RSAConference2015

San Francisco | April 20-24 | Moscone Center

CHANGE

Challenge today's security thinking

SESSION ID: DSP-R01

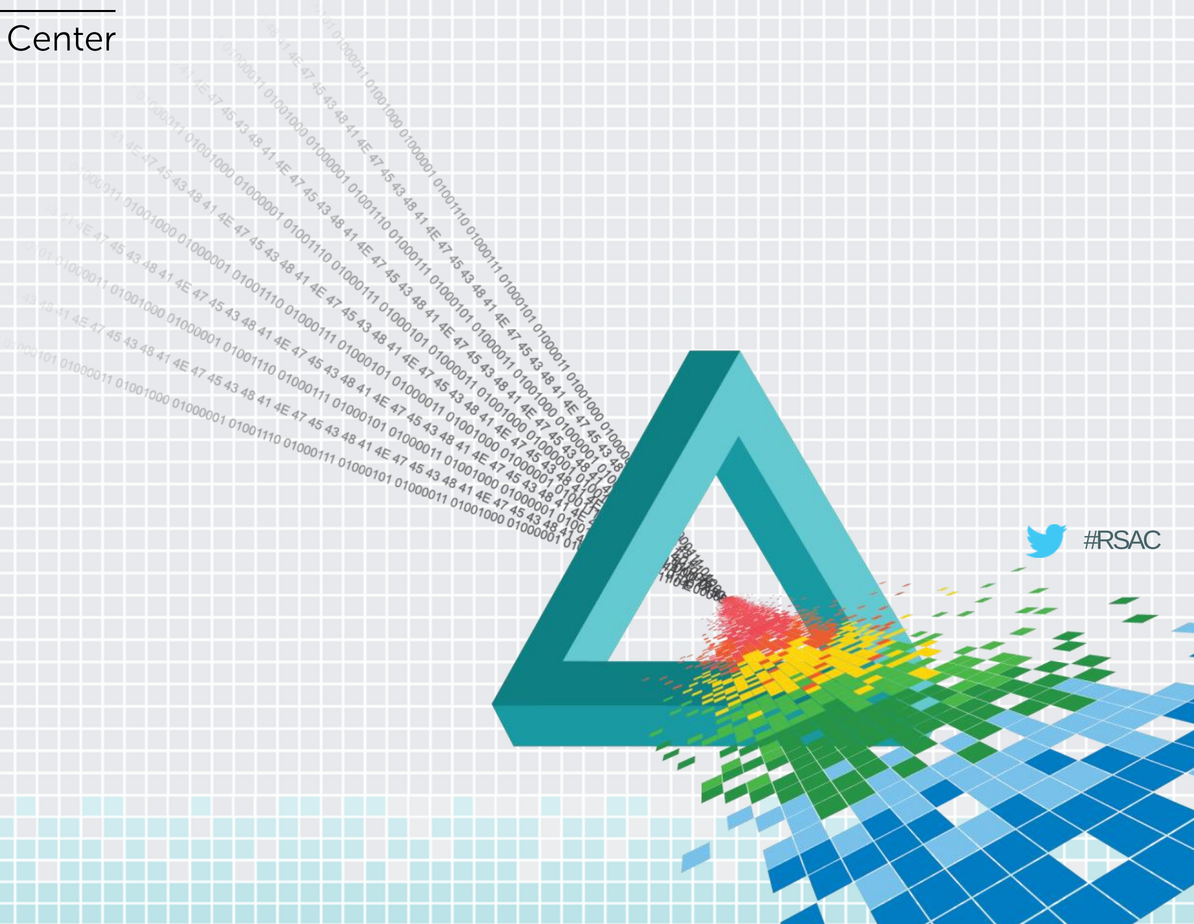
Seven Grades of Perfect Forward Secrecy

Oleg Gryb

Sr. Manager, Security Engineering
Samsung SSIC

#RSAC

PFS - Definitions



Classical PFS Definition

"Long-term secret keying material does not compromise the secrecy of the exchanged keys from earlier run"

W. Diffie, P. Oorchot, M.Wiener: Authentication and Authenticated Key Exchanges, 1992

http://people.scs.carleton.ca/~paulv/papers/sts-final.pdf

Session and Long-term Keys

Session Keys:

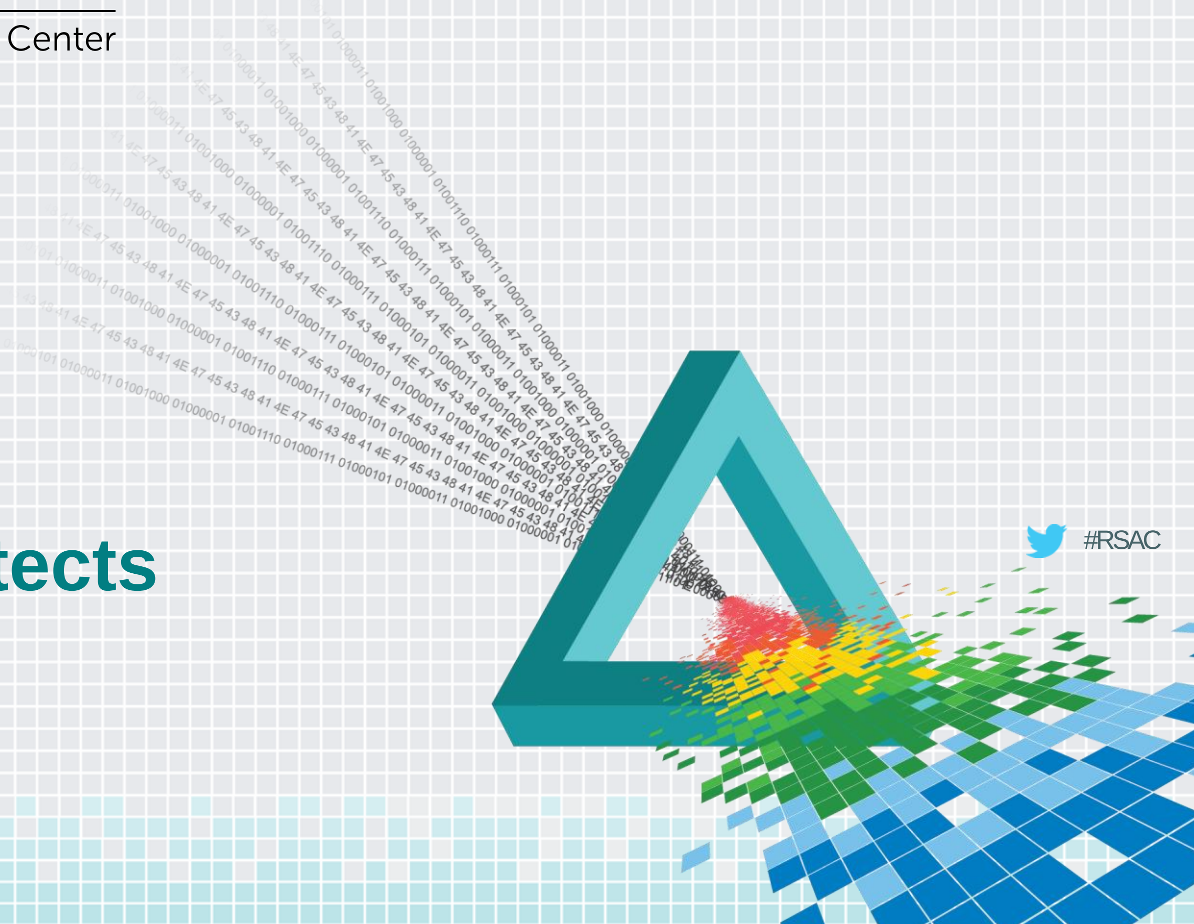
- One time symmetric key used to encrypt all messages in a session.
- Similar to a one time use password (OTP).

Long-term Keys:

- Live longer than a session. It can actually live years.
- Can be used to derive Session Key.
- Idealistically stored in an HSM appliance, but it varies.

RSAConference2015

San Francisco | April 20-24 | Moscone Center



#RSAC

PFS – What it protects against

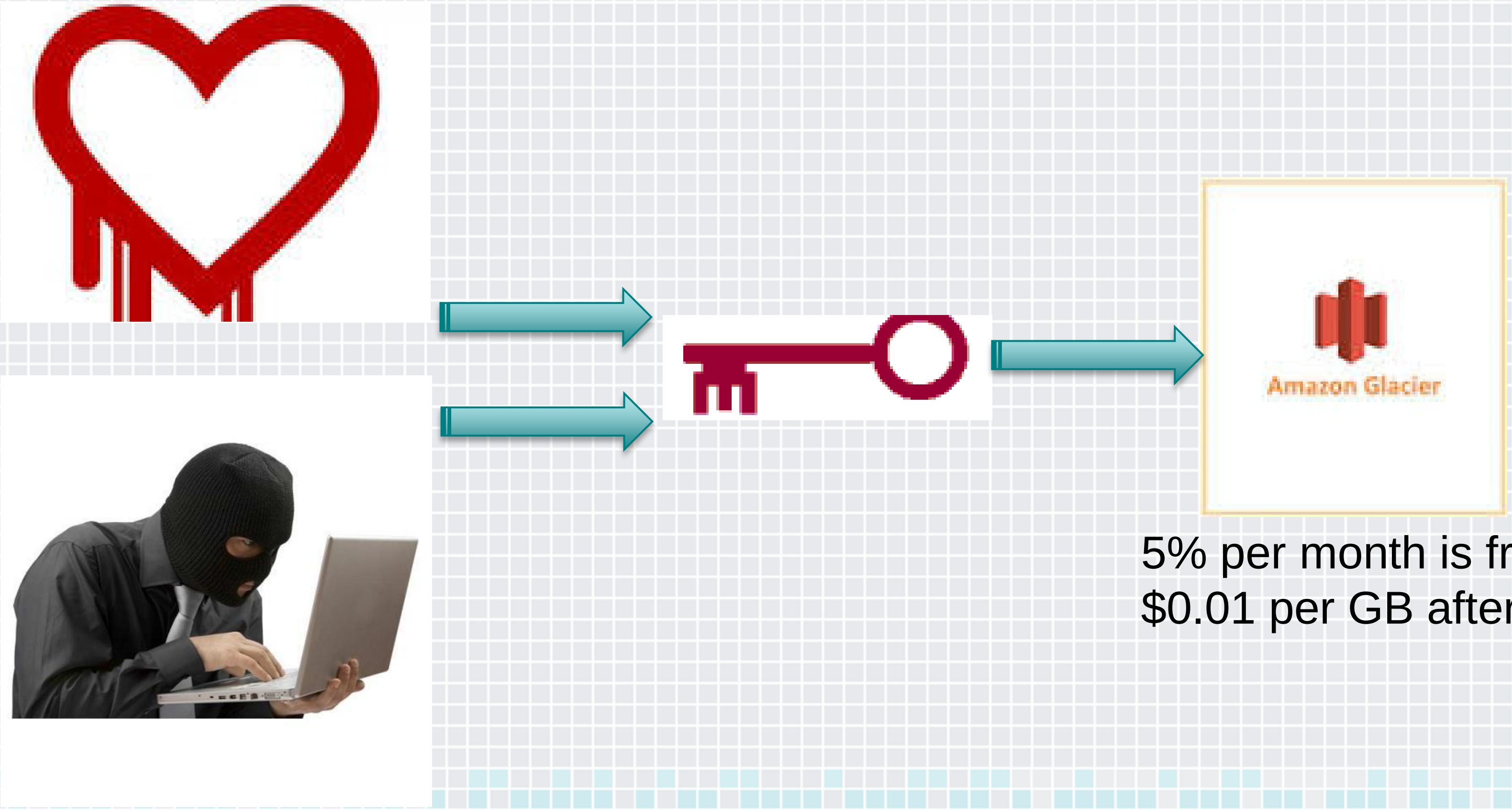
Why PFS is important

Installing TCPDUMP on DD-WRT is easy:

Emtunk's Blog

$0.01 per GB/month

Exploit, if PFS is not implemented

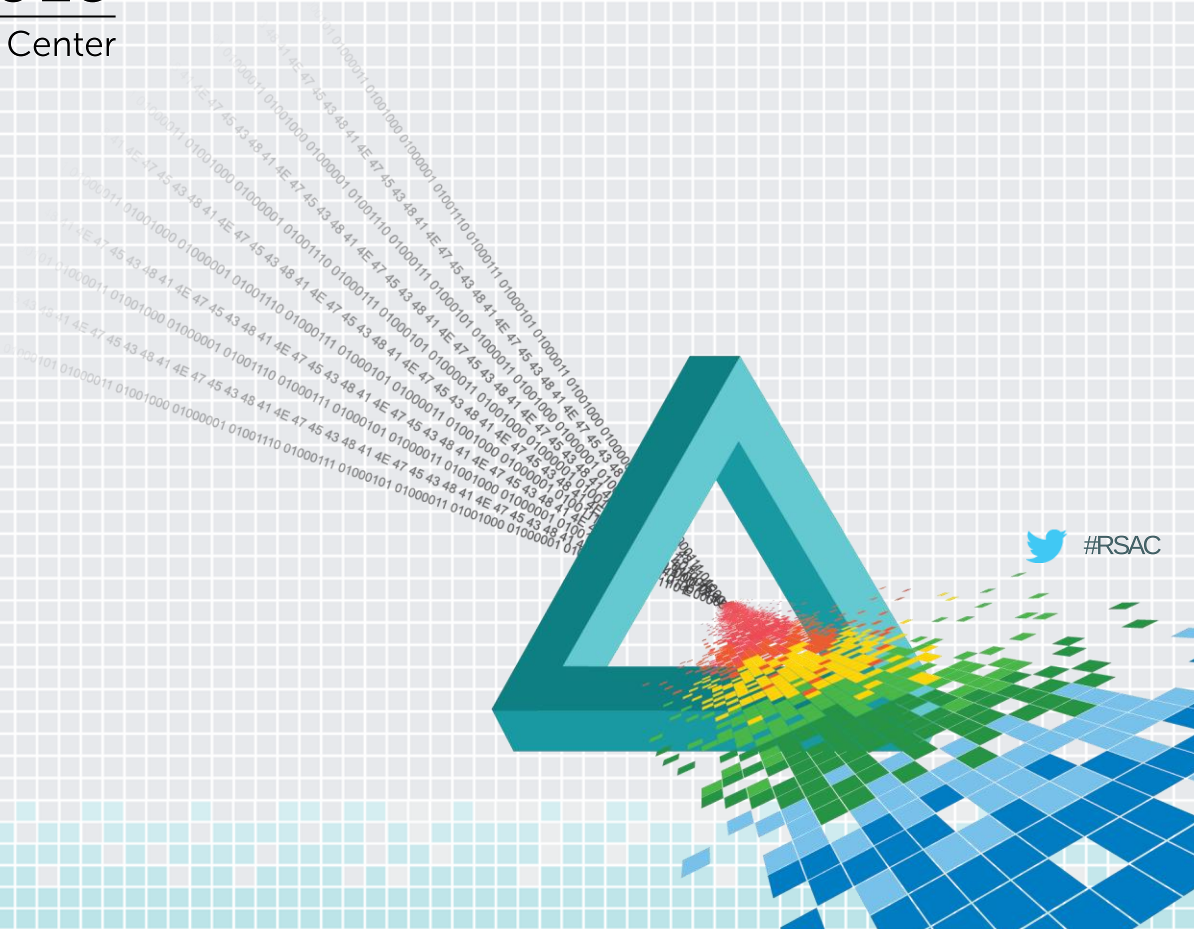


5% per month is free
$0.01 per GB after that

RSAConference2015

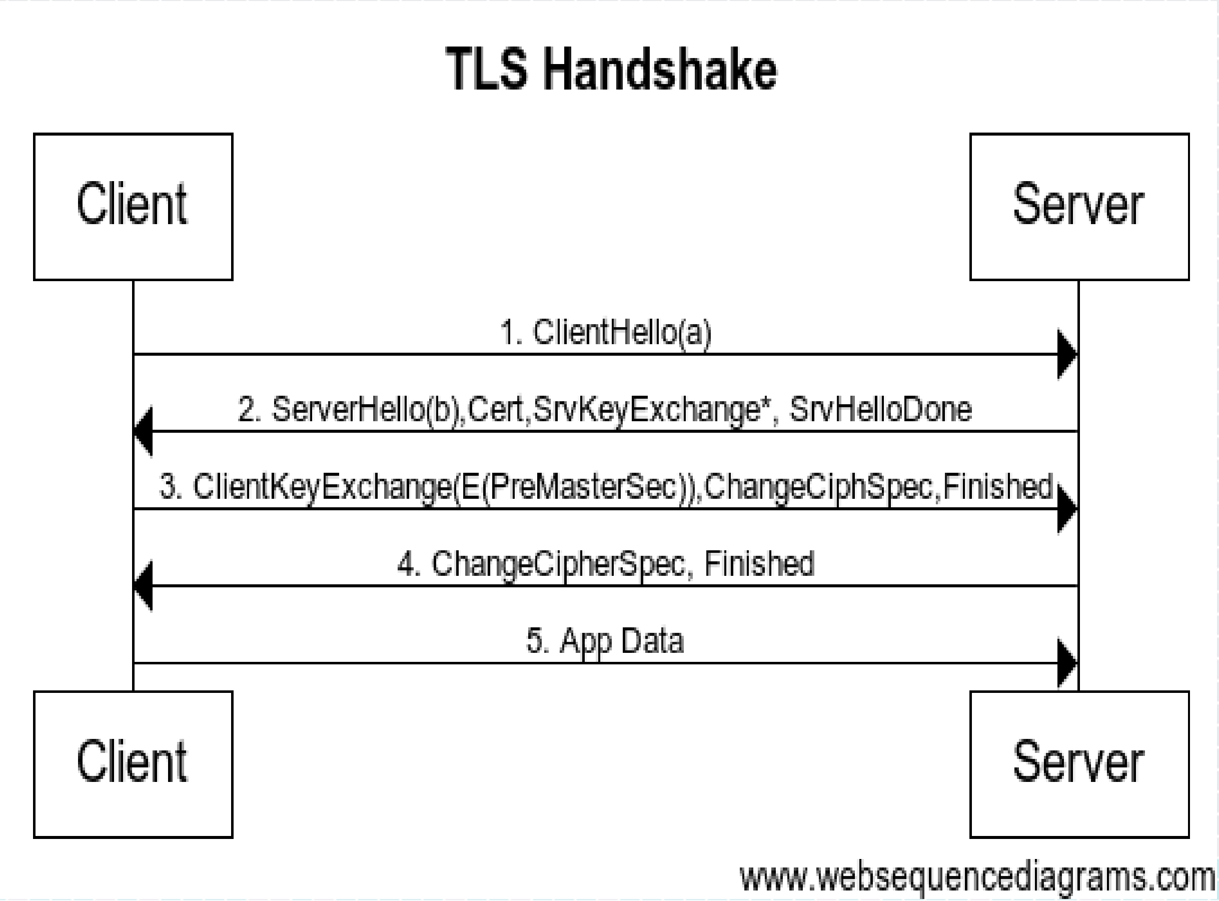
San Francisco | April 20-24 | Moscone Center

TLS Handshakes



#RSAC

SSL Handshake without PFS


TLS Handshake
Client
Server
1. ClientHello(a)
2. ServerHello(b),Cert,SrvKeyExchange*, SrvHelloDone
3. ClientKeyExchange(E(PreMasterSec)),ChangeCiphSpec,Finished
4. ChangeCipherSpec, Finished
5. App Data
Client
Server

www.websequencediagrams.com

Session key is generated from Premaster, random numbers 'a’ and 'b’.

Premaster is encrypted with long-term server's key

If long-term key is compromized,session key is compromized too.

PFS with traditional Diffie-Hellman

SrvKeyExchange will contain additional DHparams:

p – big prime

g – its primitive root: ∀a coprime p∃k : g^k≡a(mod p)

Ys=g^a mod p – this is server's public key

ClientKeyExchange will contain ClientDiffieHellmanPublic instead of RSA Premaster Secret:

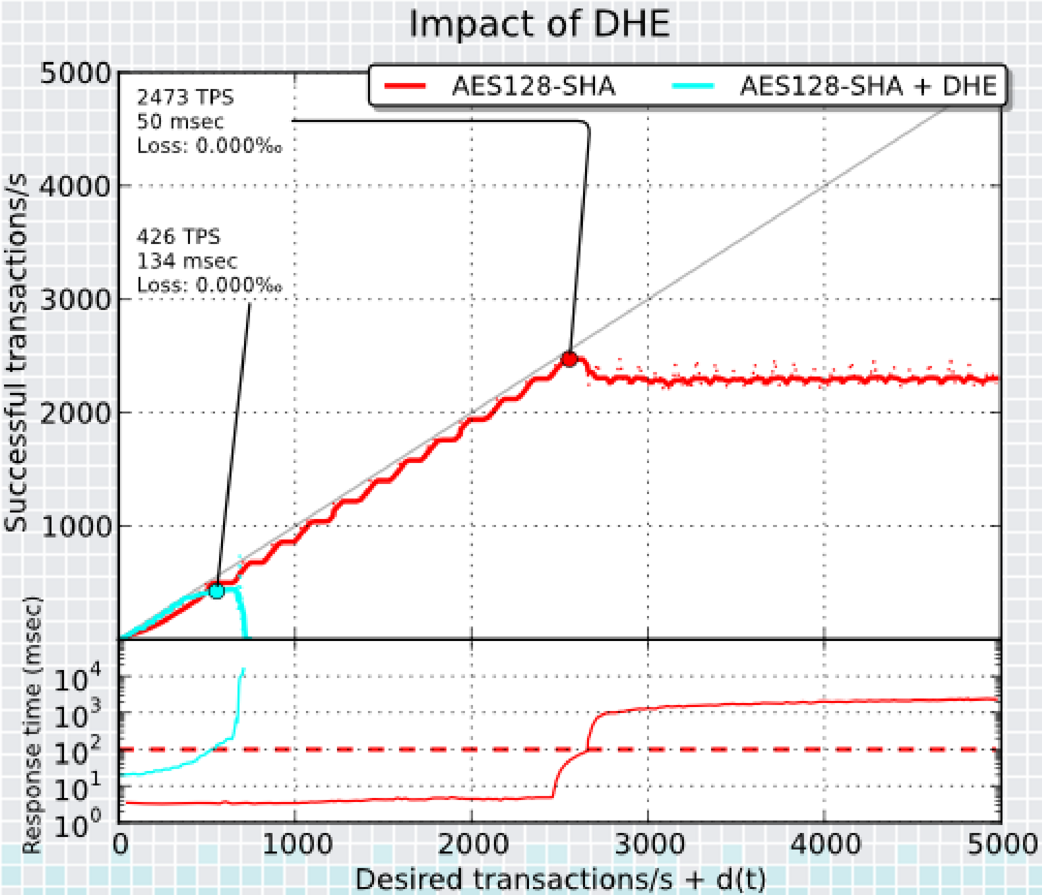
Yc=g^b mod p – this is client's public key

Where 'a' and 'b' random numbers picked up by Server and Client independently

Shared Secret = g^(ab) mod p = Ys^b mod p = Yc^a mod p

W. Diffie, M. Hellman: "New Direction in Cryptography", 1976
RFC 5246

Old DH – Performance Impact



At around 500 TPS response time for DH grows from
10ms to 10s

For traditional RSA everything runs smoothly until 2500 TPS

From Vincent Bernat’s SSL/TLS blog

DH with Elliptic Curves

SrvKeyExchange will contain EC parameters

It can be a pre-defined named curve, e.g. prime256v1, or explicitly defined curve with all necessary params:

- p – big prime, which defines a field Fp
- ECurve (α, β) ($y^2 = x^3 + \alpha x + \beta$) – short Weierstrass equation, defines E(Fp)
- ECPoint – base point G (generator)
- order - order of G (a min n for which nG is not defined)
- cofactor – order*cofactor = |E(Fp)|
- Public ECDH server key: Ys = aG

ClientKeyExchange will contain ClientECDiffieHellmanPublic with:

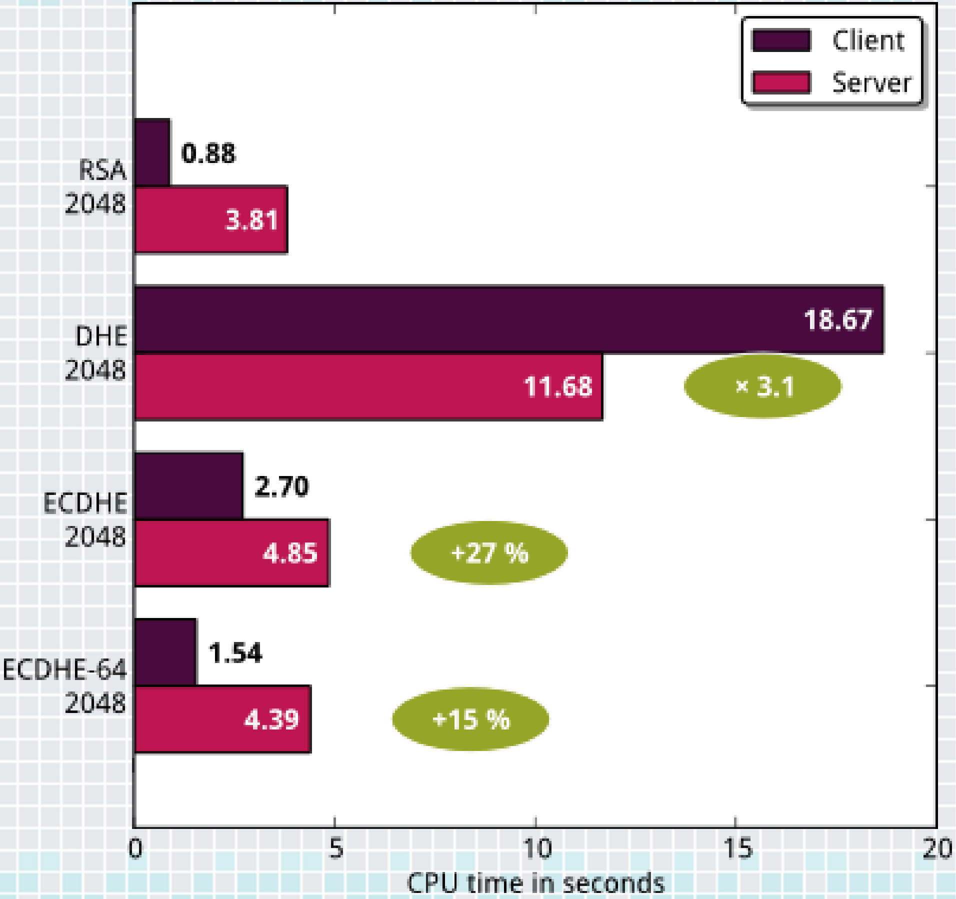
- Public ECDH client key: Yc = bG

Shared Secret = abG = aYc = bYs

An Efficient Protocol for Authenticated Key Agreement, 1998

RFC4492

ECDHE – Performance vs. RSA



On server side DHE three times slower than RSA 2048

For optimized ECDHE-64 the overhead is 15% only

From Vincent Bernat's SSL/TLS blog

TLS Cheatsheet

Handshake Algorithm	Public(*) params for session key	Private(*) params for session key	Long term key (LTK) usage	Attack complexity	Speed
Classic (RFC 5264)	Random a,b Public cert of LTK	Premaster Secret(sent encrypted) LTK	Authentication and encryption	Same as attack on RSA/DSA based PKI	Still fastest
DHE (RFC 5264)	p – big prime g – its primitive root	Random, private a,b (a & b are never sent)	Authentication only	Same as discrete logarithm problem	Times slower than RSA
ECDHE (RFC 4492)	p – big prime G – base point r – order of G k – small cofactor α– curve's param β– curve’s param	Random, private a,b (a & b are never sent)	Authentication only	Same as discrete logarithm problem	Almost the same as classical RSA

RSA®Conference2015

San Francisco | April 20-24 | Moscone Center

PFS - Grades

#RSAC

Possible PFS Implementations

As discussed, we have three major options:

- No Diffie-Hellman
- Older Diffie-Hellman without curves (DHE)
- New Diffie-Hellman with curves (ECDHE)

In addition, server can also:

- Have preferred ciphers that fall to one of the categories above
- It can support or not support newer and older DH protocols

PFS Grades

Supported	Preferred	Grade
PFS Only	ECDHE	1
PFS Only	DHE	2
PFS and non PFS	ECDHE	3
PFS and non PFS	DHE	4
DHE, ECDHE and non PFS	Non PFS	5
DHE and non PFS	Non PFS	6
PFS are not supported	Non PFS (obviously)	7

PFS Grades – More Reasoning

Why preferred ciphers are important?

◆ Client can send a list of ciphers that it supports
◆ Server will always select a preferred, even if client has a better cipher in the list

Why ECDHE vs DHE is important?

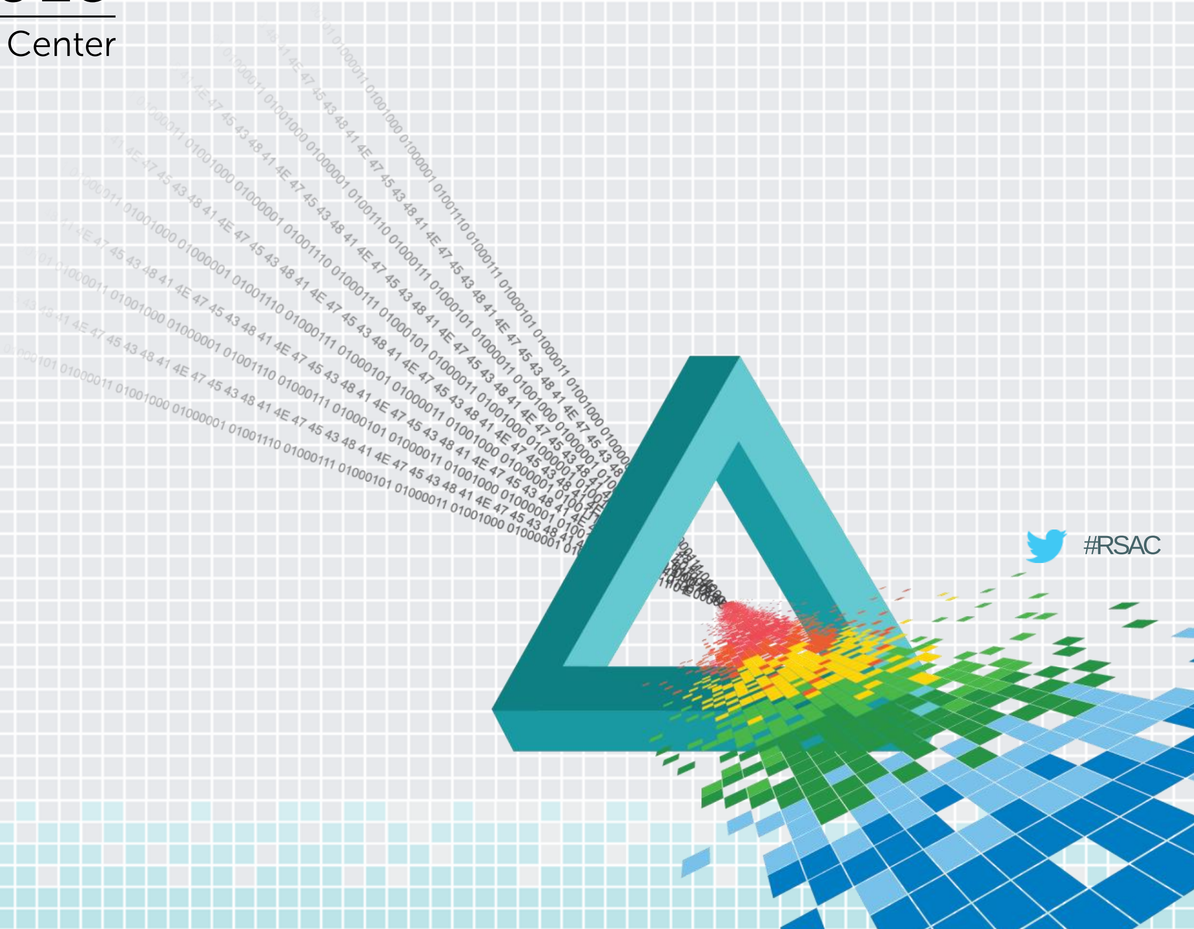
◆ Because of performance (see slides 7 and 9)
◆ If we don't care about performance, we could consider the following grades equivalent: 1 and 2, 3 and 4, 5 and 6

You can reduce the number of grades to 4 if you care about security only, but it's probably not a wise thing to do, because too many security initiatives are stopped because of “poor performance”. Example – old DHE itself vs. RSA.

RSAConference2015

San Francisco | April 20-24 | Moscone Center

PFS - Testing



#RSAC

Let us test them

Ten companies in each of the following industries have been selected:

- Manufacturing
- Finance
- Government
- InfoSec
- Defense
- Health
- Internet
- Electronics
- Education
- Software

Notes on site selections

- How – Just Googled them, e.g. “top ten health providers”

- The biggest challenge – it was difficult to find SSL protected Websites in
- Defense – everything is usually public at those ☺
- Exception – their job related portals

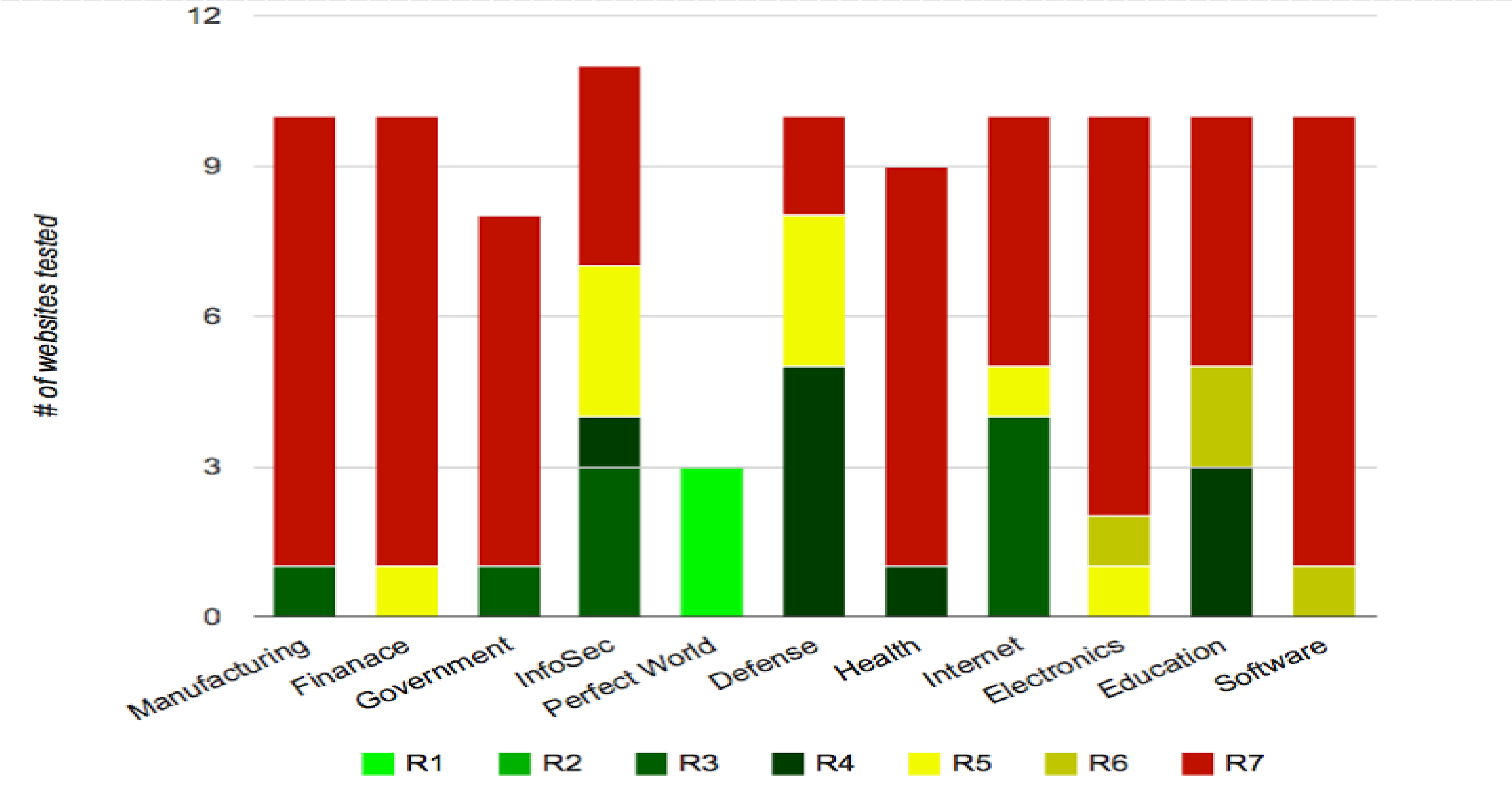
- Used a Python client with JSON configuration file

- Code for testing : sf.net/projects/pfschecker

Configuration file example

```
"statfile":"statfile.html",
"ciphers":"ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-RSA-AES256-SHA384:ECDHEECDSA-
….",
"baseline_ciphers":"AES128-SHA:RC4-MD5:RC4-SHA:AES256-SHA:DES-CBC3-SHA",
"hosts":[
{"host":"www.bank1.com","port":443,"name":"Bank One","tag":"Finanace"},
{"host":"www.bank2.com","port":443,"name":"Bank Two","tag":"Finanace"},
{"host":"www.bank3.com","port":443,"name":"Bank Three","tag":"Finanace"},
{"host":"www.bank4.com","port":443,"name":"Bank Four","tag":"Finanace"},
```

Test results



Winners and Losers

Winners:

- Internet
- InfoSec
- Defense
- Education

At least one has PFS as preferred:

- Manufacturing
- Government
- Health

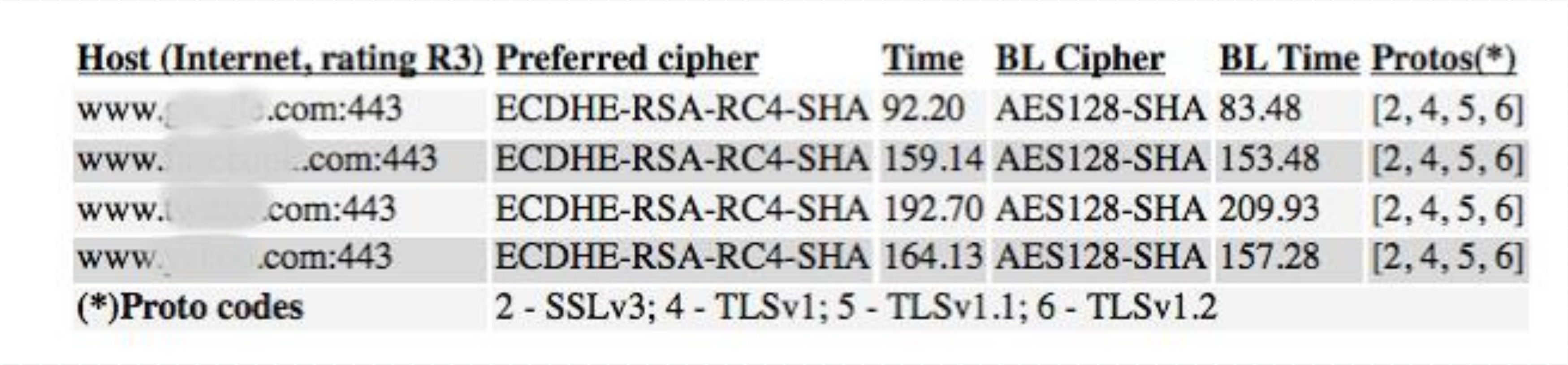
PFS not implemented as preferred:

- Finance
- Electronics
- Software

Some Thoughts:

- Finance organizations are usually very good when it comes to privacy or fraud, but do not adopt technology fast
- Internet companies might not be that good in privacy, but are quick in picking up new technologies including security
- Education/Universities are similar when it comes to innovations
- InfoSec, Defense – they ought to and could've been done even better IMO

Details for Internet Companies



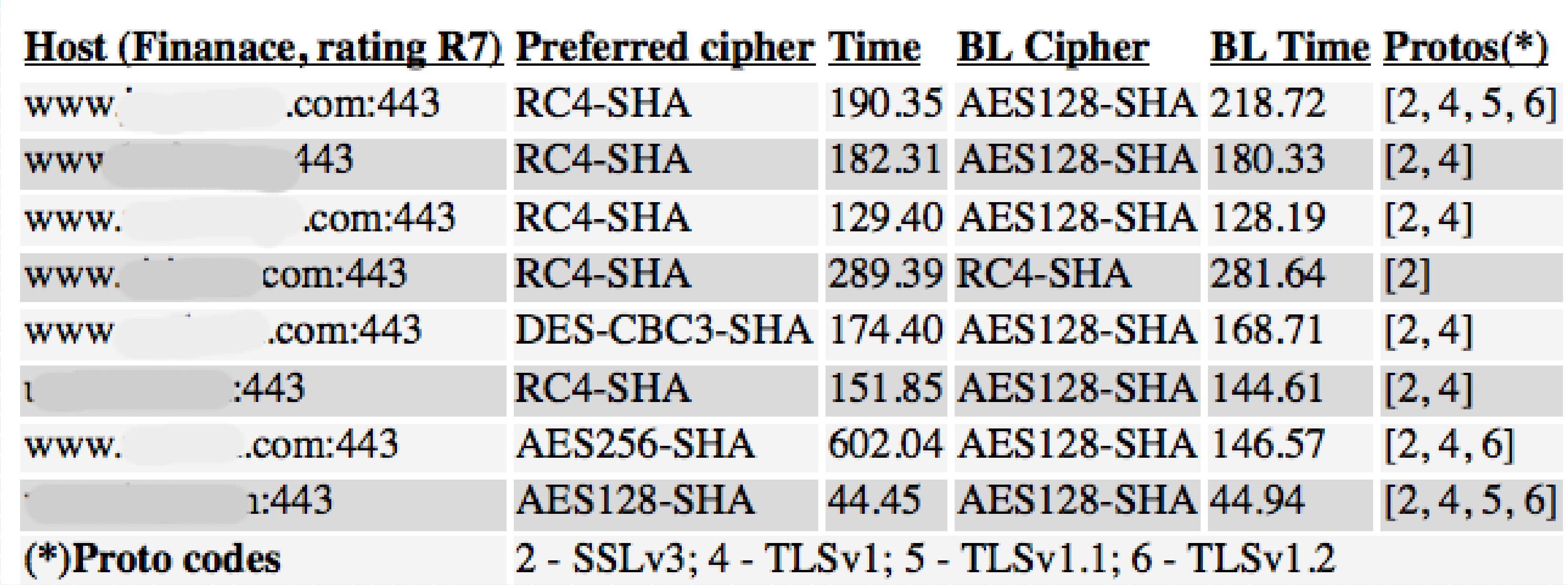
Host (Internet, rating R3)	Preferred cipher	Time	BL Cipher	BL Time	Protos(*)
www.[redacted].com:443	ECDHE-RSA-RC4-SHA	92.20	AES128-SHA	83.48	[2, 4, 5, 6]
www.[redacted].com:443	ECDHE-RSA-RC4-SHA	159.14	AES128-SHA	153.48	[2, 4, 5, 6]
www.[redacted].com:443	ECDHE-RSA-RC4-SHA	192.70	AES128-SHA	209.93	[2, 4, 5, 6]
www.[redacted].com:443	ECDHE-RSA-RC4-SHA	164.13	AES128-SHA	157.28	[2, 4, 5, 6]
(*)Proto codes	2 - SSLv3; 4 - TLSv1; 5 - TLSv1.1; 6 - TLSv1.2				

- No difference in handshake time from client point of view
- All major Internet companies graded as 3 or 4
- Everyone supports all versions of TLS
- Everyone uses the same fast preferred ECDHE cipher

Disappointment:

- SSLv3 and TLSv1 support. I would love to see only TLSv1.2

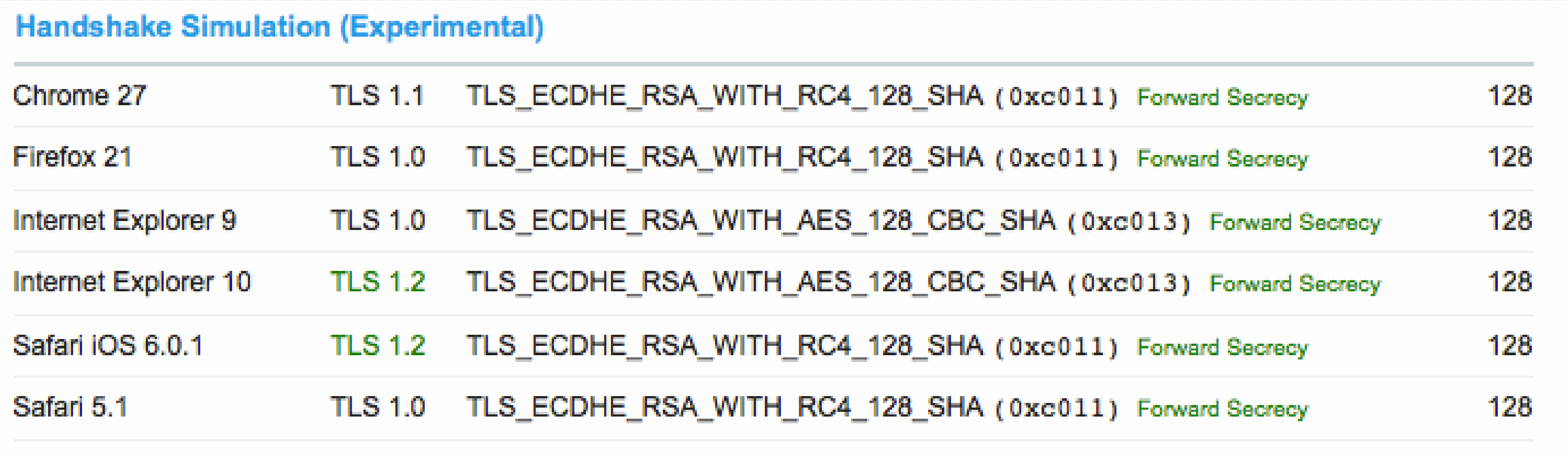
Details for Finance Sector



Host (Finanace, rating R7)	Preferred cipher	Time	BL Cipher	BL Time	Protos(*)
www.[redacted].com:443	RC4-SHA	190.35	AES128-SHA	218.72	[2, 4, 5, 6]
www[redacted]443	RC4-SHA	182.31	AES128-SHA	180.33	[2, 4]
www.[redacted].com:443	RC4-SHA	129.40	AES128-SHA	128.19	[2, 4]
www.[redacted]com:443	RC4-SHA	289.39	RC4-SHA	281.64	[2]
www[redacted].com:443	DES-CBC3-SHA	174.40	AES128-SHA	168.71	[2, 4]
[redacted]:443	RC4-SHA	151.85	AES128-SHA	144.61	[2, 4]
www.[redacted].com:443	AES256-SHA	602.04	AES128-SHA	146.57	[2, 4, 6]
[redacted]:443	AES128-SHA	44.45	AES128-SHA	44.94	[2, 4, 5, 6]
(*)Proto codes	2 - SSLv3; 4 - TLSv1; 5 - TLSv1.1; 6 - TLSv1.2				

- Too many companies (80%) don't support PFS at all (grade 7)
- Poor support for the newer TLS versions (1.1 and 1.2)

What about Browser’s Support for ECDHE

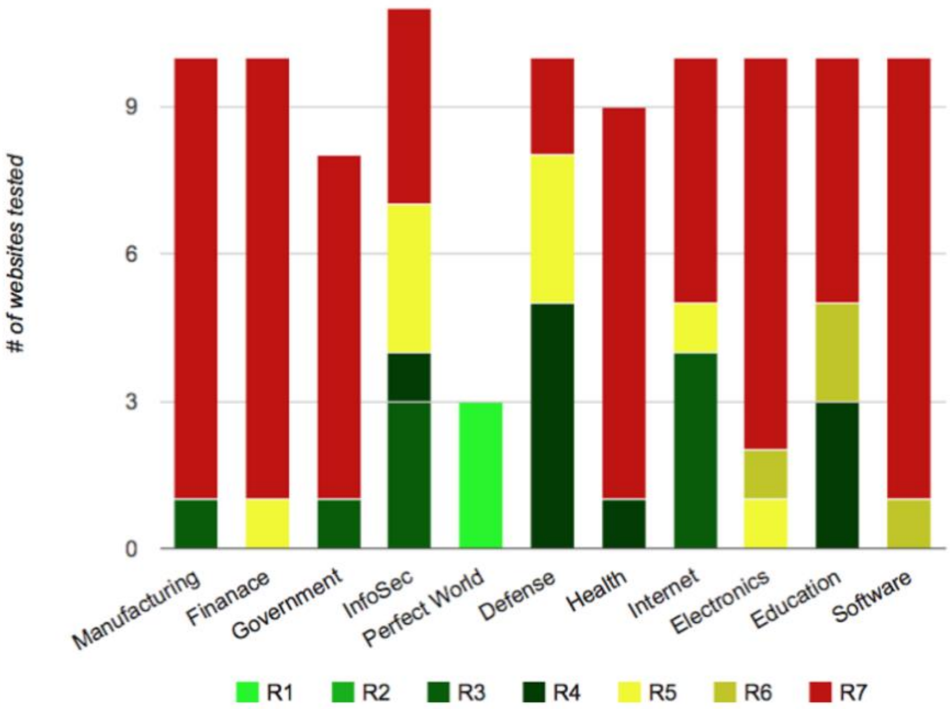


Handshake Simulation (Experimental)

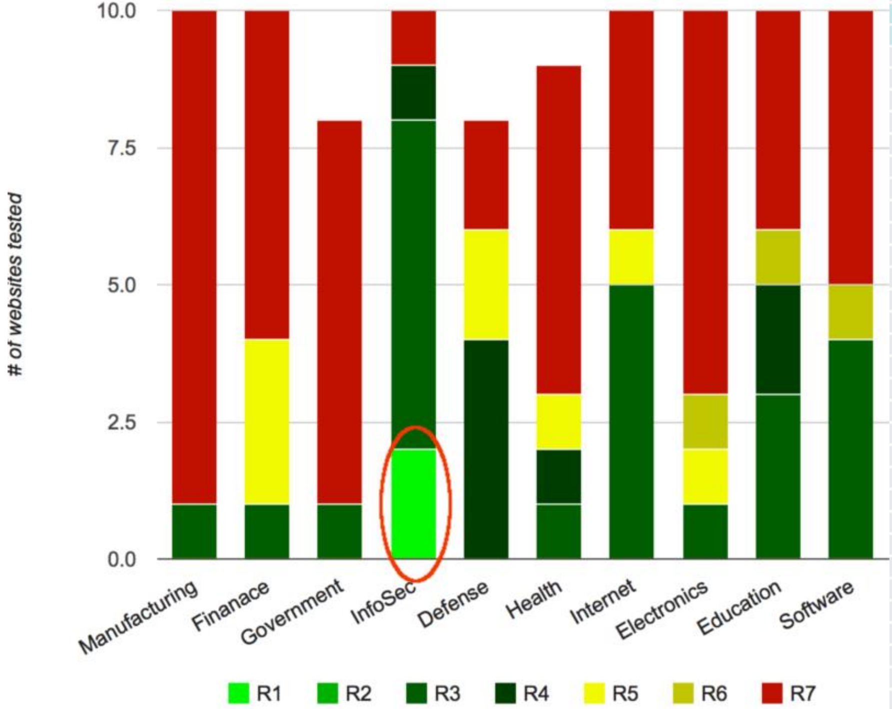
Chrome 27	TLS 1.1	TLS_ECDHE_RSA_WITH_RC4_128_SHA (0xc011)	Forward Secrecy	128
Firefox 21	TLS 1.0	TLS_ECDHE_RSA_WITH_RC4_128_SHA (0xc011)	Forward Secrecy	128
Internet Explorer 9	TLS 1.0	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)	Forward Secrecy	128
Internet Explorer 10	TLS 1.2	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)	Forward Secrecy	128
Safari iOS 6.0.1	TLS 1.2	TLS_ECDHE_RSA_WITH_RC4_128_SHA (0xc011)	Forward Secrecy	128
Safari 5.1	TLS 1.0	TLS_ECDHE_RSA_WITH_RC4_128_SHA (0xc011)	Forward Secrecy	128

[From Qualys Community Website]()

Date: 09/2014



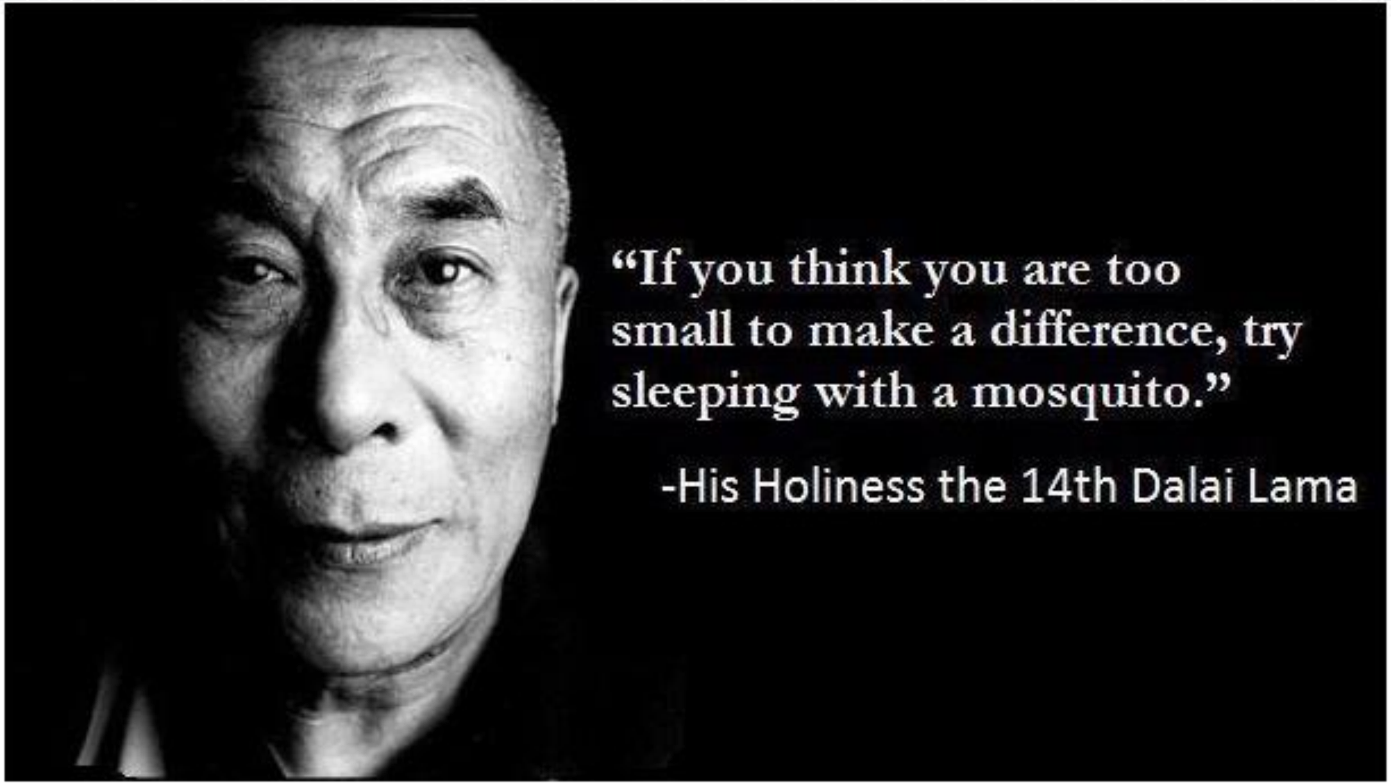
Date: 02/2015



Legend

R1	*Only PFS ciphers are supported and preferred cipher is ECDHE*
R2	*Only PFS ciphers are supported, preferred cipher is an old DHE*
R3	*PFS and non-PFS ciphers are supported. A preferred cipher is ECDHE*
R4	*PFS and non-PFS ciphers are supported. A preferred cipher is an old DHE*
R5	*PFS and non-PFS ciphers are supported including ECDHE. A preferred cipher is a non-PFS*
R6	*Old PFS (DHE) and non-PFS ciphers are supported, but ECDHE is not. A preferred cipher is a non-PFS*
R7	*PFS ciphers are not supported*

You can make a difference



"If you think you are too small to make a difference, try sleeping with a mosquito."

-His Holiness the 14th Dalai Lama

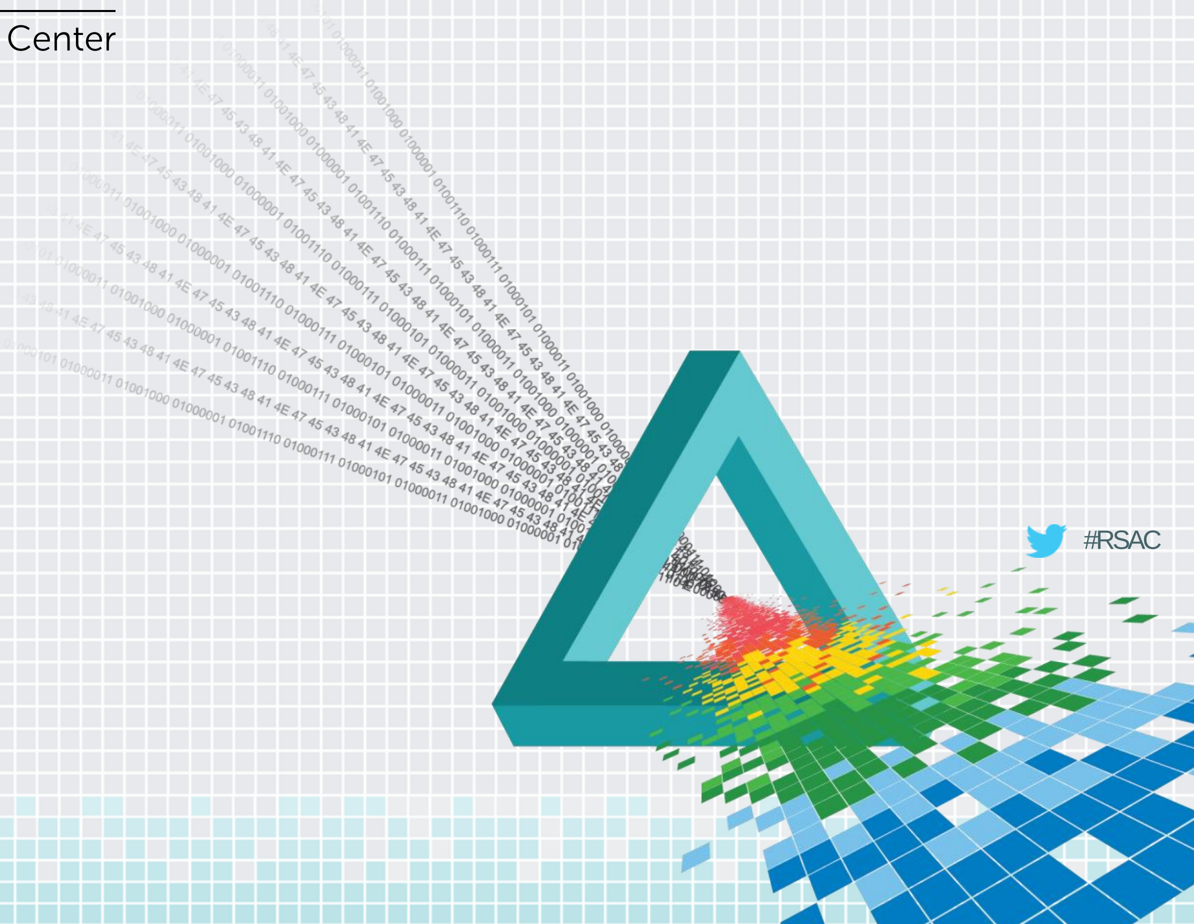
cigital

iSECpartners
part of nccgroup

RSA®Conference2015

San Francisco | April 20-24 | Moscone Center

PFS – Getting to Conclusions



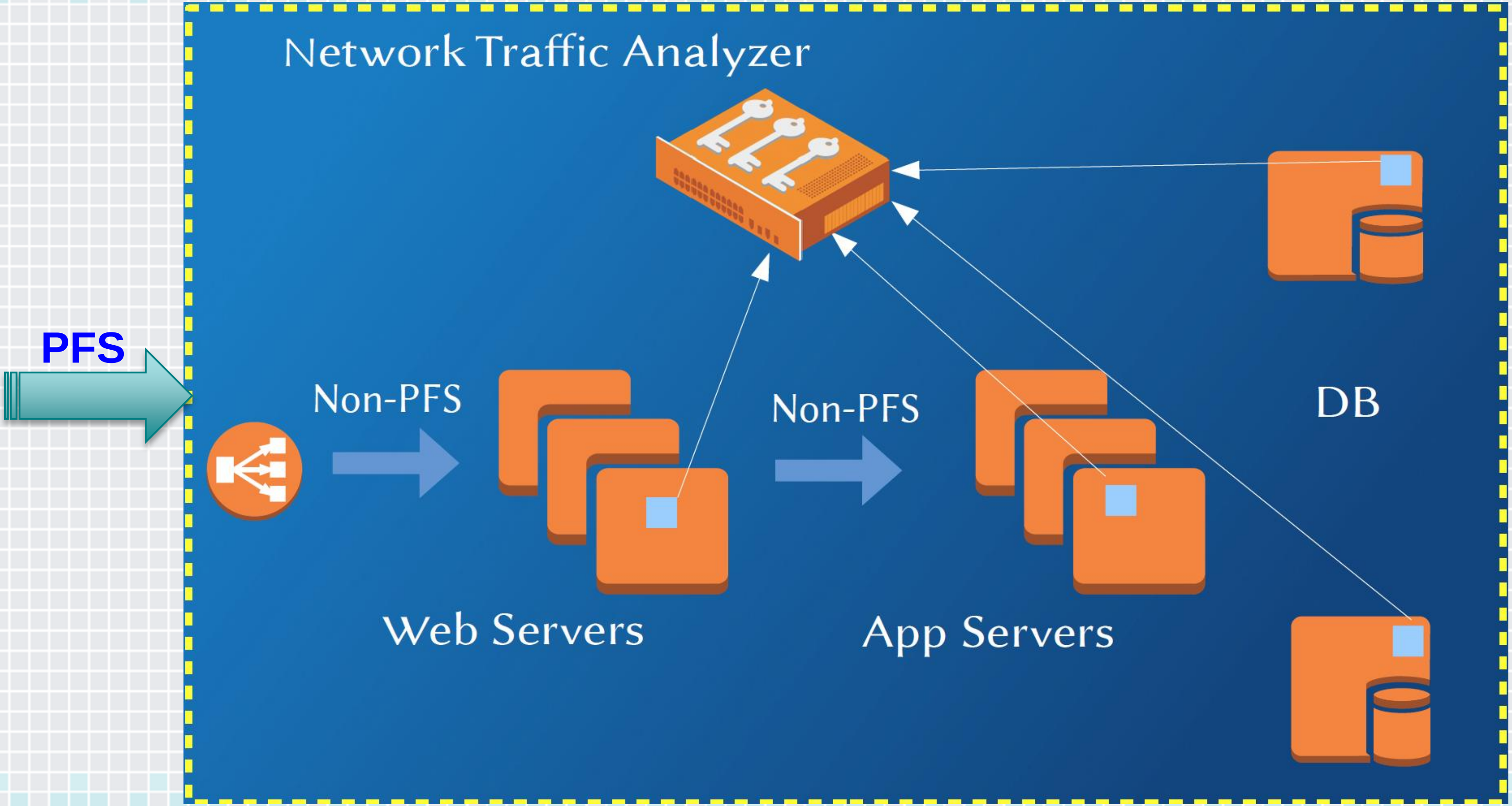
#RSAC

Conclusion

- There is no any reason why you can't move your servers to category #3 or #4 (there is a fallback on non PFS)
- To move them to the the categories #1 or #2 (there is no fallback on non-PFS) a decision about not supporting legacy browsers should be made. That decision would make a perfect sense since it'll improve the overall security of web applications.
- Other factors to consider to make a decision about not supporting “legacy browsers”:
  - They are less secure
  - You want to take the full advantage of HTML5
  - Upgrade to newer versions if usually free

Just Tell Them to Upgrade! No significant excuses have left.

There are always exceptions …



There is only one grade of perfection

ECDHE

No Fallback
on
Anything
Less Perfect

Thanks for Coming !

Q & A Time

Oleg Gryb
Sr. Manager, Security Engineering @ SSIC
Twitter: @oleggryb