The employees’ computers normally have to be configured to trust a proxy for it to be able to do this. However, in this case, the presumed proxy was given the full authority of a public CA, which is a serious breach of the CA system. This situation is similar to a failure by ANSSI in 2013.

Local Trust

That's a big list of people to Trust

Are they all trustable?

Evidently Not!



You have certificates on file that identify these certificate authorities:

Certificate Name	Security Device
certSIGN ROOT CA	Builtin Object Token
China Financial Certification Authority	
CFCA EV ROOT	Builtin Object Token
China Internet Network Informatio	
China Internet Network Inform	
Chunghwa Telecom Co., Ltd.	
ePKI Root Certification Authori	
CNNIC	
CNNIC ROOT	
COMODO CA Limited	
COMODO ECC Certification Au	
COMODO Certification Authori	
COMODO RSA Certification Au	
AAA Certificate Services	
Secure Certificate Services	
Trusted Certificate Services	
COMODO ECC Domain Validati	
COMODO RSA Domain Validati	
COMODO High Assurance Secu	
ComSign	
ComSign CA	
ComSign Secured CA	
Cybertrust, Inc	
Cybertrust Global Root	
D-Trust GmbH	
D-TRUST Root Class 3 CA 2 EV	
D-TRUST Root Class 3 CA 2 20	
Dell Inc.	
iDRAC6 default certificate	
Deutsche Telekom AG	
Deutsche Telekom Root CA 2	
Deutscher Sparkassen Verlag Gmb	
S-TRUST Authentication and E	
S-TRUST Universal Root CA	
Dhimyotis	
Certigna	
DigiCert Inc	
DigiCert Trusted Root G4	
DigiCert Global Root CA	
DigiCert Assured ID Root G3	

SECURITY ADVISER
By Roger A. Grimes Follow

The real security issue behind the Comodo hack

The Comodo hack has grabbed headlines, but more troubling is the public's ignorance over PKI and digital certificates

InfoWorld | Apr 5, 2011

News of an Iranian hacker duping certification authority Comodo into issuing digital certificates to one or more unauthorized parties has caused an uproar in the IT community, moving some critics to call for Microsoft and Mozilla to remove Comodo as a trusted root certification authority from the systems under their control. Though the hacker managed his feat by first compromising a site containing a hard-coded logon name and password, then generating certificates for several well-known sites, including Google, Live.com, Skype, and Yahoo, I'm not bothered by the

MORE LIKE THIS

Weaknesses in SSL certification exposed by Comodo security breach

Hackers target Google, Skype with rogue SSL certificates

Revoke certificates when you need to -- the right way

on IDG Answers
I'm considering a slight career change to IT security - what do I need to...

With unpleasant consequences when it all goes wrong

With unpleasant consequences when it all goes wrong

Société Générale, BNP Paribas and Crédit Agricole, are considered integral actors in the French economy, lending

VOLATILITY IS THE NEW MARKET NORM
Large swings in share prices are more common now than at any other time in recent stock market history. *PAGE 16*

Iranian activists feel the chill as hacker taps into e-mails

BY SOMINI SENGUPTA

He claims to be 21 years old, a student of software engineering in Tehran who reveres Ayatollah Ali Khamenei and despises dissidents in his country.

He sneaked into the computer systems of a security firm on the outskirts of Amsterdam. He created fake credentials that could allow someone to spy on Internet connections that appeared to be secure. He then shared that bounty with people he declines to identify.

The fruits of his labor are believed to [illegible] to tap into the online [illegible] many as 300,000 [illegible] summer. online security mechanism that is trusted by Internet users all over the world.

Comodohacker, as he calls himself, insists that he acted on his own and is unperturbed by the notion that his work might have been used to spy on anti-government compatriots.

"I'm totally independent," he said in an e-mail exchange with The New York Times. "I just share my findings with some people in Iran. They are free to do anything they want with my findings and things I share with them, but I'm not responsible."

In the annals of Internet attacks, this is most likely to go down as a moment of reckoning. For activists, it shows the

HACKER, PAGE

International Herald Tribune
Sep 13, 2011 Front Page

What's going wrong here?

- The TLS handshake cannot specify WHICH CA should be used to validate the digital certificate
- Your browser will allow ANY CA to be used to validate a certificate

What's going wrong here?

- The TLS handshake cannot specify WHICH CA should be used to validate the digital
- Your brow allow ANY CA to be used to validate a certificate

WOW! That's awesomely bad!

What's going wrong here?

- The TLS handshake cannot specify WHICH CA should be used to validate the digital
- Your brow allow ANY CA to be used to validate a certificate

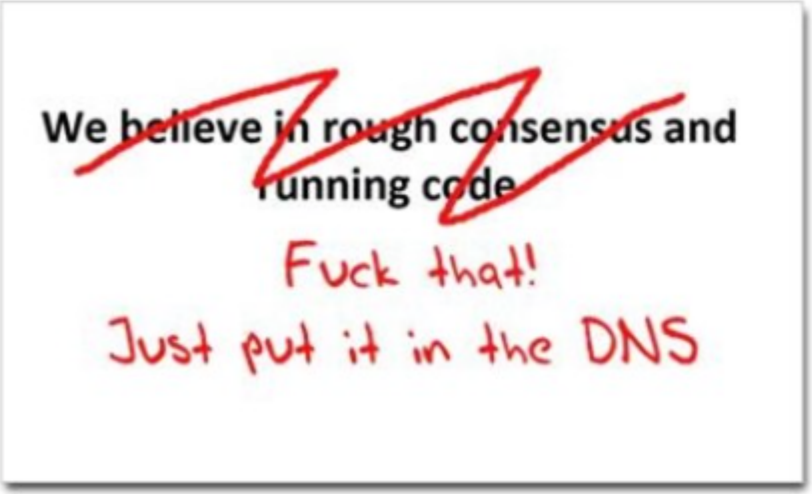
WOW! That's awesomely bad!

Here's a lock – it might be the lock on your front door for all i know.

The lock might LOOK secure, but don't worry – literally ANY key can open it!

Where now?

Lets use the DNS!



~~We believe in rough consensus and running code~~

Fuck that!

Just put it in the DNS

cafepress.com/nxdomain

Seriously

Where better to find out the public key associated with a DNS name than to look it up in the DNS?

Seriously

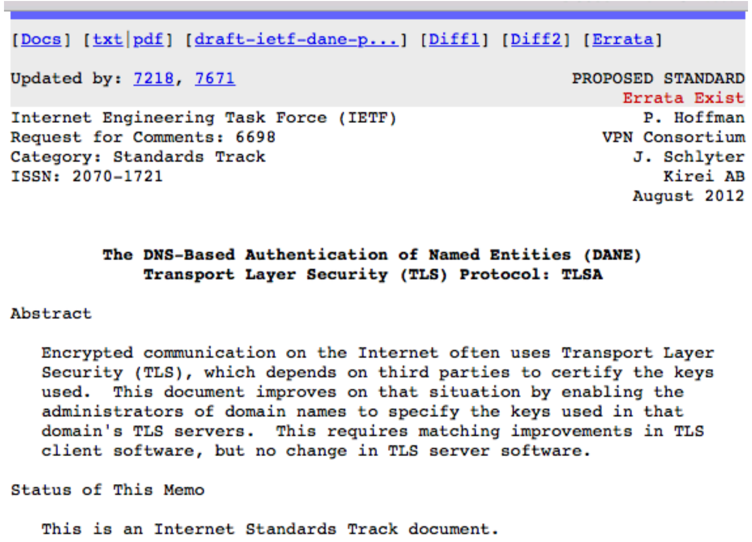
Where better to find out the public key associated with a DNS name than to look it up in the DNS?

- Why not query the DNS for the HSTS record?
- Why not query the DNS for the issu
- Why not query the DNS for t ... or the domain name cert?
- Why not query the D ... he domain name public key cert as a simple self-si ...

Who needs CA's anyway?

DANE

- Using the DNS to associated domain name public key certificates with domain name



[Docs] [txt|pdf] [draft-ietf-dane-p...] [Diff1] [Diff2] [Errata]

Updated by: 7218, 7671 PROPOSED STANDARD
Errata Exist

Internet Engineering Task Force (IETF) P. Hoffman
Request for Comments: 6698 VPN Consortium
Category: Standards Track J. Schlyter
ISSN: 2070-1721 Kirei AB
August 2012

The DNS-Based Authentication of Named Entities (DANE)
Transport Layer Security (TLS) Protocol: TLSA

Abstract

Encrypted communication on the Internet often uses Transport Layer Security (TLS), which depends on third parties to certify the keys used. This document improves on that situation by enabling the administrators of domain names to specify the keys used in that domain's TLS servers. This requires matching improvements in TLS client software, but no change in TLS server software.

Status of This Memo

This is an Internet Standards Track document.

TLS with DANE

- Client receives server cert in Server Hello
  - *Client lookups the DNS for the TLSA Resource Record of the domain name*
  - *Client validates the presented certificate against the TLSA RR*
- Client performs Client Key exchange

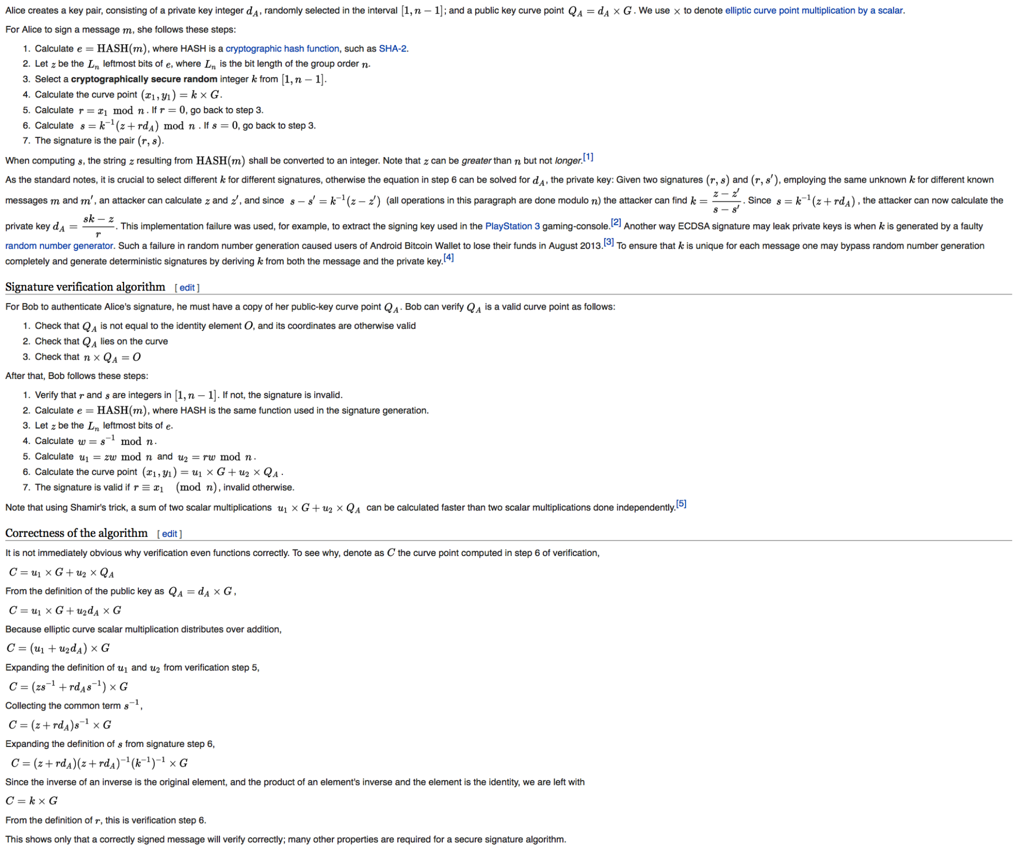
The search for small keys

BUT

- Large keys and the DNS don’t mix very well:
  - Either we try and make UDP fragmentation work reliably (for once!)
  - Or we switch the DNS to use TCP

- Neither option sounds like fun!
- So can we defer the crunch time for a while?

Enter Elliptical Curves



Alice creates a key pair, consisting of a private key integer d_A, randomly selected in the interval $[1, n-1]$; and a public key curve point $Q_A = d_A \times G$. We use $\times$ to denote elliptic curve point multiplication by a scalar.

For Alice to sign a message m, she follows these steps:

1. Calculate $e = \mathrm{HASH}(m)$, where HASH is a cryptographic hash function, such as SHA-2.
2. Let z be the L_n leftmost bits of e, where L_n is the bit length of the group order n.
3. Select a **cryptographically secure random** integer k from $[1, n-1]$.
4. Calculate the curve point $(x_1, y_1) = k \times G$.
5. Calculate $r = x_1 \mod n$. If $r = 0$, go back to step 3.
6. Calculate $s = k^{-1}(z + rd_A) \mod n$. If $s = 0$, go back to step 3.
7. The signature is the pair (r, s).

When computing s, the string z resulting from $\mathrm{HASH}(m)$ shall be converted to an integer. Note that z can be *greater* than n but not *longer*.[1]

As the standard notes, it is crucial to select different k for different signatures, otherwise the equation in step 6 can be solved for d_A, the private key: Given two signatures (r, s) and (r, s'), employing the same unknown k for different known messages m and m', an attacker can calculate z and z', and since $s - s' = k^{-1}(z - z')$ (all operations in this paragraph are done modulo n) the attacker can find $k = \frac{z - z'}{s - s'}$. Since $s = k^{-1}(z + rd_A)$, the attacker can now calculate the private key $d_A = \frac{sk - z}{r}$. This implementation failure was used, for example, to extract the signing key used in the PlayStation 3 gaming-console.[2] Another way ECDSA signature may leak private keys is when k is generated by a faulty random number generator. Such a failure in random number generation caused users of Android Bitcoin Wallet to lose their funds in August 2013.[3] To ensure that k is unique for each message one may bypass random number generation completely and generate deterministic signatures by deriving k from both the message and the private key.[4]

Signature verification algorithm [edit]

For Bob to authenticate Alice's signature, he must have a copy of her public-key curve point Q_A. Bob can verify Q_A is a valid curve point as follows:

1. Check that Q_A is not equal to the identity element O, and its coordinates are otherwise valid
2. Check that Q_A lies on the curve
3. Check that $n \times Q_A = O$

After that, Bob follows these steps:

1. Verify that r and s are integers in $[1, n-1]$. If not, the signature is invalid.
2. Calculate $e = \mathrm{HASH}(m)$, where HASH is the same function used in the signature generation.
3. Let z be the L_n leftmost bits of e.
4. Calculate $w = s^{-1} \mod n$.
5. Calculate $u_1 = zw \mod n$ and $u_2 = rw \mod n$.
6. Calculate the curve point $(x_1, y_1) = u_1 \times G + u_2 \times Q_A$.
7. The signature is valid if $r \equiv x_1 \pmod n$, invalid otherwise.

Note that using Shamir's trick, a sum of two scalar multiplications $u_1 \times G + u_2 \times Q_A$ can be calculated faster than two scalar multiplications done independently.[5]

Correctness of the algorithm [edit]

It is not immediately obvious why verification even functions correctly. To see why, denote as C the curve point computed in step 6 of verification,

$$C = u_1 \times G + u_2 \times Q_A$$

From the definition of the public key as $Q_A = d_A \times G$,

$$C = u_1 \times G + u_2 d_A \times G$$

Because elliptic curve scalar multiplication distributes over addition,

$$C = (u_1 + u_2 d_A) \times G$$

Expanding the definition of u_1 and u_2 from verification step 5,

$$C = (zs^{-1} + rd_A s^{-1}) \times G$$

Collecting the common term s^{-1},

$$C = (z + rd_A)s^{-1} \times G$$

Expanding the definition of s from signature step 6,

$$C = (z + rd_A)(z + rd_A)^{-1}(k^{-1})^{-1} \times G$$

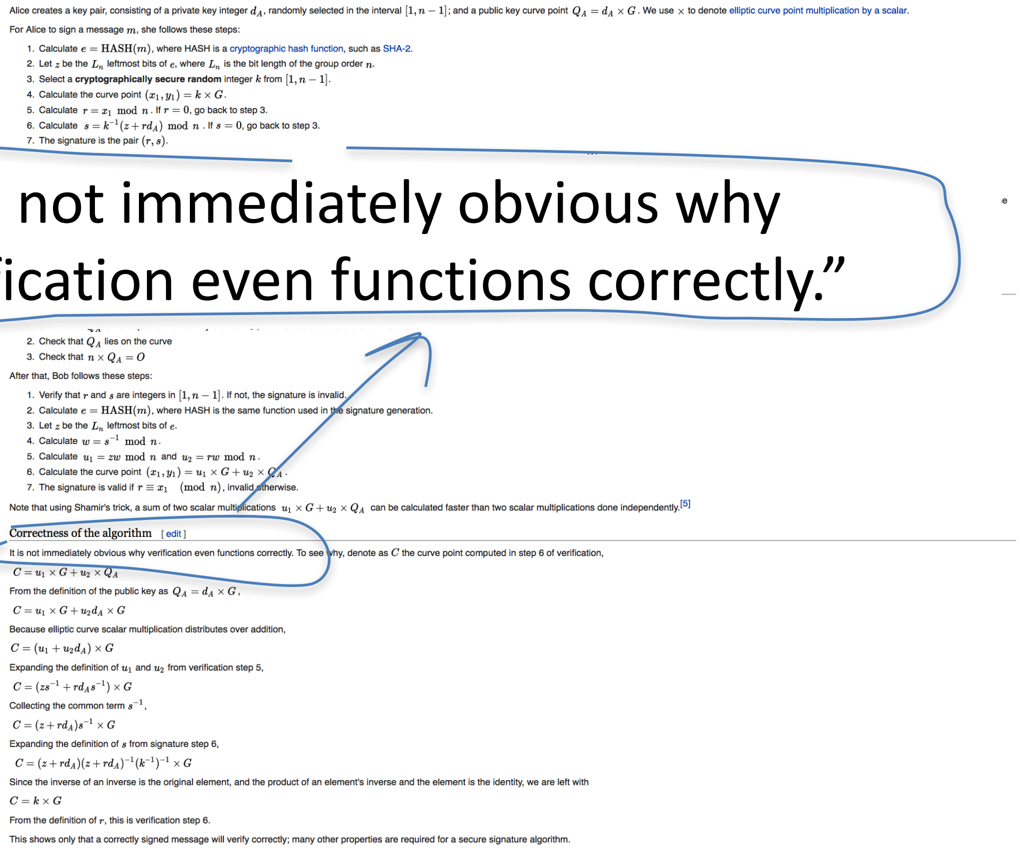
Since the inverse of an inverse is the original element, and the product of an element's inverse and the element is the identity, we are left with

$$C = k \times G$$

From the definition of r, this is verification step 6.

This shows only that a correctly signed message will verify correctly; many other properties are required for a secure signature algorithm.

Enter Elliptical Curves



Alice creates a key pair, consisting of a private key integer d_A, randomly selected in the interval $[1, n-1]$; and a public key curve point $Q_A = d_A \times G$. We use $\times$ to denote elliptic curve point multiplication by a scalar.

For Alice to sign a message m, she follows these steps:

1. Calculate $e = \mathrm{HASH}(m)$, where HASH is a cryptographic hash function, such as SHA-2.
2. Let z be the L_n leftmost bits of e, where L_n is the bit length of the group order n.
3. Select a **cryptographically secure random** integer k from $[1, n-1]$.
4. Calculate the curve point $(x_1, y_1) = k \times G$.
5. Calculate $r = x_1 \mod n$. If $r = 0$, go back to step 3.
6. Calculate $s = k^{-1}(z + rd_A) \mod n$. If $s = 0$, go back to step 3.
7. The signature is the pair (r, s).

“It is not immediately obvious why verification even functions correctly.”

2. Check that Q_A lies on the curve
3. Check that $n \times Q_A = O$

After that, Bob follows these steps:

1. Verify that r and s are integers in $[1, n-1]$. If not, the signature is invalid.
2. Calculate $e = \mathrm{HASH}(m)$, where HASH is the same function used in the signature generation.
3. Let z be the L_n leftmost bits of e.
4. Calculate $w = s^{-1} \mod n$.
5. Calculate $u_1 = zw \mod n$ and $u_2 = rw \mod n$.
6. Calculate the curve point $(x_1, y_1) = u_1 \times G + u_2 \times Q_A$.
7. The signature is valid if $r \equiv x_1 \pmod{n}$, invalid otherwise.

Note that using Shamir's trick, a sum of two scalar multiplications $u_1 \times G + u_2 \times Q_A$ can be calculated faster than two scalar multiplications done independently.[5]

Correctness of the algorithm [edit]

It is not immediately obvious why verification even functions correctly. To see why, denote as C the curve point computed in step 6 of verification,

$$C = u_1 \times G + u_2 \times Q_A$$

From the definition of the public key as $Q_A = d_A \times G$,

$$C = u_1 \times G + u_2 d_A \times G$$

Because elliptic curve scalar multiplication distributes over addition,

$$C = (u_1 + u_2 d_A) \times G$$

Expanding the definition of u_1 and u_2 from verification step 5,

$$C = (zs^{-1} + rd_A s^{-1}) \times G$$

Collecting the common term s^{-1},

$$C = (z + rd_A)s^{-1} \times G$$

Expanding the definition of s from signature step 6,

$$C = (z + rd_A)(z + rd_A)^{-1}(k^{-1})^{-1} \times G$$

Since the inverse of an inverse is the original element, and the product of an element's inverse and the element is the identity, we are left with

$$C = k \times G$$

From the definition of r, this is verification step 6.

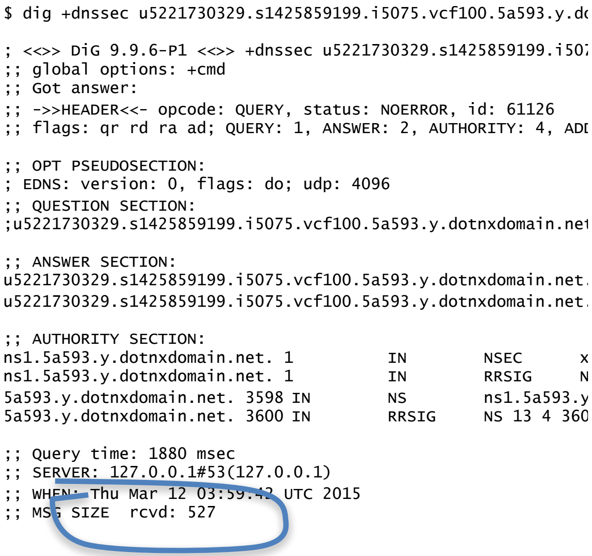
This shows only that a correctly signed message will verify correctly; many other properties are required for a secure signature algorithm.

ECDSA P-256

Elliptic Curve Cryptography allows for the construction of “strong” public/private key pairs with key lengths that are far shorter than equivalent strength keys using RSA

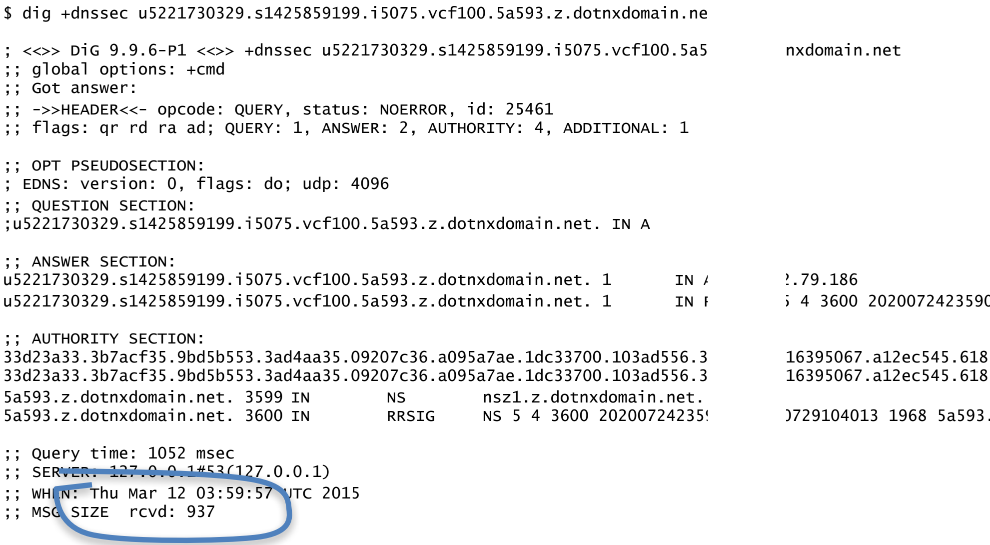
256-bit ECC public key should provide comparable security to a 3072-bit RSA public key

ECDSA vs RSS



```
$ dig +dnssec u5221730329.s1425859199.i5075.vcf100.5a593.y.d

; <<>> DiG 9.9.6-P1 <<>> +dnssec u5221730329.s1425859199.i50
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61126
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 4, AD

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;u5221730329.s1425859199.i5075.vcf100.5a593.y.dotnxdomain.ne

;; ANSWER SECTION:
u5221730329.s1425859199.i5075.vcf100.5a593.y.dotnxdomain.net
u5221730329.s1425859199.i5075.vcf100.5a593.y.dotnxdomain.net

;; AUTHORITY SECTION:
ns1.5a593.y.dotnxdomain.net. 1          IN        NSEC      x
ns1.5a593.y.dotnxdomain.net. 1          IN        RRSIG     N
5a593.y.dotnxdomain.net. 3598 IN        NS        ns1.5a593.y
5a593.y.dotnxdomain.net. 3600 IN        RRSIG     NS 13 4 360

;; Query time: 1880 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Mar 12 03:59:42 UTC 2015
;; MSG SIZE  rcvd: 527
```

ECDSA signed response – 527 octets



```
$ dig +dnssec u5221730329.s1425859199.i5075.vcf100.5a593.z.dotnxdomain.ne

; <<>> DiG 9.9.6-P1 <<>> +dnssec u5221730329.s1425859199.i5075.vcf100.5a5        nxdomain.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 25461
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 4, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;u5221730329.s1425859199.i5075.vcf100.5a593.z.dotnxdomain.net. IN A

;; ANSWER SECTION:
u5221730329.s1425859199.i5075.vcf100.5a593.z.dotnxdomain.net. 1      IN         .79.186
u5221730329.s1425859199.i5075.vcf100.5a593.z.dotnxdomain.net. 1      IN         4 3600 2020072423590

;; AUTHORITY SECTION:
33d23a33.3b7acf35.9bd5b553.3ad4aa35.09207c36.a095a7ae.1dc33700.103ad556.3        16395067.a12ec545.618
33d23a33.3b7acf35.9bd5b553.3ad4aa35.09207c36.a095a7ae.1dc33700.103ad556.3        16395067.a12ec545.618
5a593.z.dotnxdomain.net. 3599 IN        NS        nsz1.z.dotnxdomain.net.
5a593.z.dotnxdomain.net. 3600 IN        RRSIG     NS 5 4 3600 2020072423        729104013 1968 5a593.

;; Query time: 1052 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Mar 12 03:59:57 UTC 2015
;; MSG SIZE  rcvd: 937
```

RSA signed response – 937 octets

So let's use ECDSA for DNSSEC

Yes!

So let's use ECDSA for DNSSEC

Yes!

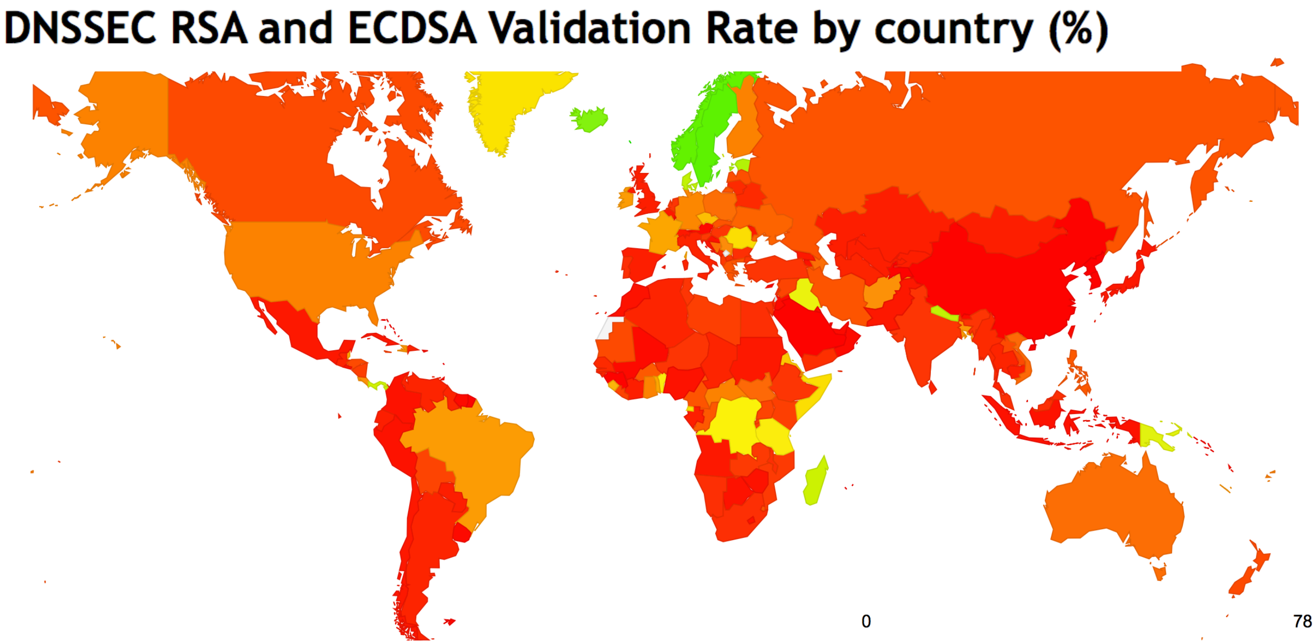
Let’s do that right now!

So lets use ECDSA for DNSSEC

Or maybe we should look before we leap...

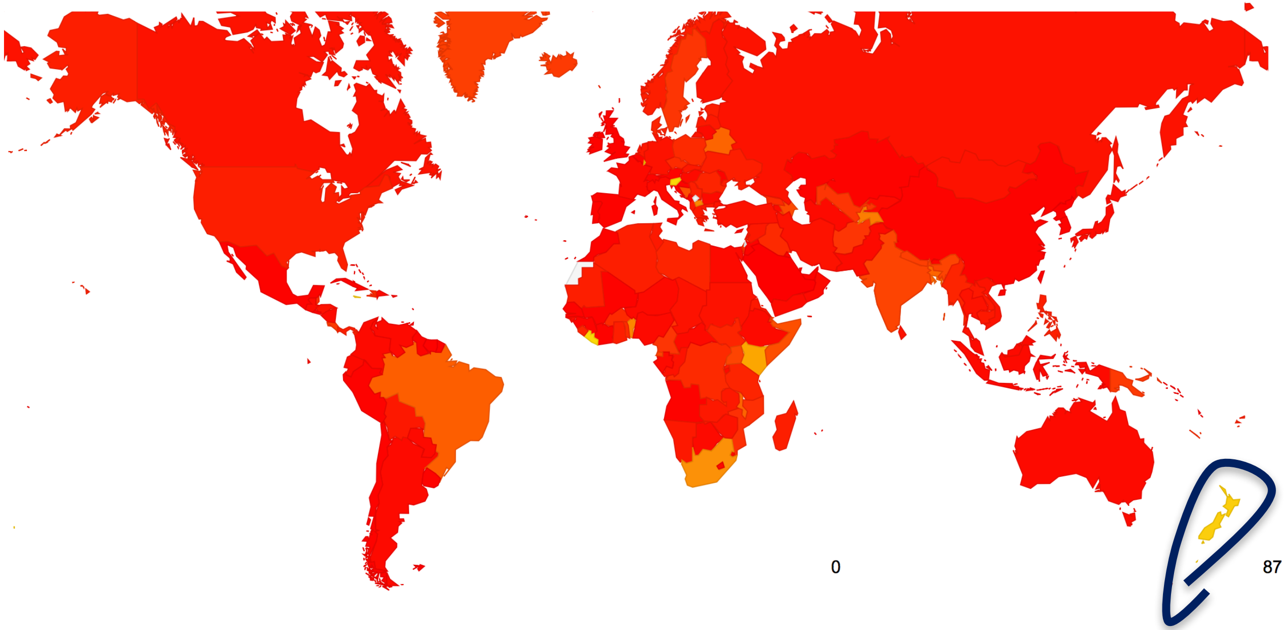
- Is ECDSA a “well supported” crypto protocol?
- If you signed using ECDSA would resolvers validate the signature?

We are now testing for where we see ECDSA Support



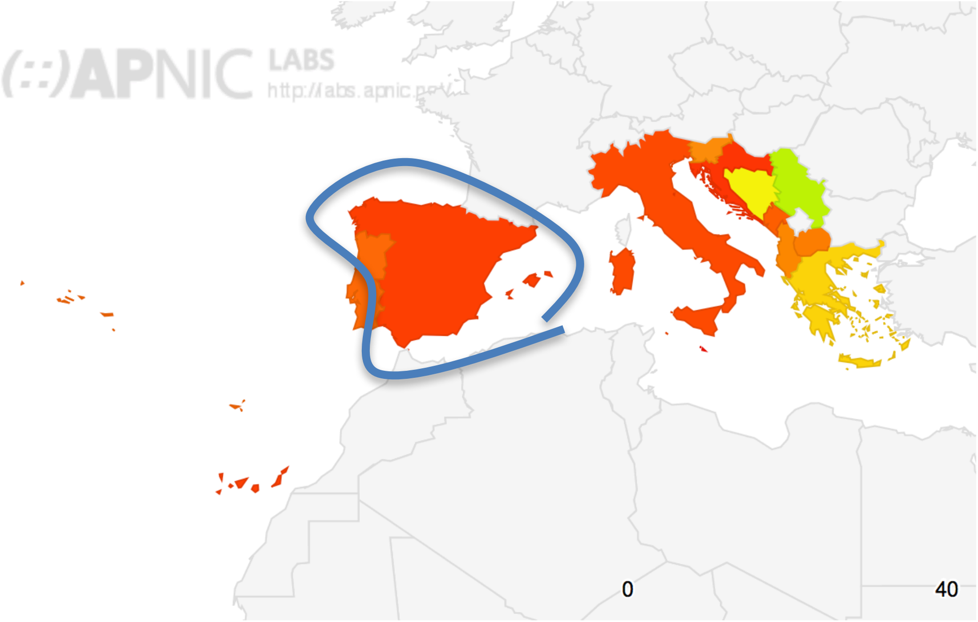
And where we don't

DNSSEC RSA and NOT ECDSA Validation Rate by country (%)



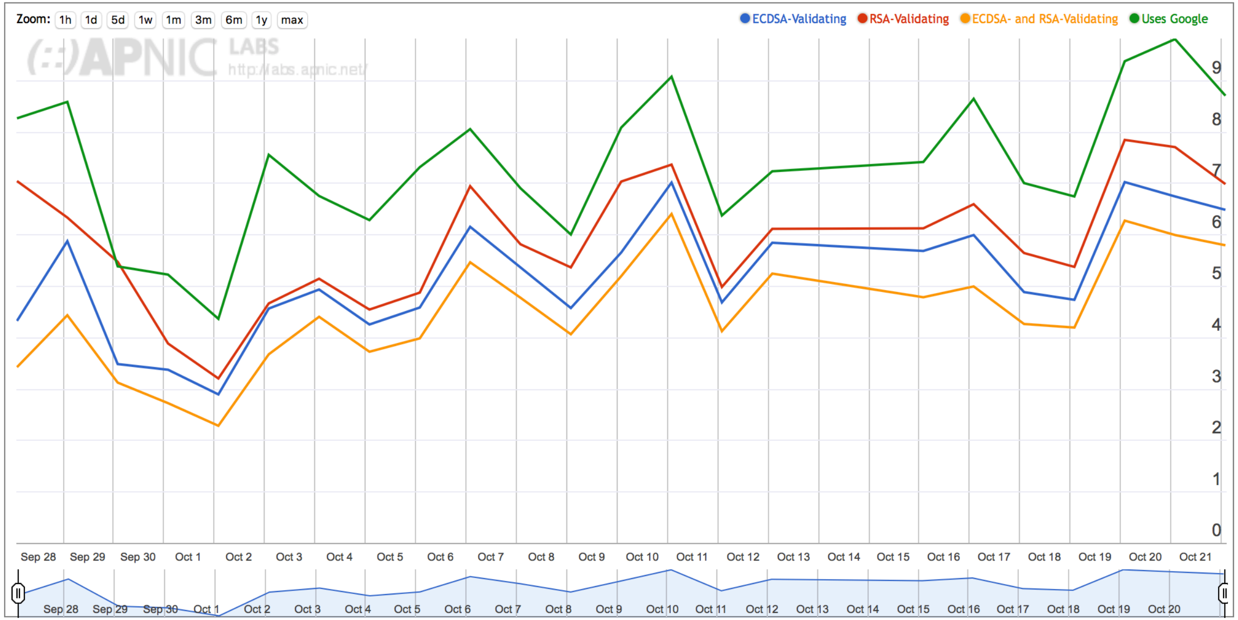
Today we're in Spain…

Region Map for Southern Europe (039)

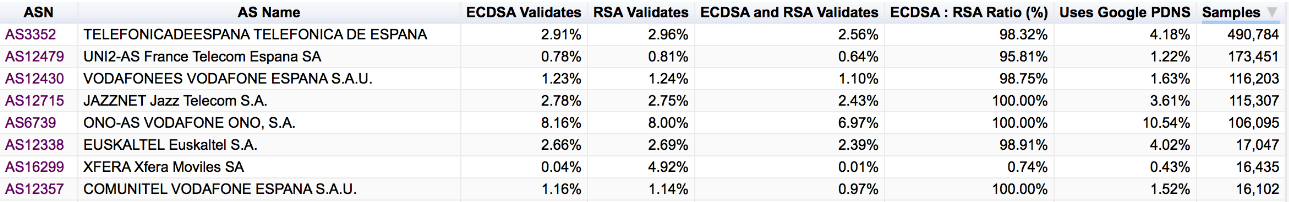


Today we're in Spain…

Use of DNSSEC-ECDSA Validation for Spain (ES)



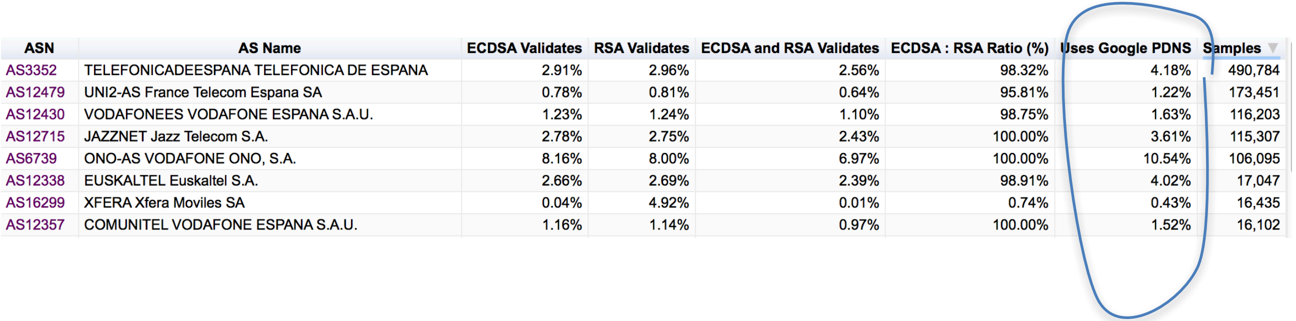
The Top 8 Spanish ISPs



ASN	AS Name	ECDSA Validates	RSA Validates	ECDSA and RSA Validates	ECDSA : RSA Ratio (%)	Uses Google PDNS	Samples ▼
AS3352	TELEFONICADEESPANA TELEFONICA DE ESPANA	2.91%	2.96%	2.56%	98.32%	4.18%	490,784
AS12479	UNI2-AS France Telecom Espana SA	0.78%	0.81%	0.64%	95.81%	1.22%	173,451
AS12430	VODAFONEES VODAFONE ESPANA S.A.U.	1.23%	1.24%	1.10%	98.75%	1.63%	116,203
AS12715	JAZZNET Jazz Telecom S.A.	2.78%	2.75%	2.43%	100.00%	3.61%	115,307
AS6739	ONO-AS VODAFONE ONO, S.A.	8.16%	8.00%	6.97%	100.00%	10.54%	106,095
AS12338	EUSKALTEL Euskaltel S.A.	2.66%	2.69%	2.39%	98.91%	4.02%	17,047
AS16299	XFERA Xfera Moviles SA	0.04%	4.92%	0.01%	0.74%	0.43%	16,435
AS12357	COMUNITEL VODAFONE ESPANA S.A.U.	1.16%	1.14%	0.97%	100.00%	1.52%	16,102

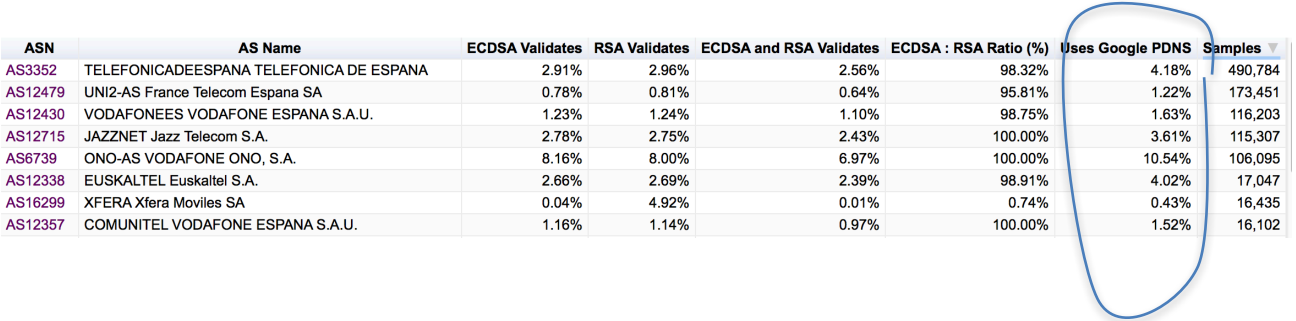
And the extent to which their uses perform DNSSEC validation with ECDSA and RSA

And it if wasn't for Google…



ASN	AS Name	ECDSA Validates	RSA Validates	ECDSA and RSA Validates	ECDSA : RSA Ratio (%)	Uses Google PDNS	Samples
AS3352	TELEFONICADEESPANA TELEFONICA DE ESPANA	2.91%	2.96%	2.56%	98.32%	4.18%	490,784
AS12479	UNI2-AS France Telecom Espana SA	0.78%	0.81%	0.64%	95.81%	1.22%	173,451
AS12430	VODAFONEES VODAFONE ESPANA S.A.U.	1.23%	1.24%	1.10%	98.75%	1.63%	116,203
AS12715	JAZZNET Jazz Telecom S.A.	2.78%	2.75%	2.43%	100.00%	3.61%	115,307
AS6739	ONO-AS VODAFONE ONO, S.A.	8.16%	8.00%	6.97%	100.00%	10.54%	106,095
AS12338	EUSKALTEL Euskaltel S.A.	2.66%	2.69%	2.39%	98.91%	4.02%	17,047
AS16299	XFERA Xfera Moviles SA	0.04%	4.92%	0.01%	0.74%	0.43%	16,435
AS12357	COMUNITEL VODAFONE ESPANA S.A.U.	1.16%	1.14%	0.97%	100.00%	1.52%	16,102

And it if wasn't for Google…

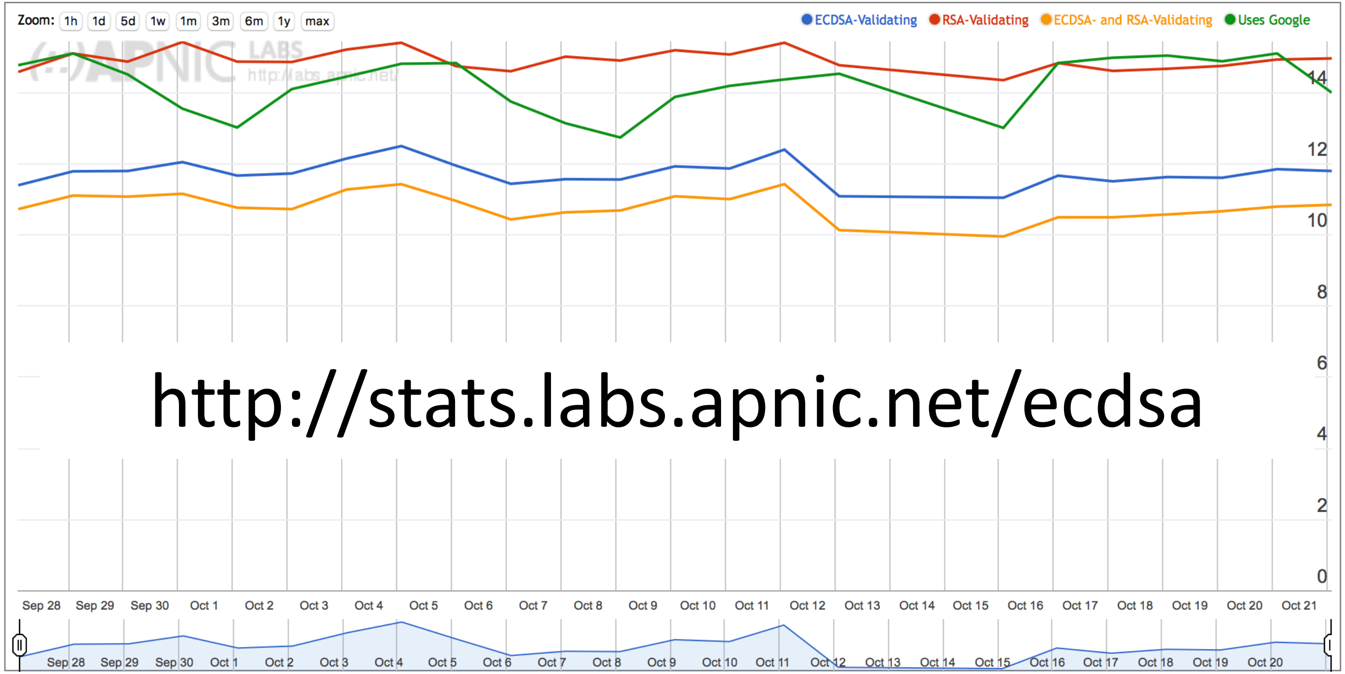


ASN	AS Name	ECDSA Validates	RSA Validates	ECDSA and RSA Validates	ECDSA : RSA Ratio (%)	Uses Google PDNS	Samples
AS3352	TELEFONICADEESPANA TELEFONICA DE ESPANA	2.91%	2.96%	2.56%	98.32%	4.18%	490,784
AS12479	UNI2-AS France Telecom Espana SA	0.78%	0.81%	0.64%	95.81%	1.22%	173,451
AS12430	VODAFONEES VODAFONE ESPANA S.A.U.	1.23%	1.24%	1.10%	98.75%	1.63%	116,203
AS12715	JAZZNET Jazz Telecom S.A.	2.78%	2.75%	2.43%	100.00%	3.61%	115,307
AS6739	ONO-AS VODAFONE ONO, S.A.	8.16%	8.00%	6.97%	100.00%	10.54%	106,095
AS12338	EUSKALTEL Euskaltel S.A.	2.66%	2.69%	2.39%	98.91%	4.02%	17,047
AS16299	XFERA Xfera Moviles SA	0.04%	4.92%	0.01%	0.74%	0.43%	16,435
AS12357	COMUNITEL VODAFONE ESPANA S.A.U.	1.16%	1.14%	0.97%	100.00%	1.52%	16,102

There would be no DNSSEC at all!

And no ECDSA!

The full daily report



http://stats.labs.apnic.net/ecdsa