

HTTPS and Secure Communication

Jogesh K. Muppala



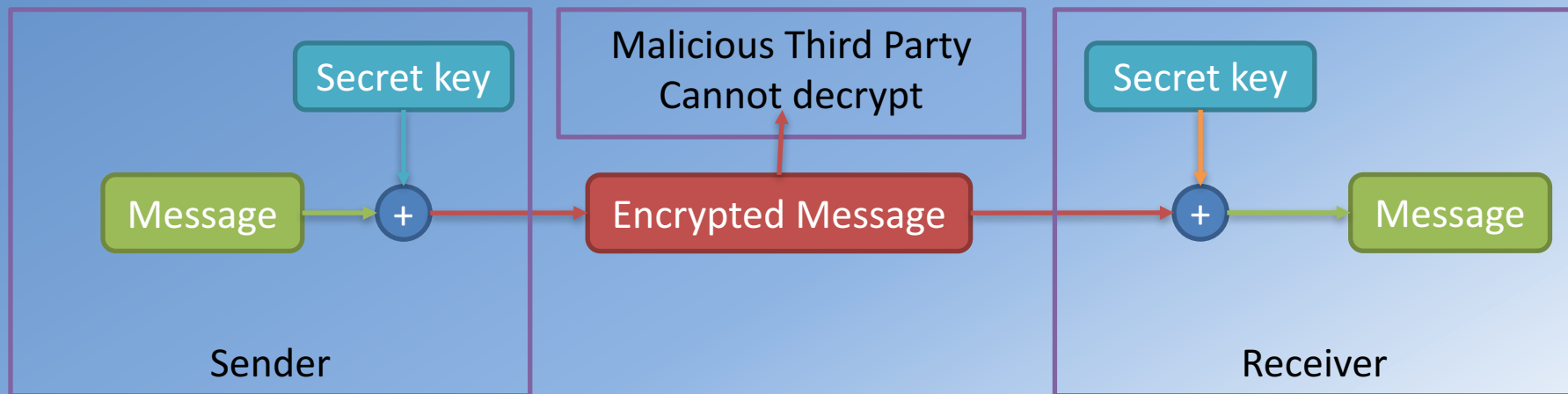
THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系



香港科技大學
THE HONG KONG UNIVERSITY OF
SCIENCE AND TECHNOLOGY

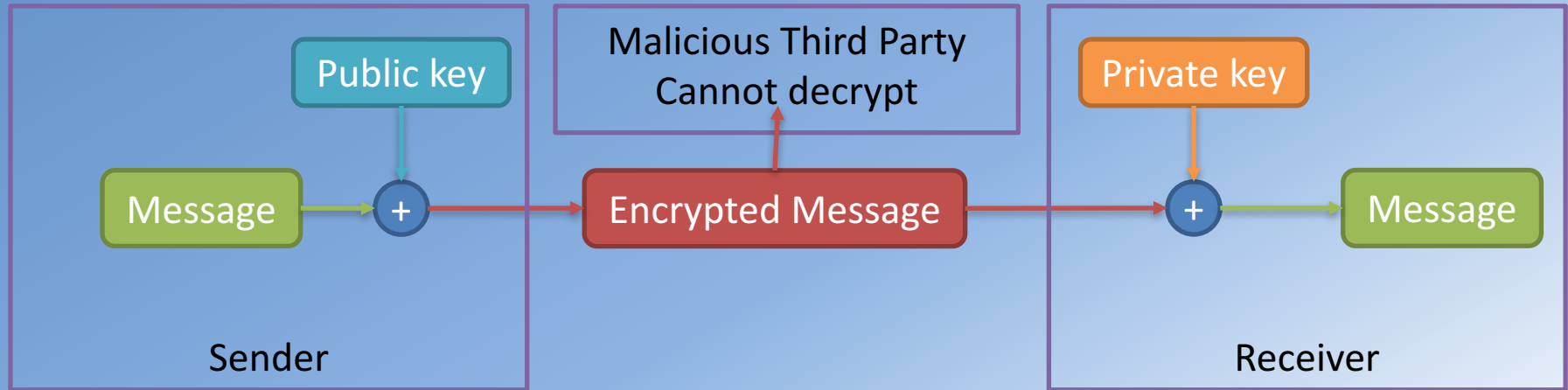
Symmetric Key Cryptography

- Symmetric Encryption:
 - Shared secret key between the two parties



Public Key Cryptography

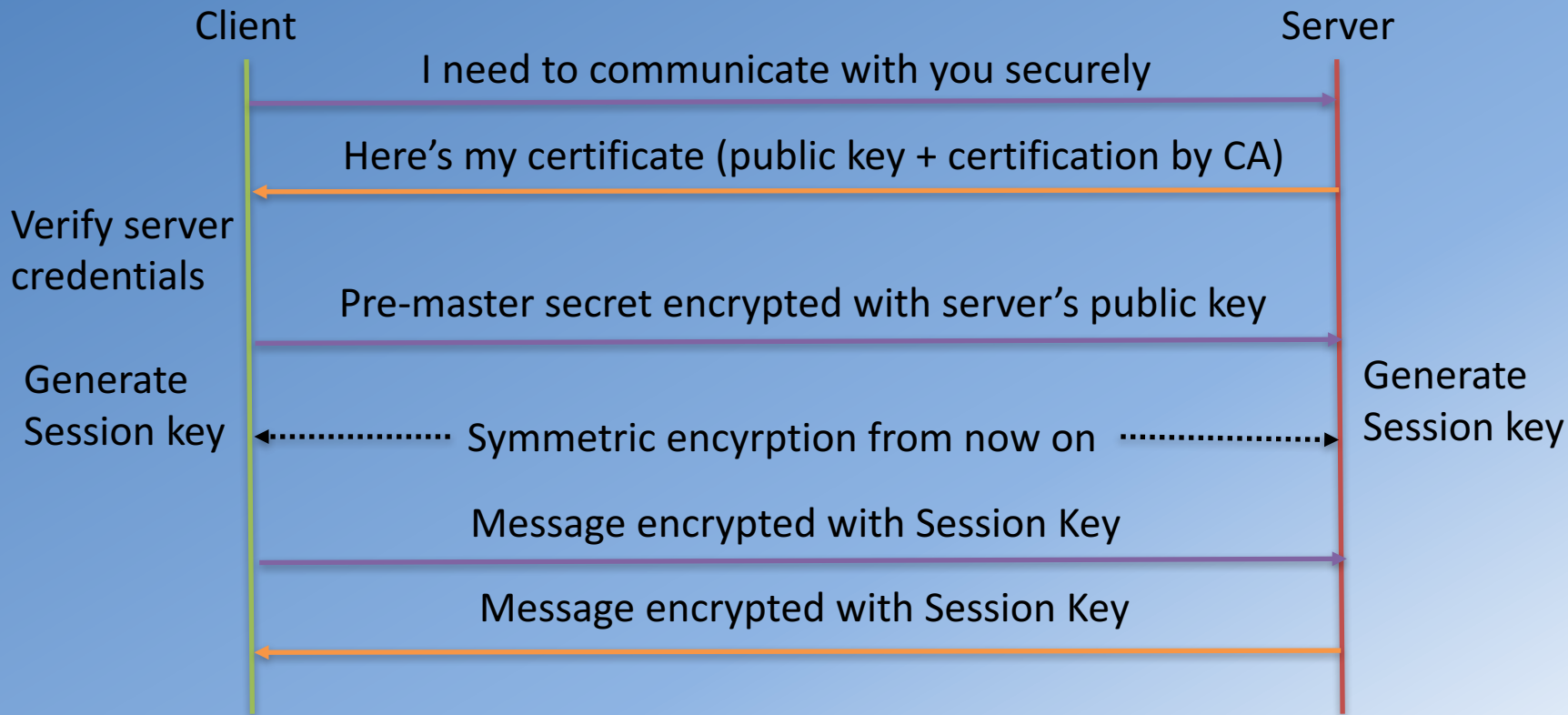
- Asymmetric Encryption:
 - Public key that can be widely distributed
 - Private key that is only known to the receiver



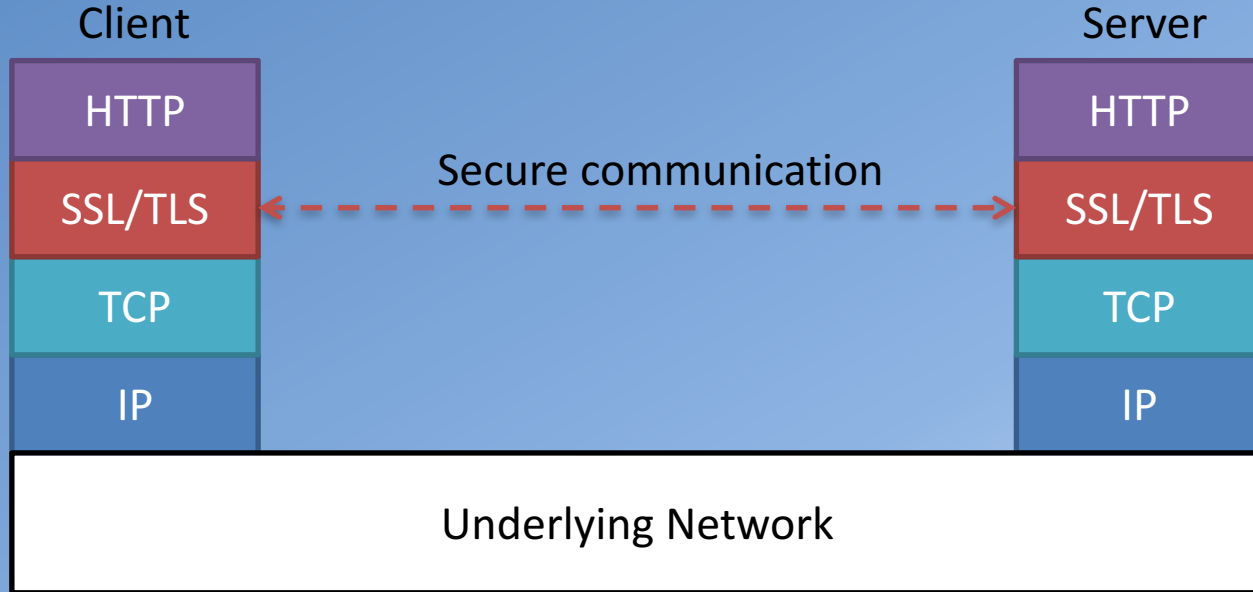
Secure Sockets Layer (SSL) / Transport Layer Security (TLS)

- Cryptographic protocols that enable secure communication over an insecure network like the Internet
- Privacy and Integrity of the communication protected
 - Uses a combination of public-key cryptography and symmetric cryptography

SSL/TLS Handshake



HTTPS



Generating Keys

- Use openssl for generating keys for testing
 - openssl genrsa 1024 > private.key
 - openssl req -new -key private.key -out cert.csr
 - openssl x509 -req -in cert.csr -signkey private.key -out certificate.pem
- For production environment / deploying to a production server you need to get the keys and certificate from a certification authority (CA) e.g., Verisign, Thawte

Node HTTPS Module

- HTTPS core module in Node:

```
var https = require('https');
```

```
var fs = require('fs');
```

```
var options = {
```

```
  key: fs.readFileSync(__dirname+'/private.key'),
```

```
  cert: fs.readFileSync(__dirname+'/certificate.pem')
```

```
};
```

```
var secureServer = https.createServer(options,app);
```