



# SSL Handshake Analysis

## Computer Measurement Group Webinar

Nalini Elkins  
Inside Products, Inc.  
[nalini.elkins@insidethestack.com](mailto:nalini.elkins@insidethestack.com)



Inside Products, Inc.

(831) 659-8360

[www.insidethestack.com](http://www.insidethestack.com)

[www.ipproblemfinders.com](http://www.ipproblemfinders.com)

# Bank Hackers Steal Millions via Malware

In late 2013, an A.T.M. in Kiev started dispensing cash at seemingly random times of day.

No one had put in a card or touched a button.

Cameras showed that the piles of money had been swept up by customers who appeared lucky to be there at the right moment.

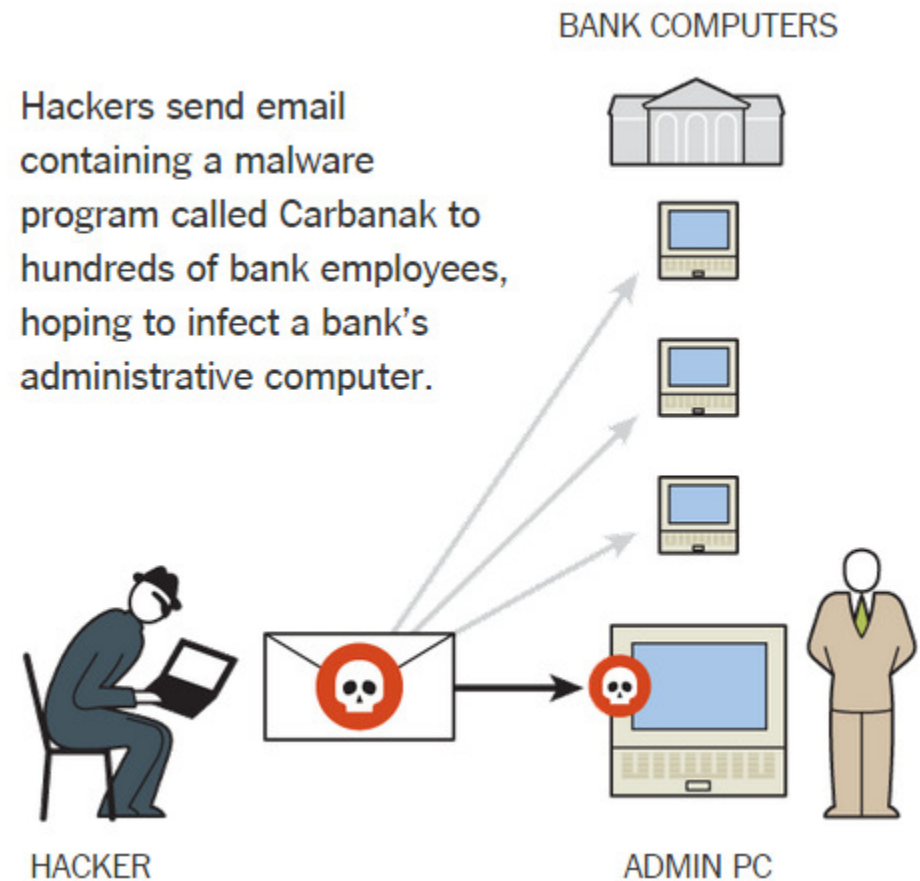
By **DAVID E. SANGER** and **NICOLE PERLROTH**  
NY Times  
FEB. 14, 2015



# Bank Hackers

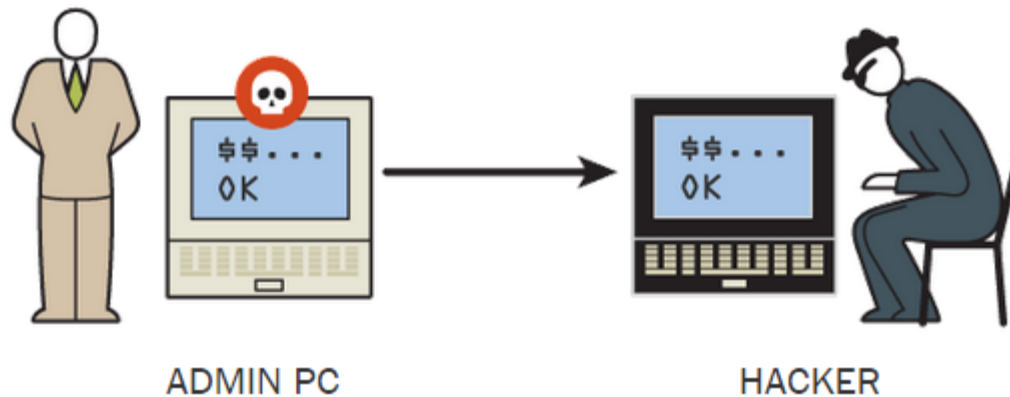
## How Hackers Infiltrated Banks

Since late 2013, an unknown group of hackers has reportedly stolen \$300 million — possibly as much as triple that amount — from banks across the world, with the majority of the victims in Russia. The attacks continue, all using roughly the same modus operandi:



# Bank Hackers

Programs installed by the malware record keystrokes and take screen shots of the bank's computers, so that hackers can learn bank procedures. They also enable hackers to control the banks' computers remotely.



# Bank Hackers

By mimicking the bank procedures they have learned, hackers direct the banks' computers to steal money in a variety of ways:

Transferring money into hackers' fraudulent bank accounts

Using e-payment systems to send money to fraudulent accounts overseas

Directing A.T.M.s to dispense money at set times and locations

# Agenda

We will diagram and discuss:

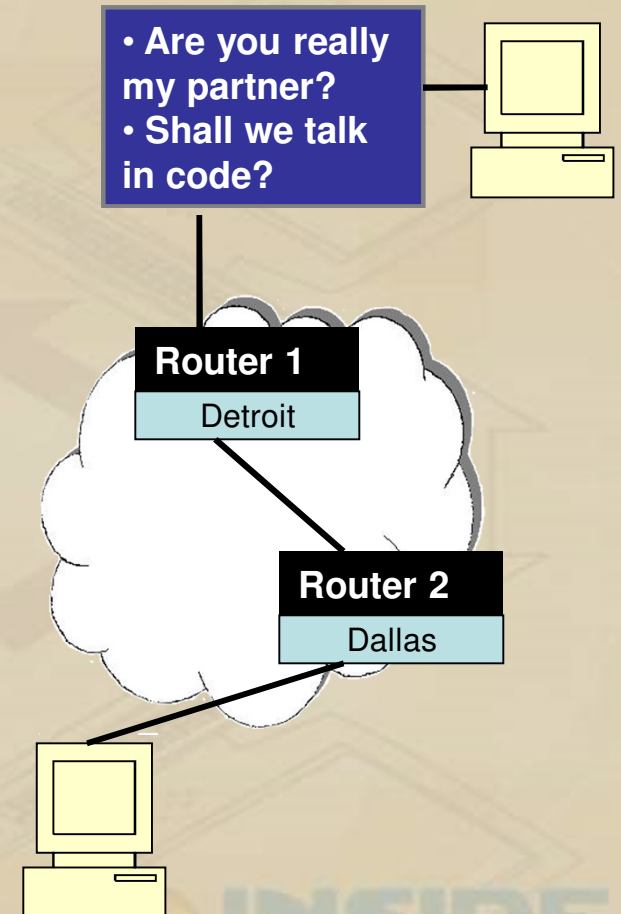
- SSL handshakes with only a server certificate,
- SSL handshakes with both server and client certificates, and
- Resumed handshakes

Then, we will move on to:

- Diagnose SSL handshake problems,
- View invalid certificate authorities and bad certificates,
- Discuss incorrect cipher suites, and
- Analyze performance issues.

# What is Secure Sockets Layer?

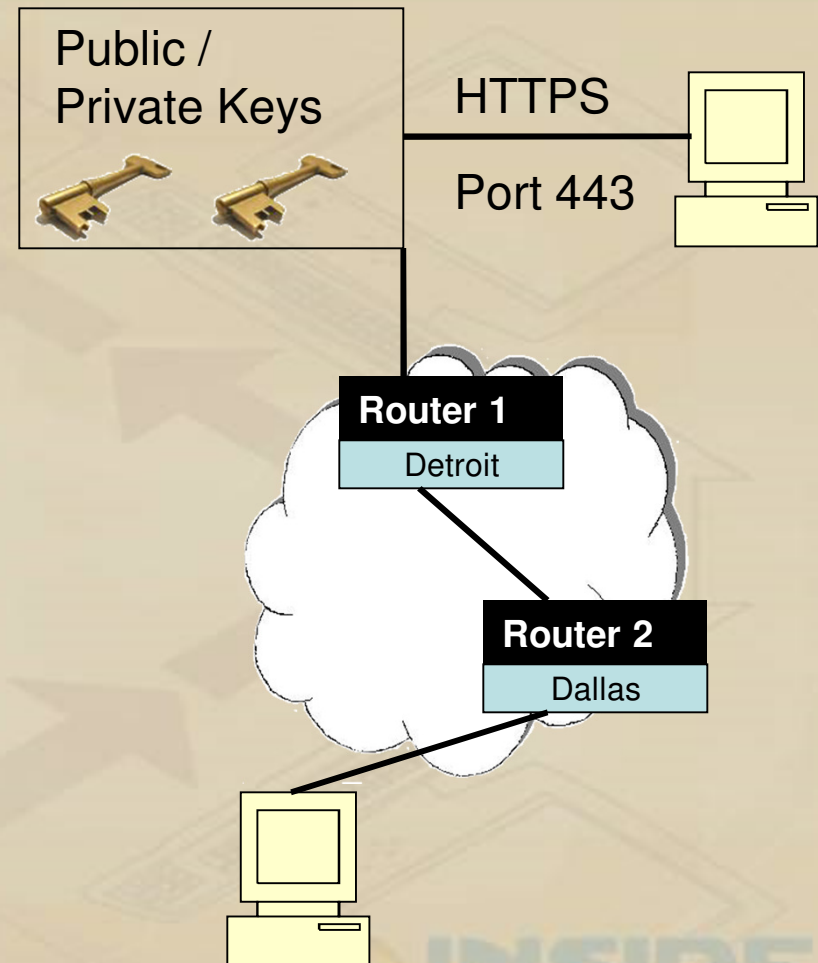
- Secure Sockets Layer (SSL) widely used protocol developed by Netscape
- Main functions:
  - Server authentication
  - Data privacy and integrity
  - Optional client authentication via digital certificate
- Multiple versions exist: SSLv2 and SSLv3. (SSLv2 prohibited in RFC6176)
- Became Internet Standard : Transport Layer Security (TLS) : RFC2246, RFC3546, RFC4346, RFC5246 etc.
- TLSv1.2 is the latest version. (TLSv1.0, 1.1 exist)
- Slight differences between SSLv3 and TLS versions.





# How Does SSL Work?

- Uses two keys:
  - public key known to everyone
  - private key known only to recipient
- URLs with SSL start with *https*: instead of *http*:
- SSL HTTP uses port 443 instead of port 80.
- Other ports (Telnet, TN3270, FTP, etc) no convention





# Key Based Algorithms

- Most modern algorithms are *key-based*.
- A *key-based algorithm* uses an *encryption key* to encrypt the message
- Receiver uses *decryption key* to decrypt the message
- Some algorithms use same key to encrypt and decrypt
- Some do not

**Encryption Key: TIGER**

**Encrypted Message:**

Scr qul wdjj gmkr mus  
smkmpmww

**Decryption Key: TIGER**

**Decrypted Message:**

The sun will come out  
tomorrow

?????

**Encryption Key: FLOWER**

**Decryption Key: TIGERS**

# Asymmetric Algorithms

- Symmetric algorithms
  - Same key to encrypt / decrypt
- Asymmetric algorithms
  - Different key to encrypt / decrypt
- Asymmetric: Public-key algorithms
  - **Private key:** Key must be known only by its owner
  - **Public key:** Key is known to everyone
  - **Relation between both keys:** What one key encrypts, the other one decrypts
- Main disadvantage: asymmetric not as fast as symmetric

*Encryption Key: TIGER*

**Encrypted Message:**

Scr qul wdjj gmkr mus  
smkmpmw

*Decryption Key: TIGER*

**Decrypted Message:**

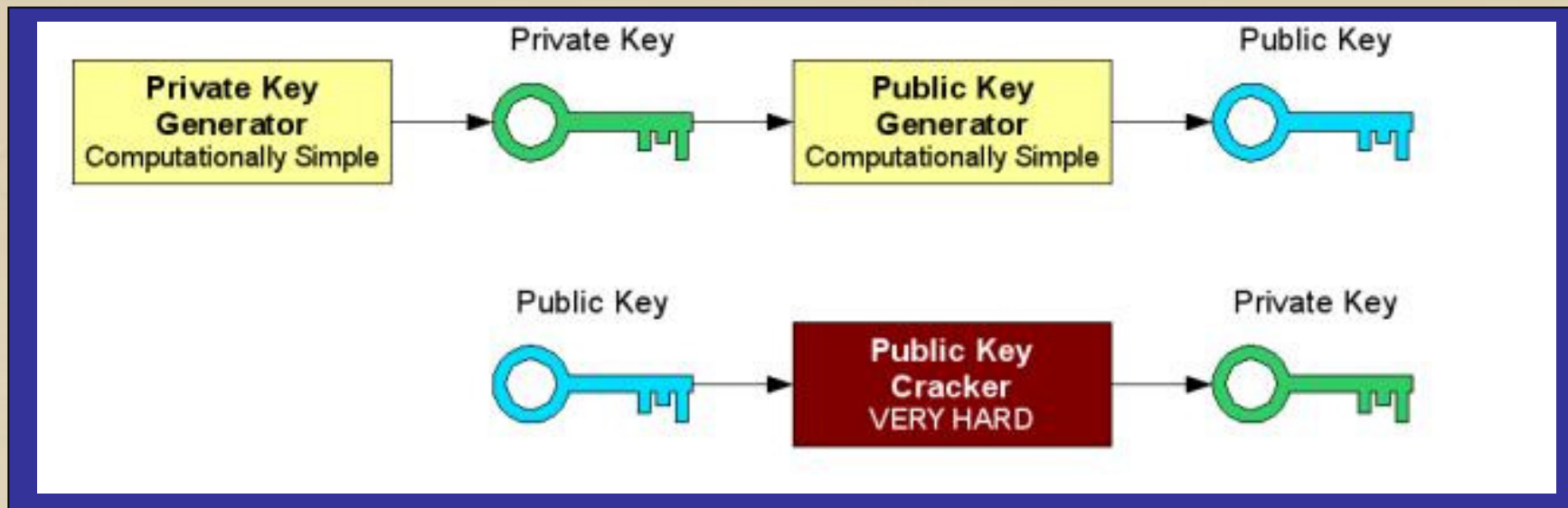
The sun will come out  
tomorrow

*Public Key: FLOWER*

*Private Key: TIGERS*

?????

# Key Generation



- Easy to compute public key from private key
- *Very hard* to compute private key from public key
- Some algorithms need *months* or even years to get private key from public key
- RSA algorithm invented in 1978 by Ron Rivest, Adi Shamir, and Leonard Adleman
- Rely on hard to invert functions
  - Factoring prime numbers
  - Modulus.

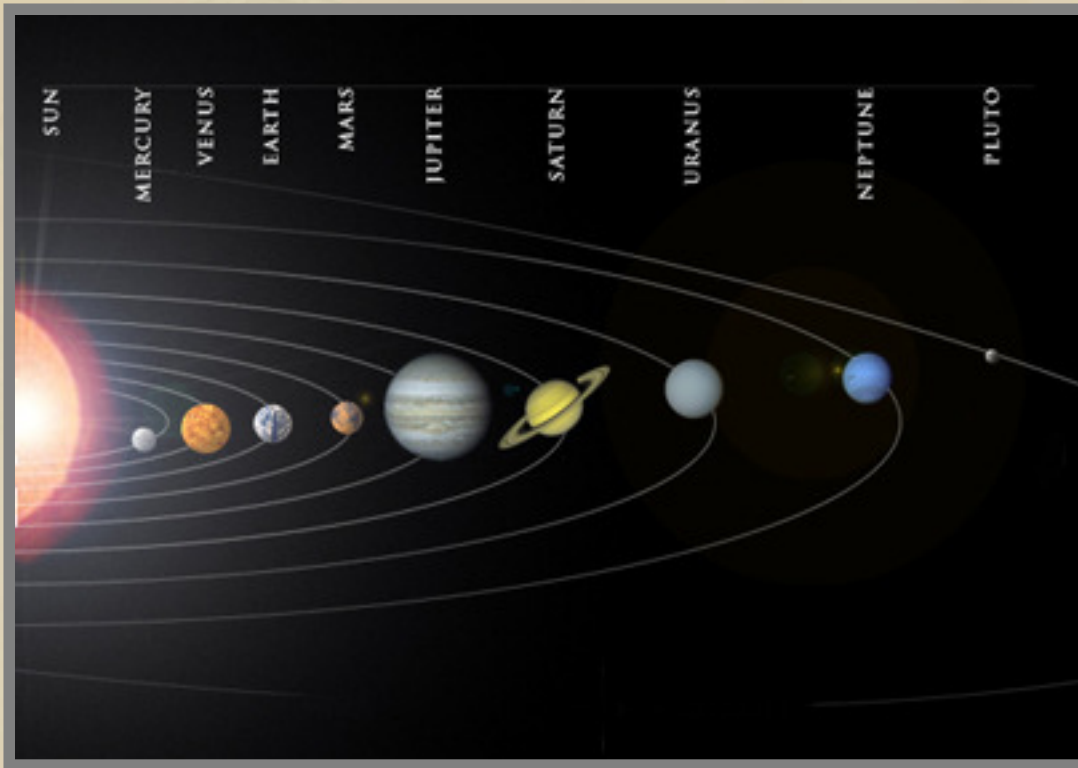
# Factoring Prime Numbers

- A **prime number** (or **prime**) has exactly two *distinct* divisors: 1 and itself.
- Smallest twenty-five prime numbers (all the prime numbers under 100) are:  
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97
- Prime factorization is a list of all the prime-number factors of a given number.
- The prime factorization does not include 1, but does include every copy of every prime factor. For instance, the prime factorization of 8 is  $2 \times 2 \times 2$ , not just "2".
- 2 is the only factor, but you need three copies of it to multiply back to 8, so the prime factorization includes all three copies

# Cracking RSA?

- Challenge was placed in Martin Gardner's column in 1977 in Scientific American. Readers invited to factor:  
 $C = 114,381,625,757,888,867,669,235,779,976,146,612,010,218,296,721,242,362,562,561,842,935,706,935,245,733,897,830,597,123,563,958,705,058,989,075,147,599,290,026,879,543,541$   
into its two prime number factors.
- First solver wins one hundred dollars! (And glory!)
- Solved 17 years later in April 26, 1994.
  - International effort 600 volunteers, workstations, mainframes, and supercomputers
  - Attacked number above for eight months
- The RSA algorithm is seen as safe.
- Numbers used are much larger

# Elliptic Curve Cryptography



- Elliptic-curve cryptography (ECC) can use smaller keys
- More efficient
- Speed and size matter
- Sensor networks, etc.

Thanks to Johannes Kepler, we know that the planets move in an elliptic orbit with the sun at one focus.



# SSL Applications

- SSL implementation: special SSL socket calls
- Transparent to TCP stack
- APIs: C/C++ or Java
- TCP applications only!
- UDP, ICMP etc not supported
- Datagram TLS (DTLS) available for UDP

SSL Socket Library:  
TCP Only





# SSL Packet Encryption

- Protects TCP packet payload
- IP header and TCP header unencrypted
- IPSec encrypts entire packet

No. -	Time	Source	Destination	Protocol	Info
259	167.343036	66.218.70.70	192.168.1.101	SSLv3	Application Data
+ Frame 258 (54 bytes on wire, 54 bytes captured)					
+ Ethernet II, Src: AsustekC_39:29:2b (00:11:d8:39:29:2b), Dst: LinksysG_e4:ae:3					
+ Internet Protocol, src: 192.168.1.101 (192.168.1.101), Dst: 66.218.70.70 (66.2					
- Transmission Control Protocol, Src Port: 2259 (2259), Dst Port: https (443), s					
Source port: 2259 (2259)					
Destination port: https (443)					
Sequence number: 588 (relative sequence number)					
Acknowledgement number: 3825 (relative ack number)					
Header length: 20 bytes					

# SSL Packet and Flow

- Typical SSL packet may contain:
  - SSL flags : type of *SSL message*.
  - Cryptographic integrity checking (typically using the MD5 or SHA-1 algorithm).
  - Encrypted data (typically using the RC2, RC4, DES, or Triple DES algorithms).
- SSL connection begins with handshake (asymmetric cryptography)
- Followed by data transfer (also called the *SSL record protocol*) uses symmetric cryptography

## SSL Packet

Source Address : Port

Dest Address : Port

SSL  
Flag

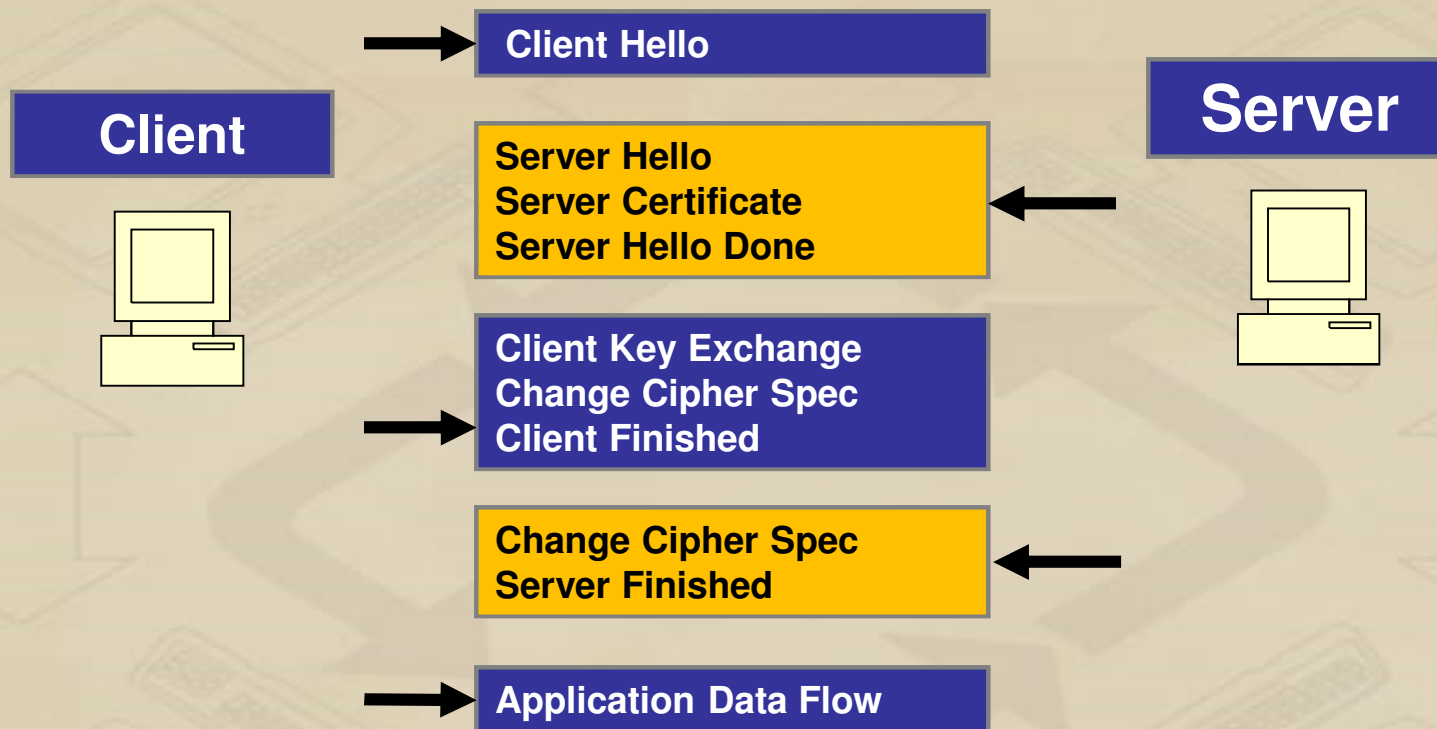
Integrity  
Checking

Encrypted  
Data

# SSL Handshake

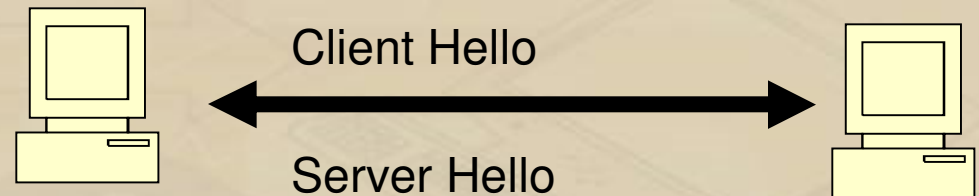
- SSL handshake: initial setup
- Exchange of information
- Includes:
  - Authentication of server
  - Decision on how data is to be encrypted
  - Optionally, authentication of client
- No encryption used initially

# Packets in SSL Handshake Server Certificate Only

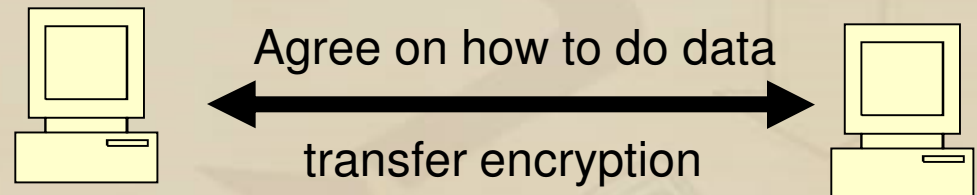


# SSL Protocol Exchanges

- The client connects to the server indicating that it wants to perform SSL. Contains sessionID. Server agrees ("server hello"). Handshake begins.



- The client and server agree on a common symmetric algorithm to be used for the data transfer that follows the handshake. Both have a list of possible algorithms in the order of preference



# Client Hello

## Secure Sockets Layer

### TLSv1 Record Layer: Handshake Protocol: Client Hello

Content Type: Handshake (22)


Version: TLS 1.0 (0x0301)

Length: 92

### Handshake Protocol: Client Hello

Handshake Type: Client Hello (1)

Length: 88

Version: TLS 1.0 (0x0301) 

### Random

Session ID Length: 0

Cipher Suites Length: 12

### Cipher Suites (6 suites)

Cipher Suite: TLS\_RSA\_WITH\_CAMELLIA\_256\_CBC\_SHA (0x0084)

Cipher Suite: TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA (0x0035)

Cipher Suite: TLS\_RSA\_WITH\_CAMELLIA\_128\_CBC\_SHA (0x0041)

Cipher Suite: TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA (0x002f)

Cipher Suite: SSL\_RSA\_FIPS\_WITH\_3DES\_EDE\_CBC\_SHA (0xfeff)

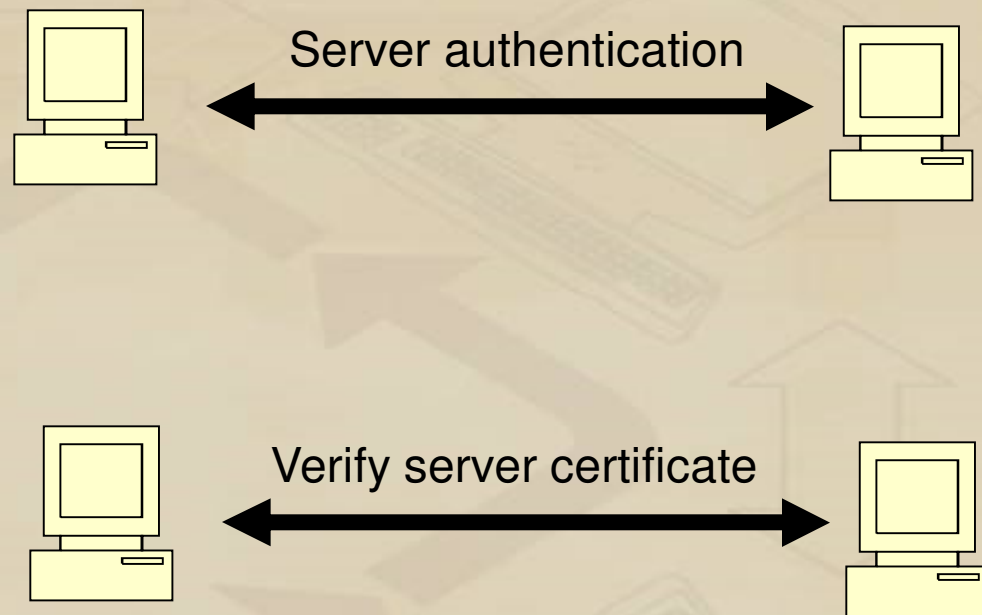
Cipher Suite: TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA (0x000a)

Compression Methods Length: 1

### Compression Methods (1 method)

# SSL Protocol Exchanges

- The server provides its public key in its certificate to the client-this is also called *server authentication*, and is a required step of the SSL handshake.
- The client verifies the integrity of the server's certificate. Depending on the client design, if this certificate is not available, the client may ask the end user to agree to either pursue the communication or to abort it.





# Server Hello

- Secure Sockets Layer

- TLSv1 Record Layer: Handshake Protocol: Server Hello



- Content Type: Handshake (22)

- Version: TLS 1.0 (0x0301)

- Length: 81

- Handshake Protocol: Server Hello

- Handshake Type: Server Hello (2)

- Length: 77

- Version: TLS 1.0 (0x0301)

- Random

- GMT Unix Time: Jan 5, 2015 13:07:51.000000000 Pacific Standard Time

- Random Bytes: 8f9d2309818b0cf23f3269a632c7016924b867d84b75f38e...

- Session ID Length: 32

- Session ID: 0c1900002176570af679c93a5988a6590d9243e9c843912f...

- Cipher Suite: TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA (0x002f)



- Compression Method: null (0)

# Server Certificate

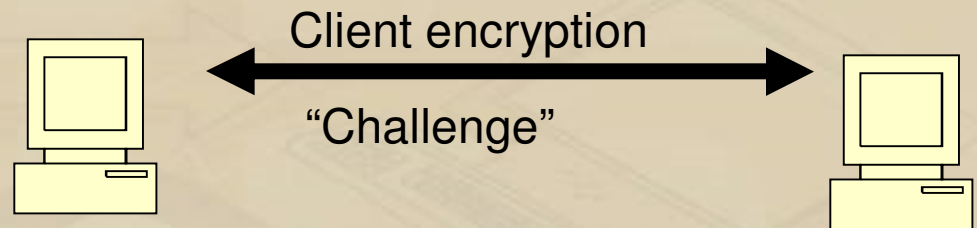
```
[-] Handshake Protocol: Certificate
    Handshake Type: Certificate (11)
    Length: 923
    Certificates Length: 920 ←
[-] Certificates (920 bytes)
    Certificate Length: 917
[-] Certificate (id-at-commonName=Nalini Elkins,id-at-organizationalUnitName=Engineering,i
    [-] signedCertificate
        version: v3 (2)
        serialNumber: 172846151
        [+ signature (sha256WithRSAEncryption)
        [-] issuer: rdnSequence (0)
            [+ rdnSequence: 6 items (id-at-commonName=Nalini Elkins,id-at-organizationalUnitName=
        [-] validity
            [-] notBefore: utcTime (0)
                utcTime: 14-05-10 13:01:12 (UTC)
            [-] notAfter: utcTime (0)
                utcTime: 15-05-05 13:01:12 (UTC)
        [+ subject: rdnSequence (0)
        [+ subjectPublicKeyInfo
        [+ extensions: 1 item
```



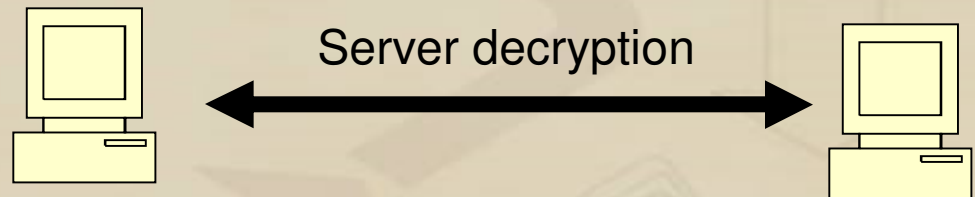
**INSIDE**  
PRODUCTS  
*Thinking Inside the Box*

# SSL Protocol Exchanges

- The client now has the server's public key and uses it to encrypt a random number, which is then sent to the server.

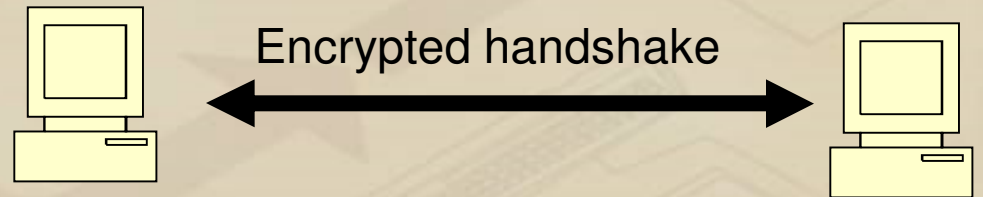
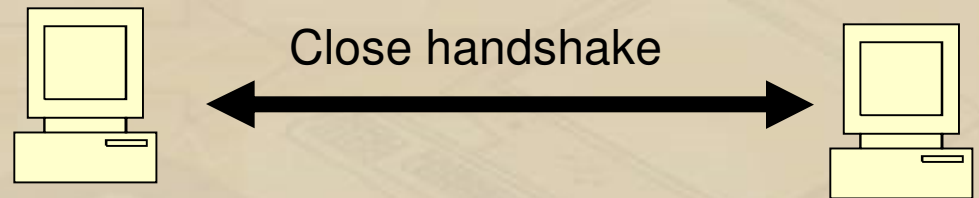


- The server retrieves the value of this random number using its private key. The decryption of the secret random number using the server's private key can cost a great deal of computing resource during the SSL handshake.



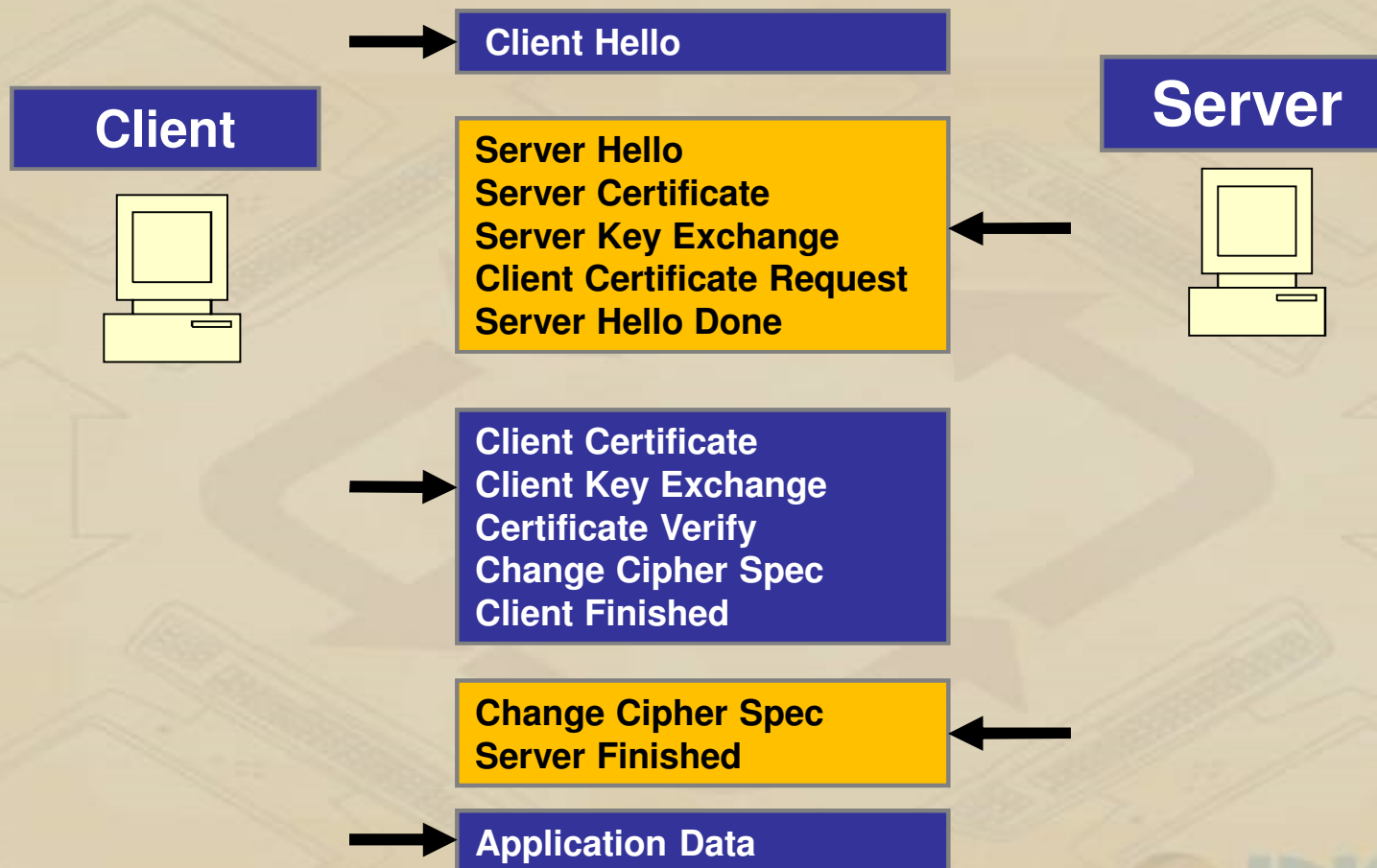
# SSL Protocol Exchanges

- At this point only the client and server know this secret random number (the pre master secret), which can then be used to generate the keys to encrypt and decrypt the data, using the symmetric algorithm previously selected. The client and the server then close the handshake phase.
- The data transfer phase (also called SSL record protocol) begins.



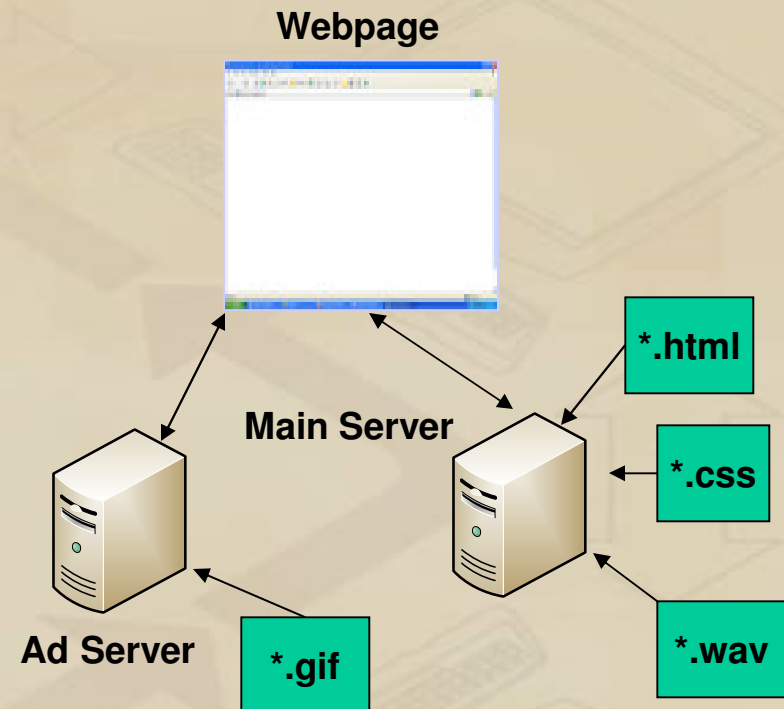
# Packets in SSL Handshake

## Both Server and Client Certificates



# When to Handshake?

- When do SSL handshakes happen?
- How can we reduce the amount of these costly handshakes?
- A handshake happens each time a TCP connection is started by the client, which means that applications which keep the connection open for a long time, such as Telnet, do handshakes rarely.
- On the other hand, protocols that open and close connection frequently, such as http, are costly in handshakes.
- The use of persistent sessions with http 1.1 reduces the number of handshakes.



# Session ID Reuse

- The server can be set up to remember the pre-master secret previously established with the client and the corresponding sessionID.
- When a client re-establishes a TCP connection to resume an SSL conversation with a server, it presents the previous sessionID

```
SSLv2 Record Layer: Client Hello
Length: 76
Handshake Message Type: Client Hello (1)
Version: SSL 3.0 (0x0300)
Cipher Spec Length: 51
Session ID Length: 0
Challenge Length: 16
```

```
Handshake Protocol: Server Hello
Handshake Type: Server Hello (2)
Length: 70
Version: SSL 3.0 (0x0300)
Random.gmt_unix_time: Mar 25, 2006
Random.bytes
Session ID Length: 32
Session ID (32 bytes)
```



# Session Tickets : RFC 5077

## Transport Layer Security (TLS) Session Resumption without Server-Side State

### Abstract

This document describes a mechanism that enables the Transport Layer Security (TLS) server to resume sessions and avoid keeping per-client session state. The TLS server encapsulates the session state into a ticket and forwards it to the client. The client can subsequently resume a session using the obtained ticket.

# Certificate Expired

```
+ Frame 9: 61 bytes on wire (488 bits), 61 bytes captured (488 bits)
+ Ethernet II, Src: Vmware_c0:00:01 (00:50:56:c0:00:01), Dst: Vmware_5d:c5:66 (0
+ Internet Protocol Version 4, Src: 192.168.3.1 (192.168.3.1), Dst: 192.168.3.3
+ Transmission Control Protocol, Src Port: 27577 (27577), Dst Port: 443 (443), :
+ Secure Sockets Layer
  + TLSv1 Record Layer: Alert (Level: Fatal, Description: Certificate Expired)
    Content Type: Alert (21)
    Version: TLS 1.0 (0x0301)
    Length: 2
  + Alert Message
    Level: Fatal (2)
    Description: Certificate Expired (45)
```

- Relatively simple problem
- Certificate expired

# Certificate Revoked

```
⊕ Frame 12: 61 bytes on wire (488 bits), 61 bytes captured (488 bits)
⊕ Ethernet II, Src: Vmware_5d:c5:66 (00:0c:29:5d:c5:66), Dst: Vmware_c0:00:01 (
⊕ Internet Protocol Version 4, Src: 192.168.3.4 (192.168.3.4), Dst: 192.168.3.1
⊕ Transmission Control Protocol, Src Port: 443 (443), Dst Port: 32123 (32123),
⊖ Secure Sockets Layer
  ⊖ TLSv1 Record Layer: Alert (Level: Fatal, Description: Certificate Revoked)
    Content Type: Alert (21)
    Version: TLS 1.0 (0x0301)
    Length: 2
    ⊖ Alert Message
      Level: Fatal (2)
      Description: Certificate Revoked (44)
```

- Using Certification Revocation
- Again, relatively simple

# No Matching Encryption Algorithm

Source	Destination	Protocol	Total Length	Info
192.168.1.108	216.219.117.244	TLSv1	142	Client Hello
216.219.117.244	192.168.1.108	TCP	40	443→1991 [ACK] Seq=1 Ack=103 win=5840 Len=0
216.219.117.244	192.168.1.108	TLSv1	126	Server Hello, Server Hello Done
192.168.1.108	216.219.117.244	TLSv1	61	Finished

## Handshake Protocol: Client Hello

### Cipher Suites (11 suites)

Cipher Suite: TLS\_RSA\_WITH\_RC4\_128\_MD5 (0x0004)  
Cipher Suite: TLS\_RSA\_WITH\_RC4\_128\_SHA (0x0005)  
Cipher Suite: TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA (0x000a)  
Cipher Suite: TLS\_RSA\_WITH\_DES\_CBC\_SHA (0x0009)  
Cipher Suite: TLS\_RSA\_EXPORT1024\_WITH\_RC4\_56\_SHA (0x0064)  
Cipher Suite: TLS\_RSA\_EXPORT1024\_WITH\_DES\_CBC\_SHA (0x0062)  
Cipher Suite: TLS\_RSA\_EXPORT\_WITH\_RC4\_40\_MD5 (0x0003)  
Cipher Suite: TLS\_RSA\_EXPORT\_WITH\_RC2\_CBC\_40\_MD5 (0x0006)  
Cipher Suite: TLS\_DHE\_DSS\_WITH\_3DES\_EDE\_CBC\_SHA (0x0013)  
Cipher Suite: TLS\_DHE\_DSS\_WITH\_DES\_CBC\_SHA (0x0012)  
Cipher Suite: TLS\_DHE\_DSS\_EXPORT1024\_WITH\_DES\_CBC\_SHA (0x0063)

## Handshake Protocol: Server Hello

Handshake Type: Server Hello (2)

Cipher Suite: TLS\_RSA\_WITH\_NULL\_SHA (0x0002)

Compression Method: null (0)

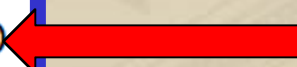
Extensions Length: 1025

## Handshake Protocol: Server Hello Done

Handshake Type: Server Hello Done (14)


Length: 0


**Not in  
Client  
Hello**



# 95% Overhead!

Drill Down	Total Server Application Data Records	Server Packets with 1 Application Data Record Per Packet	Server Packets with More Than 1 Application Data Record Per Packet	Server Packets with More Than 10 Application Data Records Per Packet	Server Average Application Data Length
	148,633	145,807	2,826	2,826	31

Total Server Application Data Records	Server Sessions Average Number of Application Data Records Per Packet	Server Sessions Packets w/More Than 1 Application Data Record Per Packet	Server Sessions Packets w/More Than 10 Application Data Records Per Packet	Server Average Application Data Length	Server Name	Total Session Time (Estimate)
7	1	0	0	57		2.115.576
148,626	52	2,826	2,826	69		66.552.658

Application Data Records	Avg Application Data Record Length	SSL
52	 21	Application Data
53	21	Application Data
52	21	Application Data
53	21	Application Data

# Encrypted vs. Unencrypted File Size

Bytes transferred with encryption: 47.682M

Bytes in original unencrypted file: 1.836M

Bytes in one SSL application data record: 21

Bytes overhead in SSL application data record: 20

-----  
Actual bytes encrypted in each SSL app. data record = 1 byte



## Secure Sockets Layer

TLShv1 Record Layer: Handshake Protocol: Encrypted Handshake Message  
Content Type: Handshake (22)  
Version: TLS 1.0 (0x0301)  
Length: 36  
Handshake Protocol: Encrypted Handshake Message

TLShv1 Record Layer: Application Data Protocol: Application Data  
Content Type: Application Data (23)  
Version: TLS 1.0 (0x0301)  
Length: 21  
Encrypted Application Data: d22fcea155448dc89b6c8a38a69c5764eb27d4288f6

TLShv1 Record Layer: Application Data Protocol: Application Data  
Content Type: Application Data (23)  
Version: TLS 1.0 (0x0301)  
Length: 21  
Encrypted Application Data: c5122f341a2bdc8276f12799fbcfc6f39220946b37

TLShv1 Record Layer: Application Data Protocol: Application Data  
Content Type: Application Data (23)  
Version: TLS 1.0 (0x0301)  
Length: 21  
Encrypted Application Data: 2aa944eba04cf680c0d2b7c15851c1355bb8b81660

TLShv1 Record Layer: Application Data Protocol: Application Data  
Content Type: Application Data (23)  
Version: TLS 1.0 (0x0301)  
Length: 21  
Encrypted Application Data: 3d50bade465c59ffc40aabeb87ae8c69b6ce7a87b6

TLShv1 Record Layer: Application Data Protocol: Application Data  
Content Type: Application Data (23)  
Version: TLS 1.0 (0x0301)  
Length: 21  
Encrypted Application Data: 80bb2ab4f8317680b8e92be8ec59e3ffbfdd4e7571

TLShv1 Record Layer: Application Data Protocol: Application Data  
Content Type: Application Data (23)  
Version: TLS 1.0 (0x0301)  
Length: 21  
Encrypted Application Data: 09aa30b68debae4b4714ca9bc9c136fb01416e8d47

TLShv1 Record Layer: Application Data Protocol: Application Data  
Content Type: Application Data (23)  
Version: TLS 1.0 (0x0301)  
Length: 21  
Encrypted Application Data: 3c6383717eaf23da03e32fa92376343e5c718c14b1

TLShv1 Record Layer: Application Data Protocol: Application Data

**Sending over 50  
application data  
records per  
packet with 20  
bytes overhead  
per each  
application data  
record!**



# Timing Problems in SSL Handshake

Total SSL Handshakes	Total Good SSL Handshakes	Total Bad SSL Handshakes	Total Resumed Handshakes	Minimum SSL Handshake Time (Seconds:Milli:Micro)	Average SSL Handshake Time (Seconds:Milli:Micro)	Maximum SSL Handshake Time (Seconds:Milli:Micro)
1	1	0	0	758.435.936	758.435.936	758.435.936

- Is it generating the Client Hello?
- Is it sending certificates?
- Is it validating the Server Certificate?
- Is it when there are multiple handshakes at the same time?

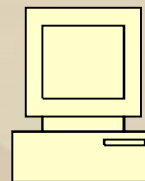
# Slowdown before Client Hello

- TCP SYN / SYN-ACK sequence completed properly

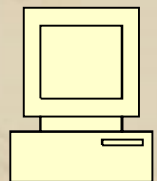


- Initial seeding of algorithms
- Start up costs







- The client connects to the server indicating that it wants to perform SSL. Contains sessionID.







Client Hello



# Slowdown before Client Hello

-	-	Packet Number	Packet Date	Source Address	Source Port	Destination Address	Destination Port	Data Length	Delta (ss.milli.micro)	Protocol
1		41	2014-02-03 10:02:25.978843		5565		3088	0	0.000.000	SYN
2		43	2014-02-03 10:02:26.051815		3088		5565	0	0.072.972	SYN, ACK
3		44	2014-02-03 10:02:26.051913		5565		3088	0	0.000.098	ACK
4		46	2014-02-03 10:02:26.730605		5565		3088	104	0.678.790	ACK, PSH

Port	5565	5565	5565	5565
TCP Protocol	<b>Ready!</b>  TCP Open Connection	<b>Set!</b>  TCP Response to Open	<b>Go!</b>  TCP window update low:262144 TCP open successful	 Your turn TCP
Application Protocol				Client Hello Session ID Length:0
Time Between Packets	0.000.000	0.072.972	0.000.098	0.678.790

Client Hello



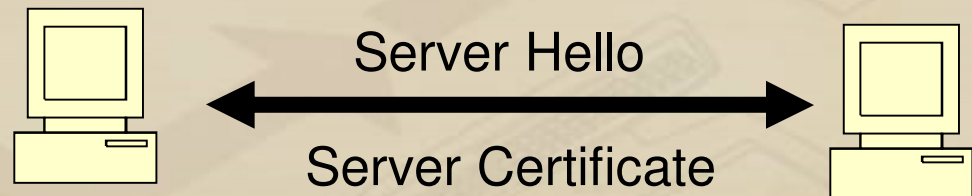
# Slowdown before Server Hello and Certificate

- Client Hello sent – SSL session requested



- Finding of algorithm
- Searching session restart
- Concurrent sessions
- Certificate chain creation







- Server agrees ("server hello"). Client and server agree on a symmetric algorithm.
- The server provides its public key in its certificate to the client-this is also called *server authentication*.



# Slowdown before Server Hello and Certificate

Source Port	Destination Port	Client Hello Time	Server Hello Time	Delta Client / Server Hello Time (SS.TTT.MMM)	Total Handshake Time (SS.TTT.MMM)
5563	3050	2014-02-03 10:02:24.842611	10:02:25.100063	0.257.452	0.302.704
5565	3088	2014-02-03 10:02:26.730605	10:02:26.778007	0.047.402	0.085.114

## Chained Certificates

-	-	Packet Number	Packet Date	Source Port	Destination Port	Certificate Authority	Not Valid Before	Not Valid After
1		14	2014-02-03 10:02:25.100063	3050	5563	VeriSign, Inc.	2006-11-08 00:00:00.0	2036-07-16 23:59:59.0
2		14	2014-02-03 10:02:25.100063	3050	5563	VeriSign, Inc.	2010-02-08 00:00:00.0	2020-02-07 23:59:59.0
3		14	2014-02-03 10:02:25.100063	3050	5563	GXS	2013-03-26 00:00:00.0	2016-03-25 23:59:59.0
4		52	2014-02-03 10:02:26.778007	3088	5565	VeriSign, Inc.	2006-11-08 00:00:00.0	2036-07-16 23:59:59.0
5		52	2014-02-03 10:02:26.778007	3088	5565	VeriSign, Inc.	2010-02-08 00:00:00.0	2020-02-07 23:59:59.0
6		52	2014-02-03 10:02:26.778007	3088	5565	GXS	2013-03-26 00:00:00.0	2016-03-25 23:59:59.0

# Chained Certificates

## Secure Sockets Layer

### TLSv1 Record Layer: Handshake Protocol: Multiple Handshake Messages

Content Type: Handshake (22)

Version: TLS 1.0 (0x0301)

Length: 4196

+ Handshake Protocol: Server Hello

+ Handshake Protocol: Certificate

Handshake Type: Certificate (11)

Length: 4114

Certificates Length: 4111

+ Certificates (4111 bytes)

Certificate Length: 1343 ←

+ Certificate (id-at-commonName=, i

Certificate Length: 1520 ←

+ Certificate (id-at-commonName=VeriSign Class 3 Secure Server CA - G3,id-at-organization

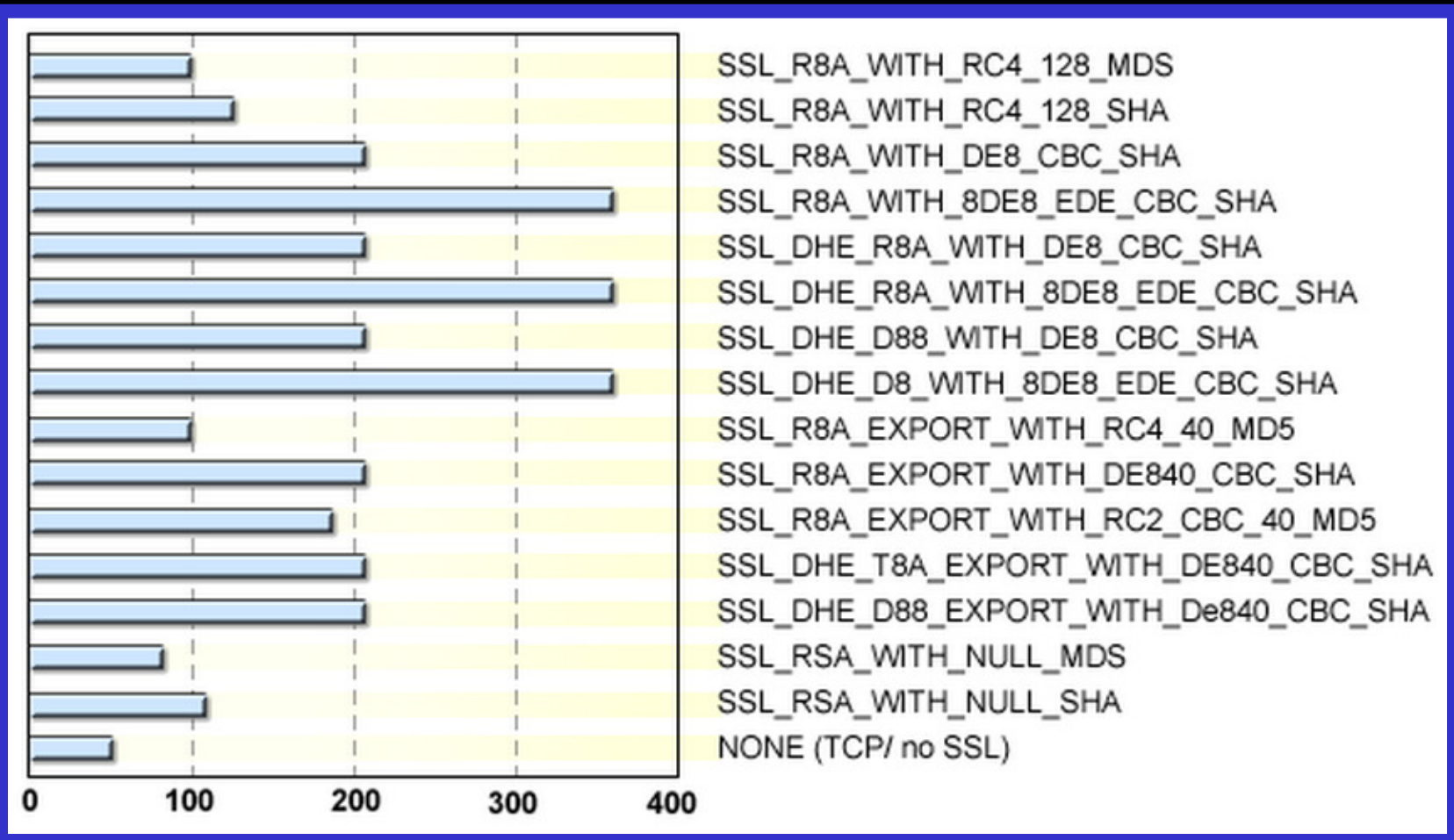
Certificate Length: 1239 ←

+ Certificate (id-at-commonName=VeriSign Class 3 Public Primary Certification ,id-at-orga

+ Handshake Protocol: Server Hello Done



# Relative Speed of Symmetric Algorithms



i5/OS Information Center

<https://publib.boulder.ibm.com/infocenter/iseri5/v5r4/index.jsp?topic=%2Frzamy%2F50%2Fsec%2Fsectussl.htm>



# Summary



- It will only get worse!
- Questions???



41ST INTERNATIONAL  
**IT CAPACITY &  
PERFORMANCE**  
CONFERENCE



NOVEMBER 2-5, 2015  
THE ST. ANTHONY HOTEL  
San Antonio, TX

**Join us in San Antonio for the 2015 CMG Conference!**

**Save the dates:**

**November 2<sup>nd</sup> to 5<sup>th</sup> at The St. Anthony in downtown San Antonio**

**3 blocks to both the Alamo and the Riverwalk**

