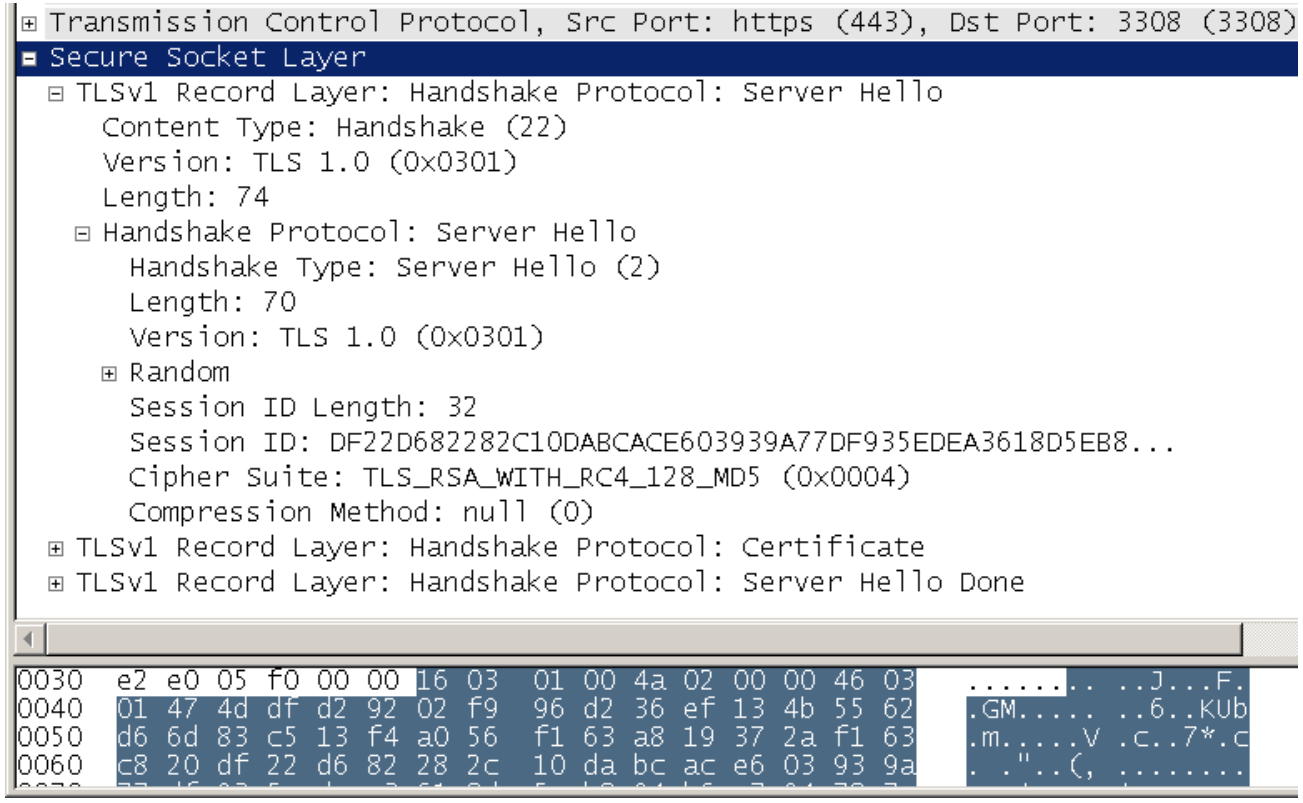
# TLS/SSL MAC security flaw

Nate Lawson
nate@rootlabs.com

Jan. 10, 2008

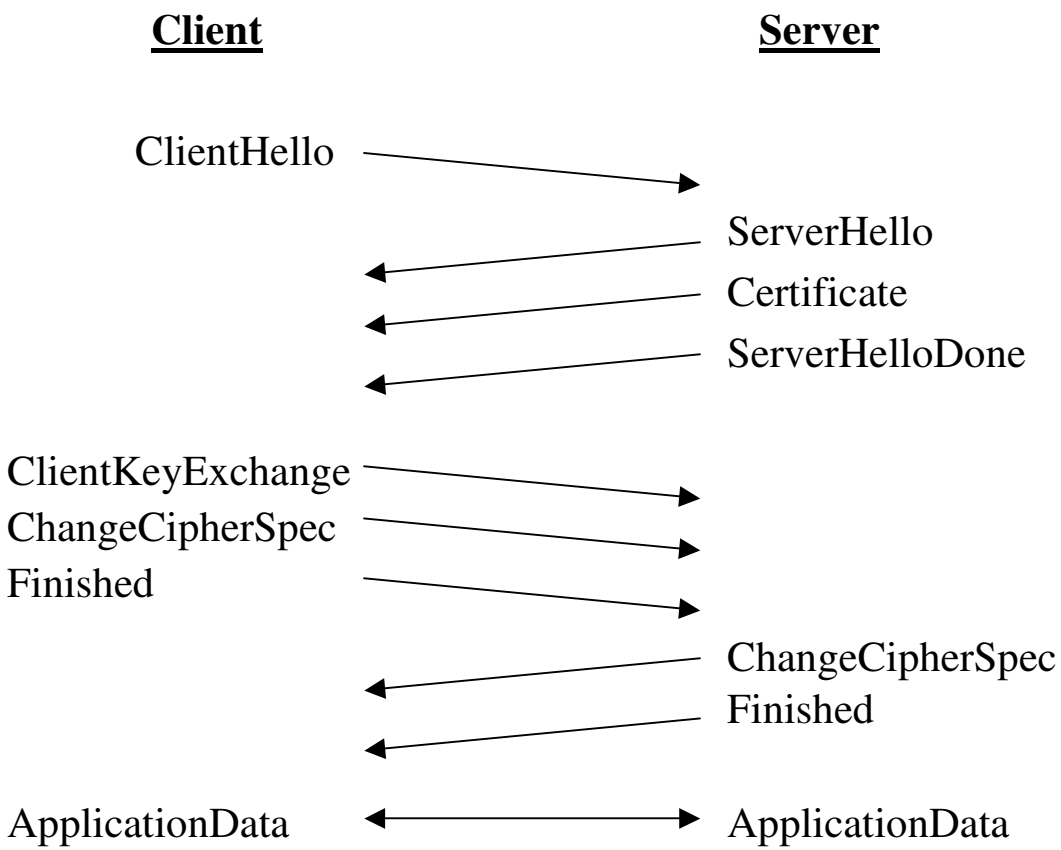
root labs

# Decoding with WireShark




root labs

# Overview of typical session



# TLS 1.1 security fixes

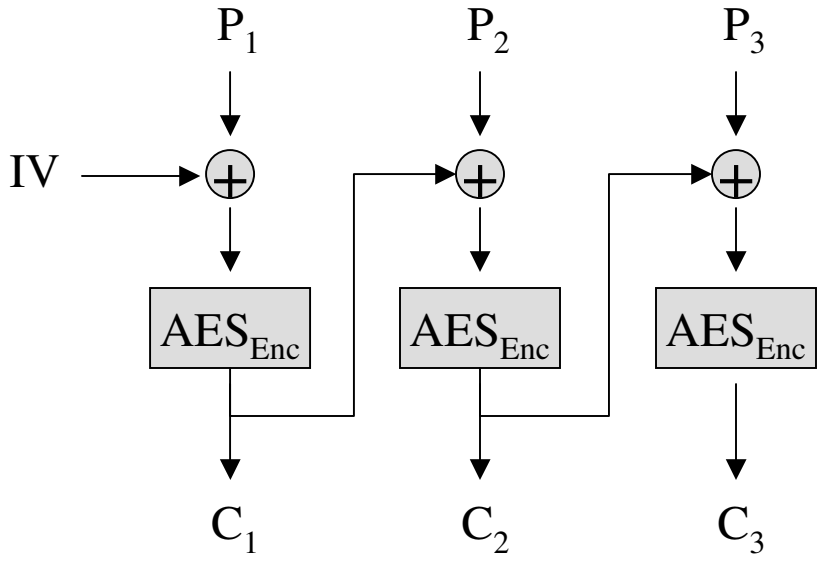
- Two security flaws fixed since TLS 1.0
  - Implicit Initialization Vector (IV) is replaced with an explicit one
  - Handling of padding errors changed to not report decryption_failed

    Credit for both: Bodo Moeller of OpenSSL

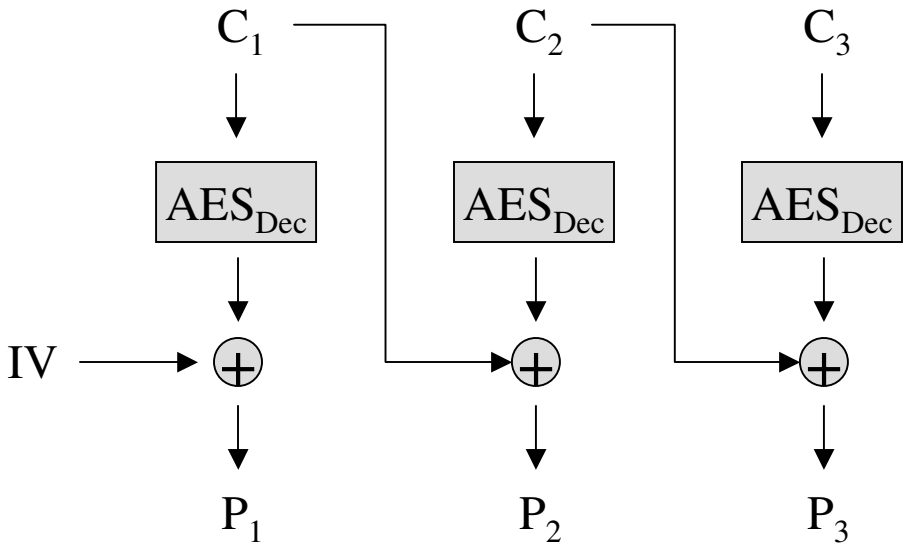
- More details and discussion on my blog:

  http://rootlabs.com (select Blog)

# CBC encryption



# CBC decryption



# TLS CBC padding

- Padding needed if message is not multiple of cipher block size
  - Pad remaining bytes of block with bytes of PaddingLen - 1
  - 3 bytes of padding = 0x2 0x2 0x2
- Example: AES-CBC, 30 bytes data
  - $P_1$: 16 bytes data
  - $P_2$: 14 bytes data || 0x1 0x1

# Padding error timing attack

- Two different errors
  - If padding verification fails, “padding_error”
  - If subsequent integrity check fails, “bad_record_mac”
- Attacker can’t see these (encrypted)
  - But, server may exit out early if padding incorrect and not bother to check MAC
  - Creates an exploitable timing channel

# CBC padding attack

- Allows guessing the last byte of a sniffed encrypted record
- Attack overview
  - Modify and replay entire record
  - Observe how long it takes for error to be returned
  - Repeat until it takes a little longer
    - Padding passed check and thus server proceeded to check the MAC of the data

# Example attack scenario

- Original message 32 bytes data
  - $C_1$: AES(IV ⊕ 16 bytes data)
  - $C_2$: AES($C_1$ ⊕ 16 bytes data)
- Attacker modifies message
  - $C_1$: 15 bytes garbage || (GuessByte ⊕ 0x0)
  - $C_2$: same
  - Truncates external length to 31 bytes
- If guess byte is correct, padding verifies and server proceeds to MAC stage
  - $P_2$: 15 bytes garbage || 0x0
    - GuessByte ⊕ RealByte ⊕ 0x0 = 0
    - PaddingLen = 1 means append one byte of 0x0

# Conclusion

- Detailed error reporting harmful to crypto
  - Surprise! You want nothing more than a big, giant FAIL at the end of your protocol
- Side channels reveal enough for an attack, even when data is encrypted
  - Surprise! Proceed (with caution) even when an error is encountered

# Recommended reading


- [TLS06] The Transport Layer Security (TLS) Protocol, Version 1.1. http://tools.ietf.org/html/rfc4346
- [Resc02] Rescarola, E. Introduction to OpenSSL programming. http://www.rtfm.com/openssl-examples/
- [WS96] David Wagner and Bruce Schneier. Analysis of the SSL 3.0 Protocol. 1996. http://citeseer.ist.psu.edu/wagner96analysis.html
- [BB03] D. Boneh and D. Brumley. Remote Timing Attacks are Practical. Proceedings of the 12th USENIX Security Symposium, August 2003. http://citeseer.ist.psu.edu/article/boneh03remote.html
- [M04] B. Moeller. Security of CBC Ciphersuites in SSL/TLS: Problems and Countermeasures. May 2004. http://www.openssl.org/~bodo/tls-cbc.txt