The Challenges of Web Security

James H. Davenport

University of Bath

12 November 2012

Agenda

1. How secure is the communication: can a third party eavesdrop on what is being shared?
2. Is the "end" really who my device thinks it is, or am I the victim of a "man-in-the-middle" attack?
3. Is the "end" my device is talking to the entity I intend my device to be talking to?

The first two are essentially technical problems, but the third is definitely socio-technical.

Internet and Security

We don't normally shout our PIN numbers out in crowded supermarkets, so why should we broadcast them on wireless networks?

It's not only James Bond who wants cryptography?

Numbers rather than Padlocks (I)

Idea due to Diffie & Hellman (1976) [3]
A wishes to send x to B.
A and B each think of a random number, say a and b.

A's action	Message	B's action
multiply x by a		
	xa $\searrow$	
		multiply message by b
	$xba = xab$ $\swarrow$	
divide message by a		
	xb $\searrow$	
		divide message by b

In practice, to avoid guessing, and numerical errors, x, a and b are whole numbers modulo some *large* prime p.

- A single processor can perform a few thousand million operations per second
- So maybe 'large' should mean more than that
- We could image the 'bad guys' having a thousand processors
- and maybe waiting weeks
- So maybe 'large' should mean more than that
- In fact 'large' is recommended to be $> 10^{150}$, e.g.

100000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000067

Numbers rather than Padlocks (I) — snag

A's action	Message	B's action
multiply x by a		
	xa $\searrow$	
		multiply message by b
	$xba = xab$ $\swarrow$	
divide message by a		
	xb $\searrow$	
		divide message by b

Eavesdropper computes $\frac{xa \cdot xb}{xab} = x$.
So replacing the padlocks by numbers has given the eavesdropper the chance of doing arithmetic.

Numbers rather than Padlocks (II)

Let's be more subtle : upgrade from multiplication to powers.

A's action	Message	B's action
raise x to power a		
	x^a $\searrow$	
		raise message to power b
	$(x^b)^a = (x^a)^b$ $\swarrow$	
take ath root of message		
	x^b $\searrow$	
		take bth root of message

Surely this frustrates the eavesdropper?

But what about logarithms?

A's action	Message	B's action
raise x to power a		
	x^a $\searrow$	
		raise message to power b
	$(x^b)^a = (x^a)^b$ $\swarrow$	
take ath root of message		
	x^b $\searrow$	
		take bth root of message

Eavesdropper computes

$$\frac{\log(x^a)\cdot\log(x^b)}{\log(x^{ab})} = \frac{a\log(x)\cdot b\log(x)}{ab\log(x)} = \log(x).$$

Essentially the same trick as before, but with logarithms!

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 7, so that $\log(7) = 1$.

1	2	3	4	5	6	7	8	9	10
0						1			
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40

So $\log(49) = 2$, but $49 = 1 \cdot 41 + 8 \equiv 8$ since we are working modulo 41, and $\log(7 \cdot 8) = 3$, but $7 \cdot 8 = 56 \equiv 15$, so $\log(15) = 3$.

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 7, so that $\log(7) = 1$.

1	2	3	4	5	6	7	8	9	10
0						1	2		
11	12	13	14	15	16	17	18	19	20
				3					
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40

And we can fill in: $8 \cdot 8 = 64 \equiv 23$, so $\log(23) = 4$. Also $8 \cdot 15 = 120 \equiv -3 = 38$ so $\log(38) = 2 + 3 = 5$ and $\log(9) = 10$.

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 7, so that $\log(7) = 1$.

1	2	3	4	5	6	7	8	9	10
0						1	2	10	
11	12	13	14	15	16	17	18	19	20
				3					
21	22	23	24	25	26	27	28	29	30
		4							
31	32	33	34	35	36	37	38	39	40
							5		

$15^2 \equiv 20$, so $\log(20) = 6$. $20^2 = 400 \equiv 31$, so $\log(31) = 12$.

Do logarithms exist?

Remember that we are working modulo a *large* prime p. For simplicity, I will take $p = 41$, since it's small enough, and logs base 7, so that $\log(7) = 1$.

1	2	3	4	5	6	7	8	9	10
0						1	2	10	
11	12	13	14	15	16	17	18	19	20
				3					6
21	22	23	24	25	26	27	28	29	30
		4							
31	32	33	34	35	36	37	38	39	40
12							5		

and we can keep going, but it's a tedious process:
p operations for a table
methods taking roughly $\sqrt{p}$ operations are known, and faster methods taking roughly $e^{c\sqrt{\log p \log\log p}}$ operations, or even $e^{c'\sqrt[3]{\log p \log^2 \log p}}$ operations, but it's still tedious!

Simplicity can be dangerous

- Not all p are equally difficult!
- In particular, we would like p to be such that $q = \frac{p-1}{2}$ is also prime, so that q is a *Sophie Germain* prime
- **Conjecturally**, there are infinitely many of these

Also,beware of shortcuts! In the 1980s, the Federal Reserve Bank needed such a system, and used $GF(2^{127})$ rather than a prime near that.

- Coppersmith [1] broke this with a $e^{1.35\sqrt[3]{\log p \log^2 \log p}}$ attack, pragmatically 7 hours CPU on a 38.5MHz machine (one of the fastest in the world in 1982!).

But it takes three messages

sequentially. Can we do better? Let x be a **public** number. Again, A and B choose random numbers a and b.

A's action	Message	B's action
raise x to power a		raise x to power b
	x^a ↘ ↙ x^b	
	↙ ↘	
raise message to power a		raise message to power b
$(x^b)^a$		$(x^a)^b$

Now they are *both* in possession of $(x^a)^b = (x^b)^a$, which can be used as the key for any standard cipher.

Two messages, and in parallel!

This is *one* reason why secure websites display a padlock: to assure you that they have gone through this process between *your* browser and the web site: so the *communication* is secure.

The man in the middle (also known as Janus)

Again, A and B choose random numbers a and b.
But J chooses a' and b'.

Diffie–Hellman 1			Diffie–Hellman 2		
A does	Message	J does	J does	Message	B does
make x^a		make $x^{b'}$	make $x^{a'}$		make x^b
	x^a $\searrow\swarrow$ $x^{b'}$			$x^{a'}$ $\searrow\swarrow$ x^b	
	$\swarrow\searrow$			$\swarrow\searrow$	
$(x^{b'})^a$		$(x^a)^{b'}$	$(x^b)^{a'}$		$(x^{a'})^b$
Secure communication A $\leftrightarrow$ J			Secure communication J $\leftrightarrow$ B		

The man in the middle (also known as Janus)

Again, A and B choose random numbers a and b.
But J chooses a' and b'.

Diffie–Hellman 1			Diffie–Hellman 2		
A does make x^a	Message	J does make $x^{b'}$	J does make $x^{a'}$	Message	B does make x^b
	x^a ↘↙ $x^{b'}$ ↙↘			$x^{a'}$ ↘↙ x^b ↙↘	
$(x^{b'})^a$		$(x^a)^{b'}$	$(x^b)^{a'}$		$(x^{a'})^b$
Secure communication A $\leftrightarrow$ J			Secure communication J $\leftrightarrow$ B		

Insecure communication between A and B
J can read everything

Public Secrets! (I)

Original idea due to Rivest, Shamir & Adleman (1977) [6]
The analogy is with a signature: anyone who knows my signature can check that it's mine, but in principle only I can produce it

Theorem (Fermat's Little Theorem (special case))

Let $N = pq$ where p, q are different primes, then

$$m^{N-p-q+1} \equiv 1 \pmod{N}$$

(provided m is not divisible by p or q)

Corollary (RSA)

If $de \equiv 1 \pmod{N - p - q + 1}$, $(m^e)^d \equiv m \pmod{N}$

We think of e as the **encryption** exponent, and d as the **decryption** exponent

Public Secrets! (II)

Therefore, if I

- publish (my) N and d, but keep e (and p, q) secret
- Send you $c := m^e \pmod{N}$.
- You can compute $c^d = (m^e)^d \equiv m \pmod{N}$
- and be sure that only I could have constructed c

Of course m must be self-identifying

Breaking Public Secrets!

- Clearly *if* I can factor N, compute p and q, then I can compute e
- Factoring is hard! Best known algorithms again take $e^{c\sqrt[3]{\log N \log^2 \log N}}$, with $c \approx 7.1$.
- The current world record is a 768-bit number [4], using 2000 CPU-years (and 2 elapsed years)
- A 1024-bit number would be 10^{11} times as difficult
- *If* I know d, e, then I can factor N [2]
- Nothing precludes there being a way of computing c some other way

The Real Problem is Publishing

Original idea was literally that: XX Bank would publish the number in the paper

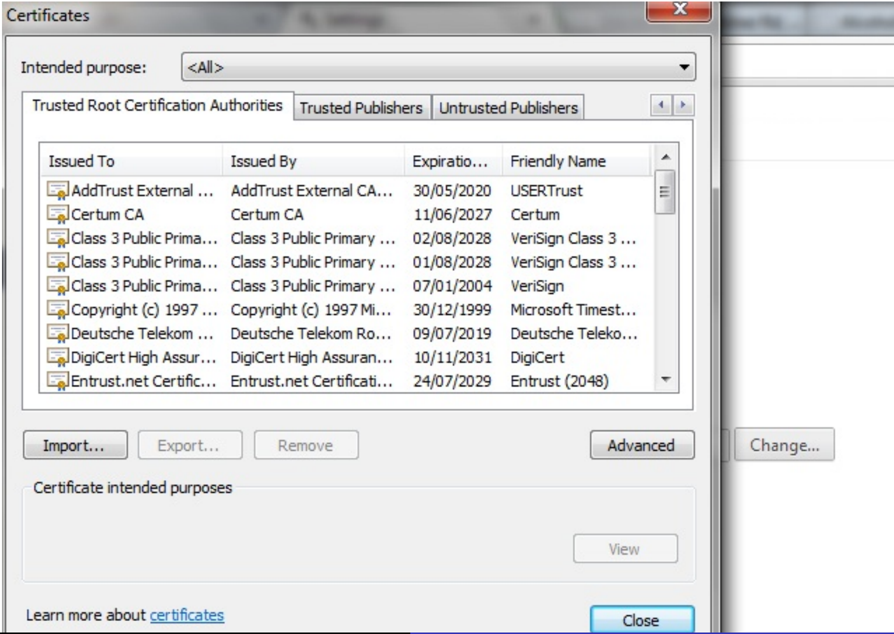
- Certificates are quite long 2×1024 bits $= 512$ hexadecimal digits (compared with 8 or 16 for a wireless key)
- My bank, and my supermarket, and my railway company, and Amazon, and ...
- How do I guarantee genuineness?

Hence what we need is a **Public Key Infrastructure** (PKI)

today's Web Public Key Infrastructure

- Your browser has certain "Certificate Authorities" built into it
-
-
-

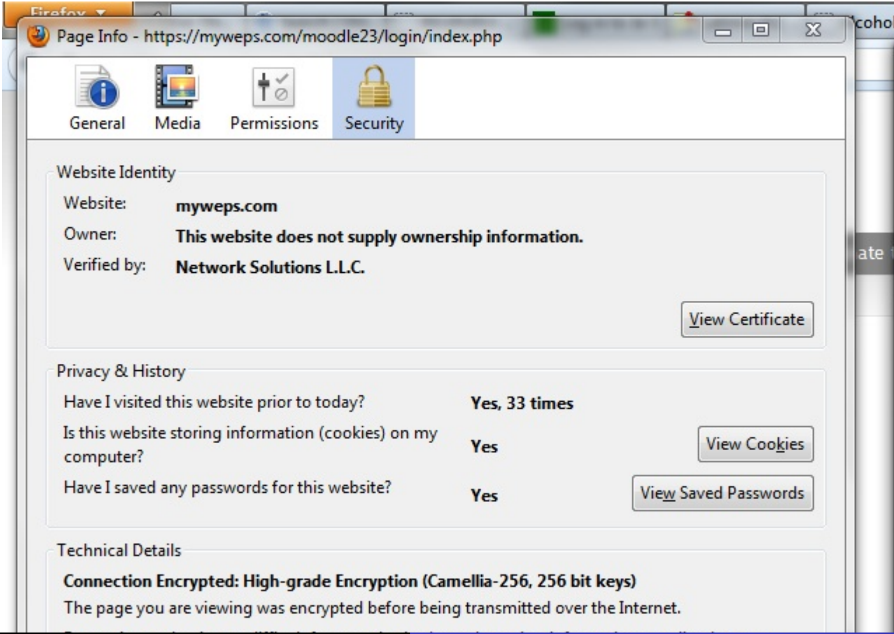
today's Web Public Key Infrastructure (Chrome)



today's Web Public Key Infrastructure

- Your browser has certain "Certificate Authorities" built into it
- And these are used to sign the certificates of sites
-
-

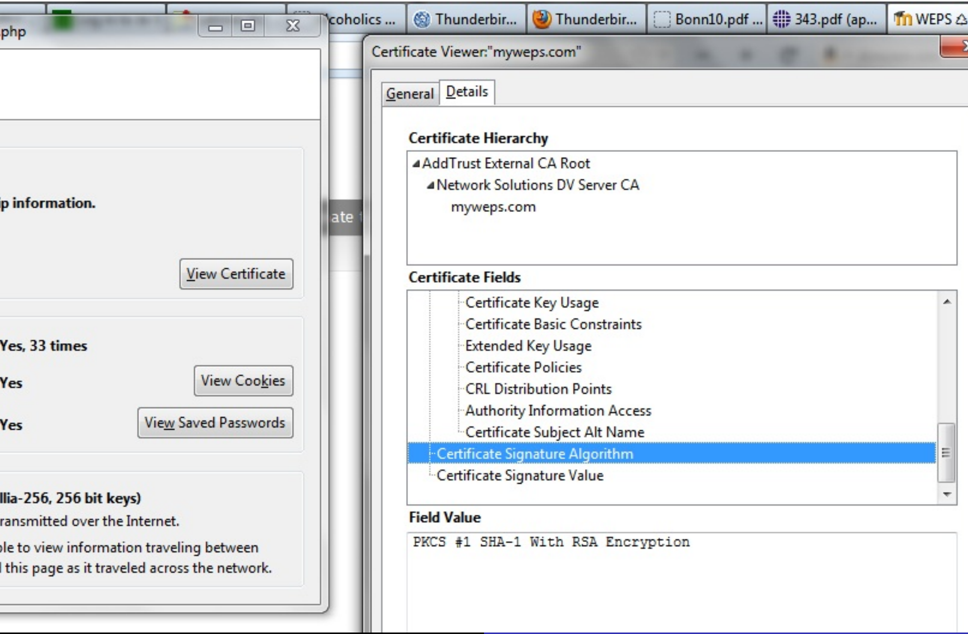
today's Web Public Key Infrastructure (Firefox)



today's Web Public Key Infrastructure

- Your browser has certain "Certificate Authorities" built into it
- And these are used to sign the certificates of sites
- Quite possibly through several layers
-

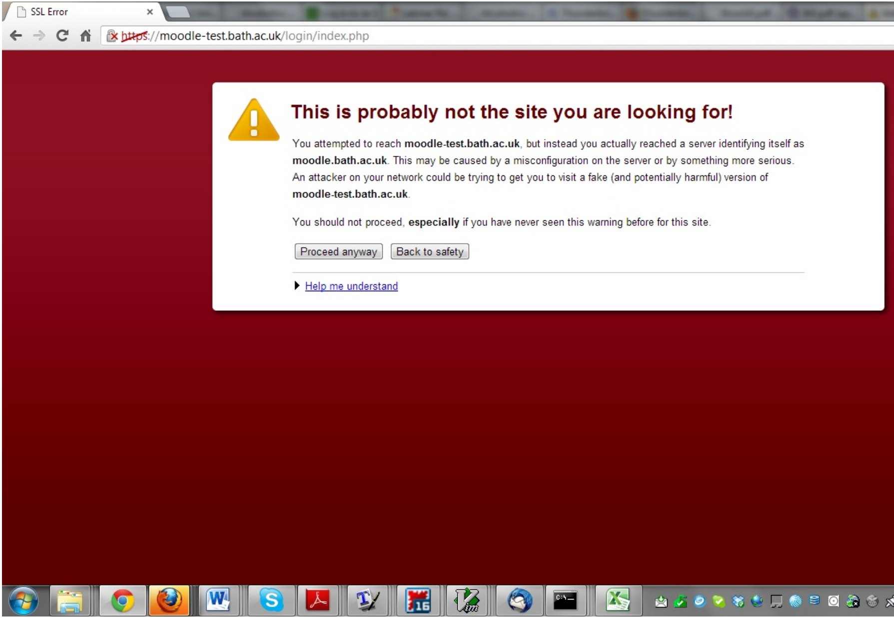
today's Web Public Key Infrastructure (Firefox)



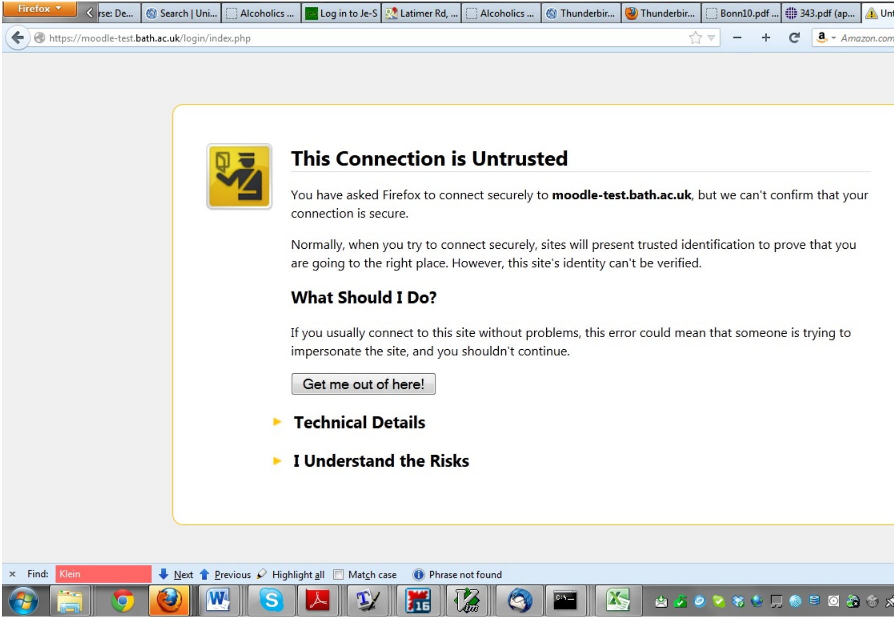
today's Web Public Key Infrastructure

- Your browser has certain “Certificate Authorities” built into it
- And these are used to sign the certificates of sites
- Quite possibly through several layers
- If this doesn't check out, you get a warning

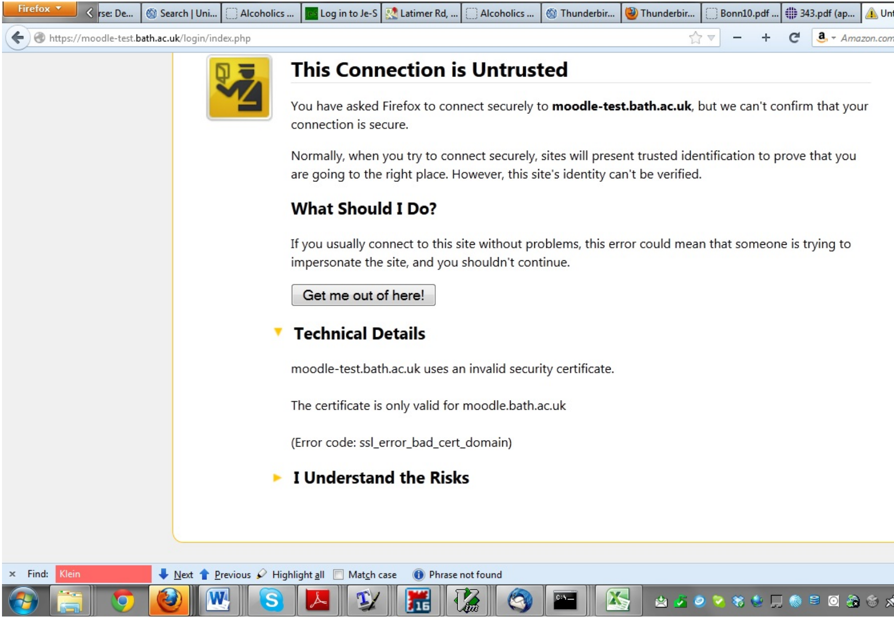
today's Web Public Key Infrastructure (Chrome)



today's Web Public Key Infrastructure (Firefox 1)



today's Web Public Key Infrastructure (Firefox 2)



This System has its Flaws

- There's no mechanism for revoking a certificate:
- we just have to wait for it to expire
- (and certainly not of a root certificate, where the expiry periods are **long**)
- My biggest worry is about some of the intermediate authorities
- There are various technical chinks in the armour
- Especially for cloud-based servers [5]

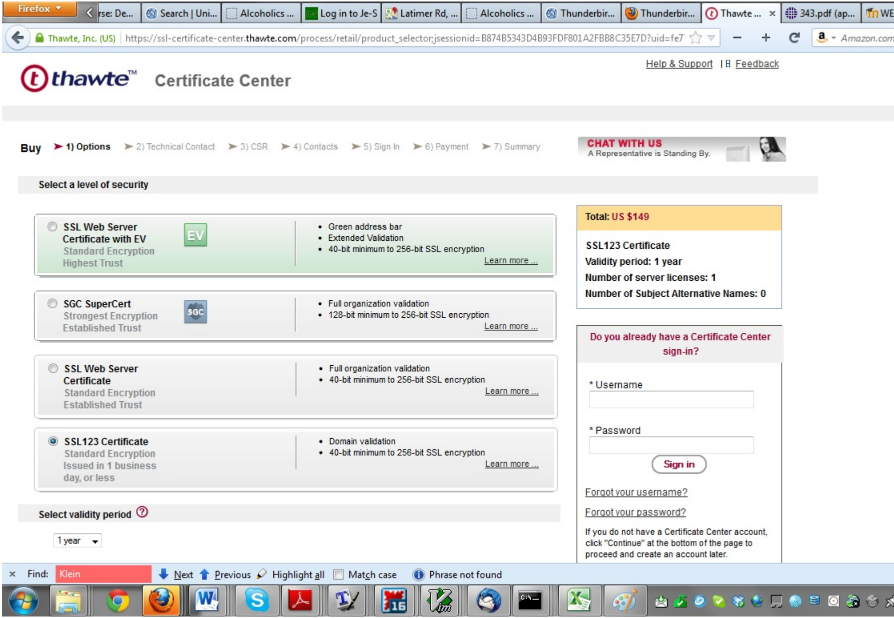
Nevertheless it seems to work pretty well from a technical point of view

- Getting a certificate is pretty easy
-
-
-
-
-

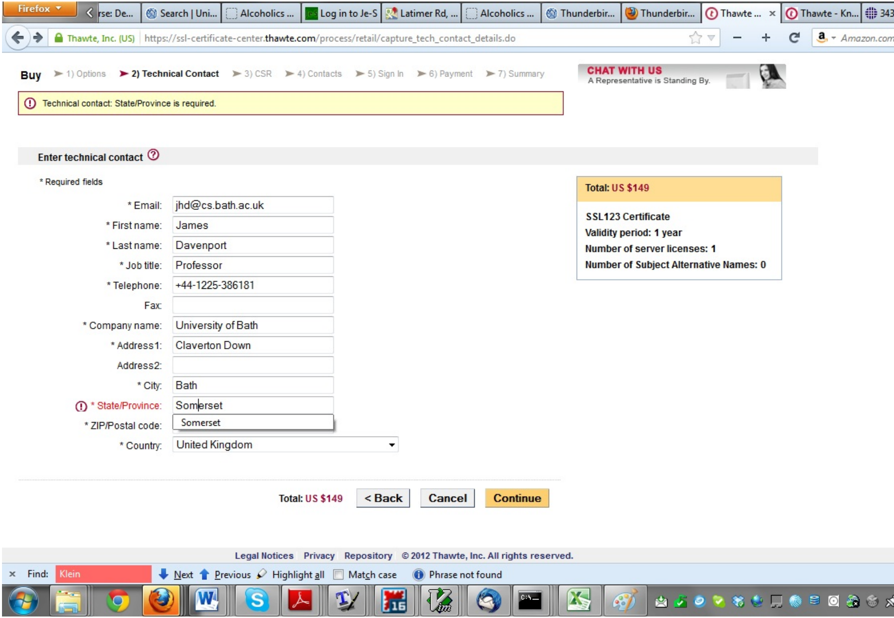
  1
-

[1] Pointed out by the audience

The Human Side of Security (Thawte)



The Human Side of Security (Thawte)



Human Beings have their Flaws

- Getting a certificate is pretty easy
- Basically, all you need is to be `postmaster@mydomain.co.uk` to get the certificate e-mailed to you
- And getting the domain is easy
- 5 minutes and £5.39 to get `JamesDavenport.me.uk`
- And probably `www.british-airway.co.uk`
- Or many other forms of "typo-squatting" (such as expiry-date squatting)[2]

[2] Pointed out by the audience

Solutions?

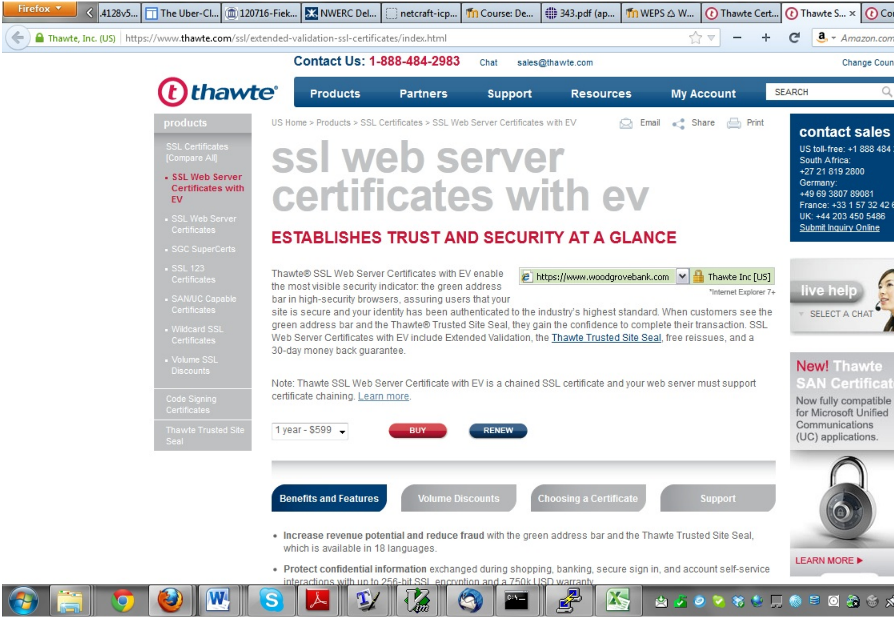
> *Whoever thinks his problem can be solved using cryptography, doesnt understand his problem and doesnt understand cryptography.*
> *Attributed by Roger Needham and Butler Lampson to each other*

Basically, two families of solutions

- Certificate-based solutions, such as “Extended Verification”

-

Extended validation (Thawte)



Solutions?

Whoever thinks his problem can be solved using cryptography, doesnt understand his problem and doesnt understand cryptography.

Attributed by Roger Needham and Butler Lampson to each other

Basically, two families of solutions

- Certificate-based solutions, such as "Extended Verification"
- How much "extended validation" can the CA purchase for the price difference: $450?
- Name-based solutions, such as Nominet's `.uk` proposal

Nominet's .uk proposal

http://www.nominet.org.uk/sites/default/files/Nominet_FINAL_electronic_form3_0.pdf

> *To further support the economic growth of the UK internet, we are holding a three month consultation about the potential introduction of a new service known as direct.uk, which would be specifically designed for businesses that are or want to get online, with a new shorter domain name of internet.uk rather than internet.co.uk.*
>
> *Proposed key features include; verification to check a registrant has a UK address, daily monitoring for malicious software and viruses, and a digital signature which minimises the risks of a domain name being hijacked. These measures would be supported by a trustmark to give consumers a clear sign that it was a verified domain name.*

References

D. Coppersmith.
Fast Evaluation of Logarithms in Fields of Characteristic Two.
IEEE Trans. Information Theory, IT-30:587–594, 1984.

J.-S. Coron and A. May.
Deterministic Polynomial-Time Equivalence of Computing the RSA Secret Key and Factoring.
J. Cryptology, 20:39–50, 2007.

W. Diffie and M.E. Hellman.
New Directions in Cryptography.
IEEE Trans. Inform. Theory, IT-22:644–654, 1976.

T. Kleinjung, K. Aoki, J. Franke, A.K. Lenstra, E Thomé, J.W. Bos, P. Gaudry, A. Kruppa, P.L. Montgomery, D.A. Osvik, H. te Riele, A. Timofeev, and P. Zimmermann.
Factorization of a 768-bit RSA modulus.
In T. Rabin, editor, *Proceedings CRYPTO 2010*, pages 333–350.

T. Ristenpart and S. Yilek.
When Good Randomness Goes Bad: Virtual Machine Reset Vulnerabilities and Hedging Deployed Cryptography.
In *Proceedings ISOC NDSS 2010*, 2010.

R.L. Rivest, A. Shamir, and L. Adleman.
U.S. Patent 4405829 — Cryptographic Communications System and Method.