SSL/TLS技术原理

2016/08

提纲

- 一个简化安全系统的设计

- SSL/TLS
  - SSL/TLS与简化系统的对比
  - TLS 1.0/1.1/1.2
  - SSL的安全要素
  - 著名的SSL漏洞

- SSL实现
  - 开源实现
  - 商用实现

一个简化安全系统的设计

- 常见的实现：加密的ZIP和RAR文件

- PKI的算法基础
  - 对称加密算法
  - 非对称算法
  - 摘要算法

- 层次递进的安全性需求
  - 摘要和加密 -> 需要密钥（对称加密的密钥/IV，HMAC计算的密钥）
  - 密钥传递 -> 密钥协商（RSA数字信封/ECDHE密钥协商）
  - 密钥协商中的公钥保护 -> 签名
  - 签名的验证 -> 数字证书及CA证书链
  - 防止重放攻击：双方的随机数

提纲

- 一个简化安全系统的设计

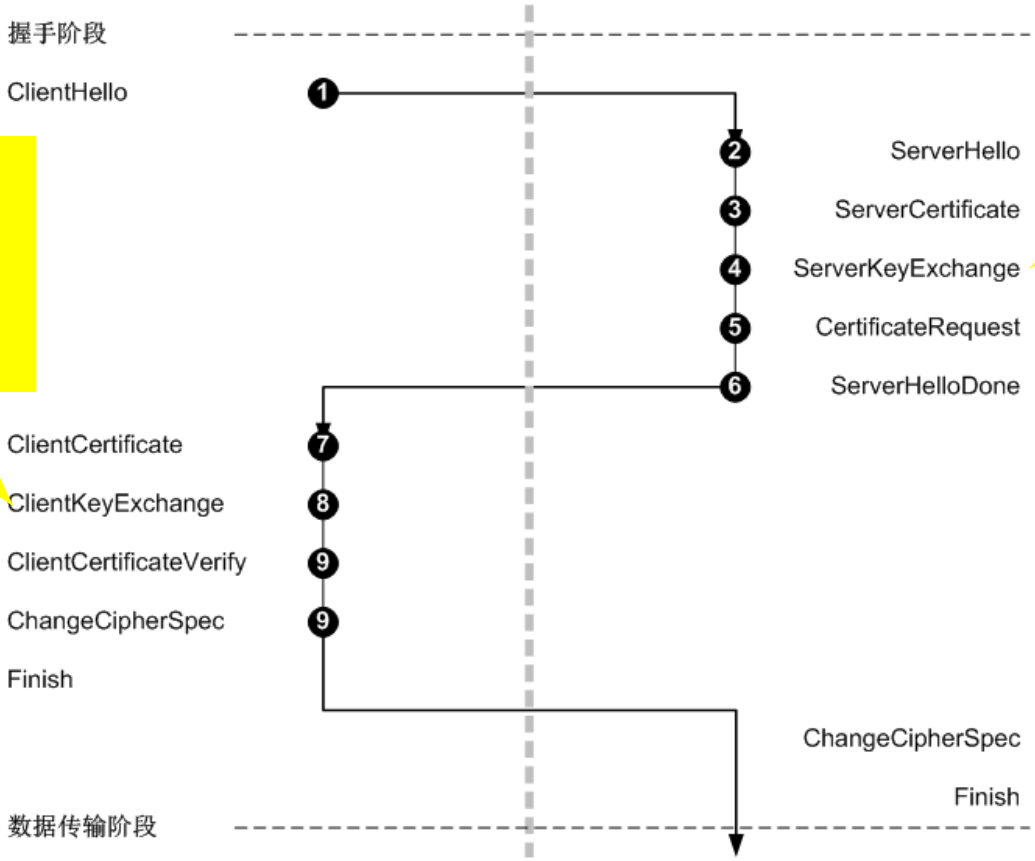
- SSL/TLS
  - SSL/TLS与简化系统的对比
  - TLS 1.0/1.1/1.2
  - SSL的安全要素
  - 著名的SSL漏洞

- SSL实现
  - 开源实现
  - 商用实现

SSL/TLS与简化系统的对比

- 身份认证
  - 签名/验签（RSA，ECDSA）
  - 证书验证（证书链验证，有效期验证，黑名单和OCSP验证）
- 密钥协商
  - RSA算法（公钥加密/私钥解密）
  - ECC算法（ECDH密钥协商）
  - 临时密钥(TEMP RSA/ECDHE)
- 数据加密(两个工作密钥，两个IV)
  - AES/DES/RC4
- 完整性保护（两个HMAC密钥）
  - SHA1/SHA256
- 防止重放攻击
  - Client Random
  - Server Random

SSL协议



RSA：数字信封
ECDHE：密钥交换

此步骤客户端将完成PreMasterSecret的计算，并通过PRF推导出4个工作密钥和两个IV

客户端签名

RSA：可选
ECDHE：必须

TLS 1.1/1.2

- TLS 1.0 - TLS 1.1（显式IV）
  - The Implicit Initialization Vector (IV) is replaced with an explicit IV to protect against Cipher block chaining (CBC) attacks.
  - Handling of padded errors is changed to use the bad_record_mac alert rather than the decryption_failed alert to protect against CBC attacks.

- TLS 1.1 - TLS 1.2（可协商的PRF算法，采用单一HASH）
  - The MD5/SHA-1 combination in the pseudorandom function (PRF) was replaced with cipher-suite-specified PRFs.
  - The MD5/SHA-1 combination in the digitally-signed element was replaced with a single hash. Signed elements include a field explicitly specifying the hash algorithm used.
  - TLS Extensions definition and AES Cipher Suites were merged in.
  - Tighter checking of EncryptedPreMasterSecret version numbers.

SSL的安全要素

- 身份认证
  - Certificate：证书的验证
  - ClientVerify：客户端签名，服务端验签
  - ServerKeyExchange：服务端对密钥交换参数签名，客户端验签

- 密钥交换
  - RSA：客户端在ClientKeyExchange中用服务端公钥加密PreMasterSecret
  - ECDHE：客户端和服务端各自产生一对临时ECC密钥，分别计算PreMasterSecret
    - Client: ECDH_ComputeKey(client_tmp_pri, server_tmp_pub)
    - Server: ECDH_ComputeKey(server_tmp_pri, client_tmp_pub)
  - SM2：客户端和服务端各自产生一对临时ECC密钥，结合自身的加密密钥对分别计算PreMasterSecret
    - Client: SM2_KeyExchange(client_enc_pri, client_tmp_pri server_enc_pub, server_tmp_pub)
    - Server: SM2_KeyExchange(server_enc_pri, server_tmp_pri, client_enc_pub, client_tmp_pub)
  - ECDHE和SM2的密钥交换方式被称为PFW - Perfect Forward Secrecy

SSL的安全要素

- 数据加密
  - 序列加密算法：RC4，ZUC（祖冲之密码算法）
  - 分组加密算法：DES，AES，SM1，SM4
- 数据完整性
  - SHA1_HMAC
  - SHA256_HMAC
  - SM3_HMAC
- PRF密钥推导
- SSL Session

```
Struct {
  ConnectionEnd          entity;
  PRFAlgorithm           prf_algorithm;
  BulkCipherAlgorithm    bulk_cipher_algorithm;
  CipherType             cipher_type;
  uint8                  enc_key_length;
  uint8                  block_length;
  uint8                  fixed_iv_length;
  uint8                  record_iv_length;
  MACAlgorithm           mac_algorithm;
  uint8                  mac_length;
  uint8                  mac_key_length;
  CompressionMethod      compression_algorithm;
  opaque                 master_secret[48];
  opaque                 client_random[32];
  paque                  server_random[32];
}SecurityParameters;
```

已知的SSL漏洞——协议漏洞

- SSL 3.0重协商
  - 协议设计的重大漏洞
  - TLS 1.0引入“安全重协商”解决此漏洞
- BEAST(Browser Exploit Against SSL/TLS)
  - 针对CBC模式算法攻击
  - SSL 3.0/TLS 1.0可以改用序列算法，或者采用OpenSSL的send “empty TLS record”进行一定程度的弥补
  - TLS 1.1引入了Explicit IV解决此问题
- CRIME(Compression Ratio Info-leak Made Easy)
  - 通过禁用压缩算法避免此问题
- POODLE(Padding Oracle On Downgraded Legacy Encryption)
  - 协议设计的漏洞
  - 通过禁用SSL 3.0解决

已知的SSL漏洞——实现漏洞

- Heartbleed
  - OpenSSL 1.0.1g解决此问题
- Early CCS（Early ChangeChipherSpec）
  - OpenSSL 1.0.1h解决此问题

提纲

- 一个简化安全系统的设计

- SSL/TLS
  - SSL/TLS与简化系统的对比
  - TLS 1.0/1.1/1.2
  - SSL的安全要素
  - 著名的SSL漏洞

- SSL实现
  - 开源实现
  - 商用实现

SSL实现/商用实现

- Microsoft - Schannel
  - IE浏览器
  - IIS服务器
  - 其他基于Windows SDK编程的应用程序
- Apple - Secure Transport
  - OS X
  - IOS
- Sun/Oracle - Java JSSE

SSL实现/开源实现

- OpenSSL
  - Linux: Apache/Nginx/…
  - OpenJDK
  - Android SSL
  - LibreSSL/BoringSSL
- NSS(Network Security Services)
  - Netscape
  - Mozilla Firefox
- BouncyCastle
  - Java
  - C#
- PolarSSL/GnuTLS
- Go Lang

实践和练习

- 资料阅读
  - http://git.koal.com/training/gw-training/wikis/openssl
- 抓包分析
  - 使用Wireshark分析IE/Chrome/Firefox访问公司邮件系统的SSL过程
  - 使用Wireshark的深度分析技术，获取SSL中的明文
- 编程练习（Java，WinHttp，CURL+OpenSSL）
  - 获得公司邮件系统的邮件列表页面

谢谢各位

2016-08-12