CS356: TLS

W. Michael Petullo

University of Wisconsin–La Crosse

As of November 1, 2021

Cryptographic Primitives: Symmetric-key Cryptography

- Only k and t must remain secret (Kerckhoff 1883)
- Allows cryptographers to rigorously examine cipher

```
[user@host]$ gpg --cipher-algo
    AES256 --symmetric FILE
[user@host]$ gpg --output FILE
    --decrypt FILE.gpg
```

- ~~3DES~~
- ~~RC4~~
- AES
- Salsa

Cryptographic Primitives: Asymmetric-key Cryptography

- k_p and k_s are related; difficult to derive k_s from k_p
- Receiver makes k_p public; sender uses k_p to encrypt
- Receiver uses k_s to decrypt

- RSA
- Curve25519

```
[user@host]$ gpg --full-generate-key
[user@host]$ gpg --output mike.gpg --armor --export mike@flyn.org
```

```
[user@host]$ gpg --import <mike.gpg
[user@host]$ gpg --encrypt --recipient mike@flyn.org FILE
```

```
[user@host]$ gpg --output FILE --decrypt FILE.gpg
```

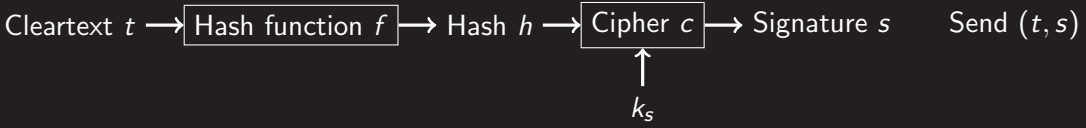
Cryptographic Primitives: Cryptographic Hash

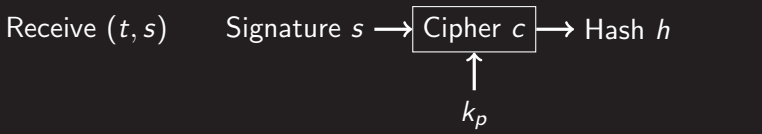
1. Given h, cannot determine t
2. Given t and f, cannot find t': $f(t) = f(t') = h$
3. Cannot find some u and u' where $f(u) = f(u')$

```
[user@host]$ sha256sum FILE
```

- ~~MD5~~
- ~~SHA-1~~
- SHA-256
- SHA-512

Cryptographic Primitives: Digital Signatures






```
[user@host]$ gpg --sign FILE
```

```
[user@host]$ gpg --output FILE --decrypt FILE.gpg
```

- RSA
- Ed25519

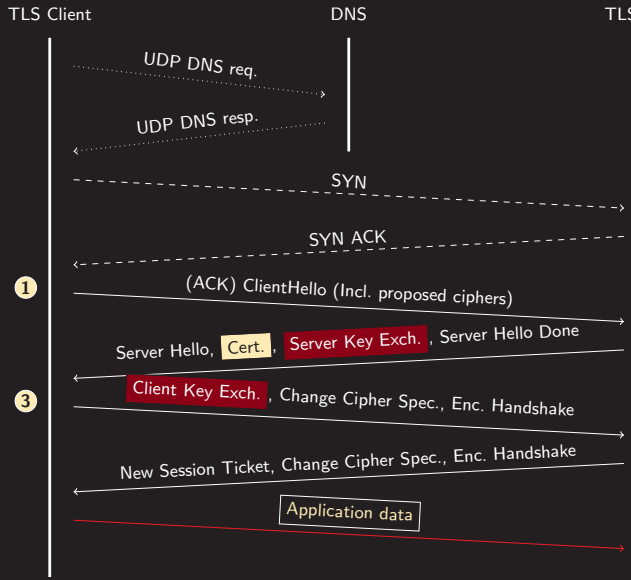
TLS: Certificate

```
-----BEGIN PRIVATE KEY-----
[REDACTED]
-----END PRIVATE KEY-----
-----BEGIN CERTIFICATE-----
MIIGIjCCBQqgAwIBAgISBEt/lwvghbfYmo0BK8+p5suXMA0GCSqGSIb3DQEBCwUA
MDIxCzAJBgNVBAYTAlVTMRYwFAYDVQQKEw1MZXQncyBFbmNyeXB0MQswCQYDVQQD
...
A59PTWXgeGmr875FbwPAFpsT2y2P4nUqPbI9lijdr1KkzRoqPeORINhdsIsJjjkF
hctHK0Cvf0/IG81aWLmohOLDjXwIkHxZ72viHo0FazK5OIWje74=
-----END CERTIFICATE-----
```

TLS: Certificate

```
[user@host]$ openssl x509 -in flyn.org.pem -noout -text
Certificate:
  Data:
    ...
    Issuer: C=US, O=Let's Encrypt, CN=R3
    ...
    Subject: CN=flyn.org
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
        RSA Public-Key: (4096 bit)
        Modulus:
          00:ac:d5:5a:f1:2c:b6:6a:85:c4:30:0b:56:cc:73:
          ...
        Exponent: 65537 (0x10001)
  Signature Algorithm: sha256WithRSAEncryption
    45:57:9f:34:75:38:d5:34:54:43:46:83:c9:50:d4:9c:3d:fa:
    ...
```

Cryptographic Systems: TLS with www.lwn.net



Cipher Suite

ECDHE	Key exchange
RSA	Signature scheme
AES	Symmetric encryption
SHA384	Hashing

TLS: Attack

A cryptologic attack on TLS is outside the scope of this course.

. . . but, perhaps we could cause a client to skip TLS.

TLS: Attack

1. Assume the identity of TLS server (e.g., use `arpspoof`).
2. Capture the HTTP GET from a client.
3. Forward GET to TLS server over HTTPS.
4. Act as a proxy between HTTP and HTTPS.

TLS: BSD Socket Service

See sample BSD Socket Service in tlsproxy project.

TLS: mbed TLS

See mbed TLS sample in repository.

TLS: Assignments

Graded Homework Aquinas: tlsproxy

Reading Read 0x700–0x753

https://www.flyn.org/courses/cs356/schedule