Introduction to practical cryptography

Ivan Vankov (gatakka)

Principal blockchain architect at CognitionFoundry

IBM champion

What is the definition of SECURE?

Lets encrypt the message “Free me”

Lets encrypt the message “Free me”

01000110011100100110010101100101001000000110110101100101

Lets encrypt the message “Free me”

01000110011100100110010101100101001000000110110101100101

01110100010101101100111111110011110100011010011001010101

Lets encrypt the message “Free me”

01000110011100100110010101100101001000000110110101100101

⊕

01110100010101101100111111110011110100011010011001010101

00110010000100100101010101001011011110001110010110011000

Lets decrypt the message

00110010000100100101010101001011011110001110010110011000 0

⊕

01110100010101101100111111110011110100011010011001010101

Problems with this encryption

- Key size

Problems with this encryption

- Key size
- Key reuse

Problems with this encryption

- Key size
- Key reuse
- Message authentication

Problems with this encryption

- Key size
- Key reuse
- Message authentication
- Digital signature

Problems with this encryption

- Key size
- Key reuse
- Message authentication
- Digital signature
- Key exchange

Symmetric cryptography use the same key for encryption and decryption

Asymmetric criptography, or public key cryptography, use separate keys for encryption and decryption

AES – Advanced Encryption Standard

- Symmetric encryption
- One of the most tested and used
- CPU accelerated
- Is a standard, not implementation
- Very flexible

AES is designed in a such a way, so same key can be used multiple times!

AES – Advanced Encryption Standard

- AES-128
- AES-192
- AES-256

AES cipher modes

- ECB
- CBC
- PCBC
- CFB
- OFB
- CTR
- GCM*

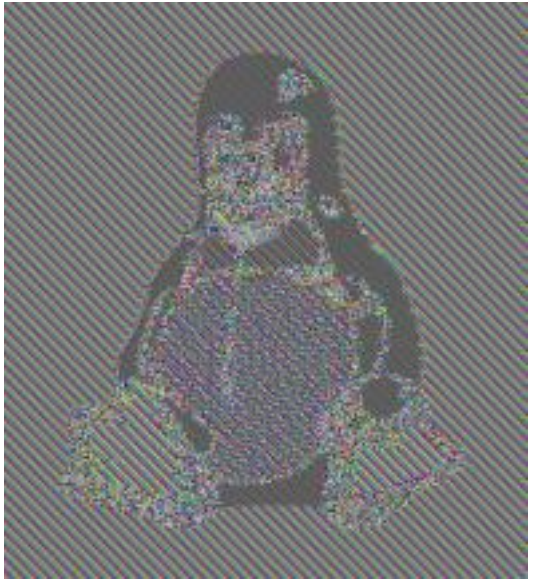
AES ECB split the message in equal size blocks of 16 bytes and encrypts every one of them using the key

ECB encrypts data, but not the patterns!

NEVER use ECB mode!!!

AES ECB

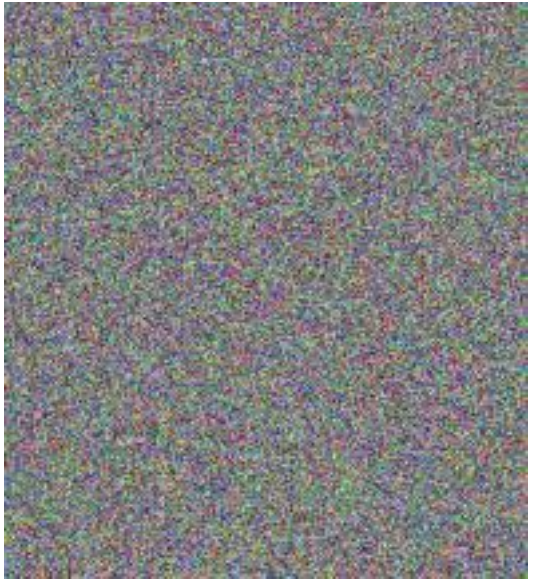
AES ECB



AES CBC

- Splits data into blocks, same as ECB
- Encrypt first block
- XOR second block with first one
- Encrypts second bloc
- XOR third block with second one

AES CBC



AES CBC problems

- Same message with same key will give same result
- First block is not XOR-ed, it will always be the same for same message with same key

AES CBC IV

- IV (initialization vector) is random bytes
- It is used to XOR the first block
- This will change all blocks

AES CBC IV

- IV may not be secure (encrypted)
- IV MUST be random and unpredictable
- IV must be different for every encryption operation
- IV recommended size is 16 bytes
- IV is prepended to encrypted message

AES CBC padding

- Message length in bytes MUST be multiple of 16
- If not, add padding to the message, and then encrypt

MAC - Message authentication code

AES-GCM

- Works in a similar way as CBC
- Encrypt and append MAC to the message
- Take care of the padding
- Make verification of the MAC before decryption
- Decrypt the message and fix the padding

Random numbers

- Do not use PRNG, they are predictable
- Use cryptographic random number generator

Public key cryptography

- RSA and Elliptic curves
- They use 2 keys – private key and public key
- Keep private key secret and protected
- Public key can be shared
- Public key can be efficiently generated from private key
- It is hard to find private key from public key
- Same private key always generate same public key

Public key cryptography

- Can be used for data encryption/decryption*
- Can be used for digital signatures
- Can be used for key exchange

Action	Private key	Public key
Encrypt*		Yes
Decrypt*	Yes	
Sign	Yes	
Verify signature		Yes
Key exchange	Yes	Yes

RSA key size

- RSA1024 is considered weak
- RSA2048 is recommended
- RSA3072 if you have CPU power
- >=RSA3072 use Elliptic curves

Elliptic curves key size

- Key size is fixed by the chosen curve
- X25519 – ~255 bits
- CurveP256 – ~256 bits
- Secp256k1 – ~256 bits *(not supported in all platforms)*

Key size comparison

AES	RSA	Elliptic curves
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

Key generation

1. Generate random number, this is the private key.
2. Apply specific mathematical operation, depending on the algorithm, – this is the public key

Key generation

- In RSA not all numbers can be used as private keys, this is why key size is so big
- In Elliptic curves any random number is a valid private key

Digital signatures

- Provide message authentication, similar to MAC
- Signature can be verified using public key
- Because public key is connected to one private key, if verification is successful, it is highly likely that the message was signed by the owner of the private key
- Digital signature size is dependent on key size

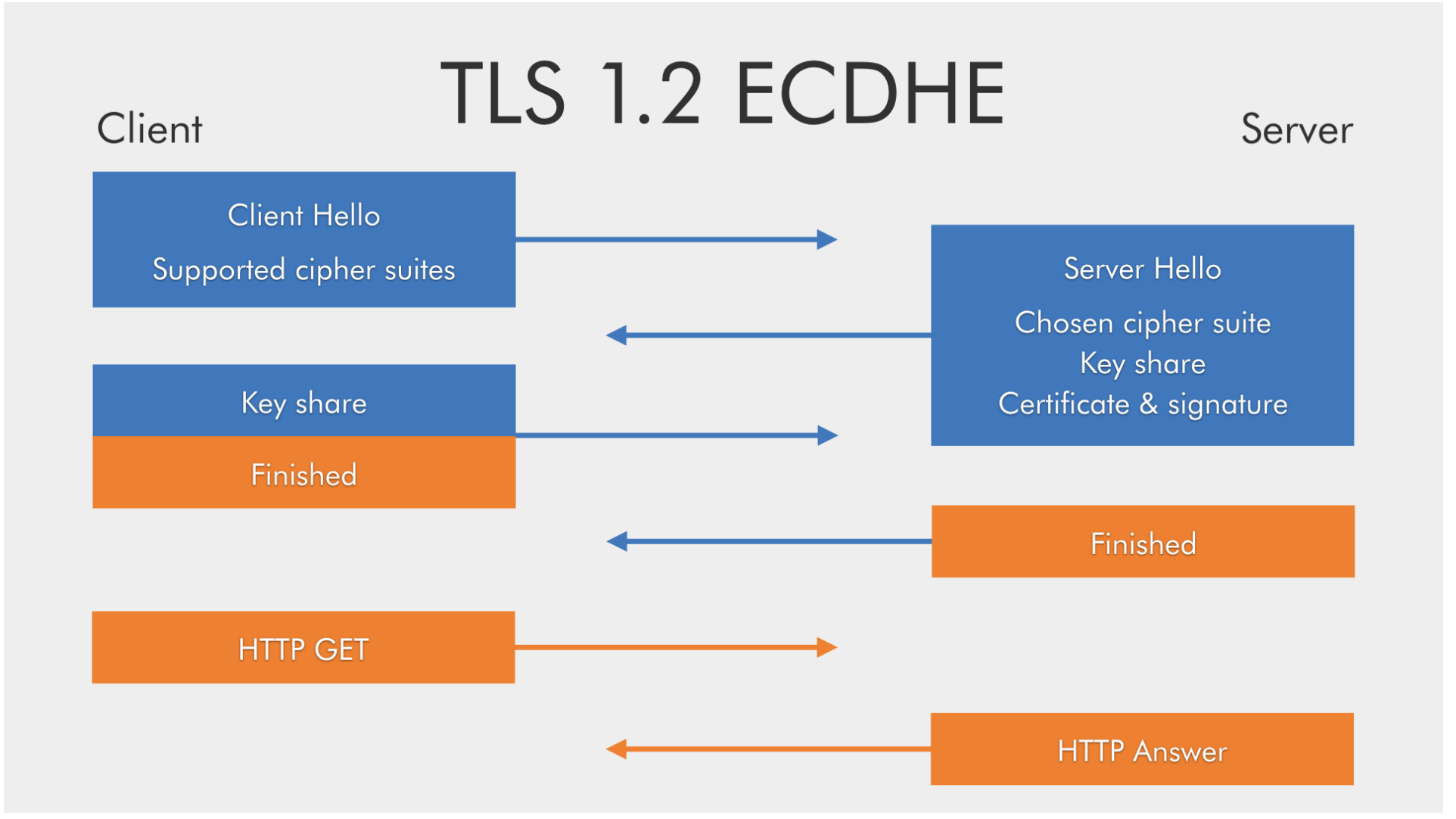
Public key cryptography is **NOT** practical for encryption and decryption!

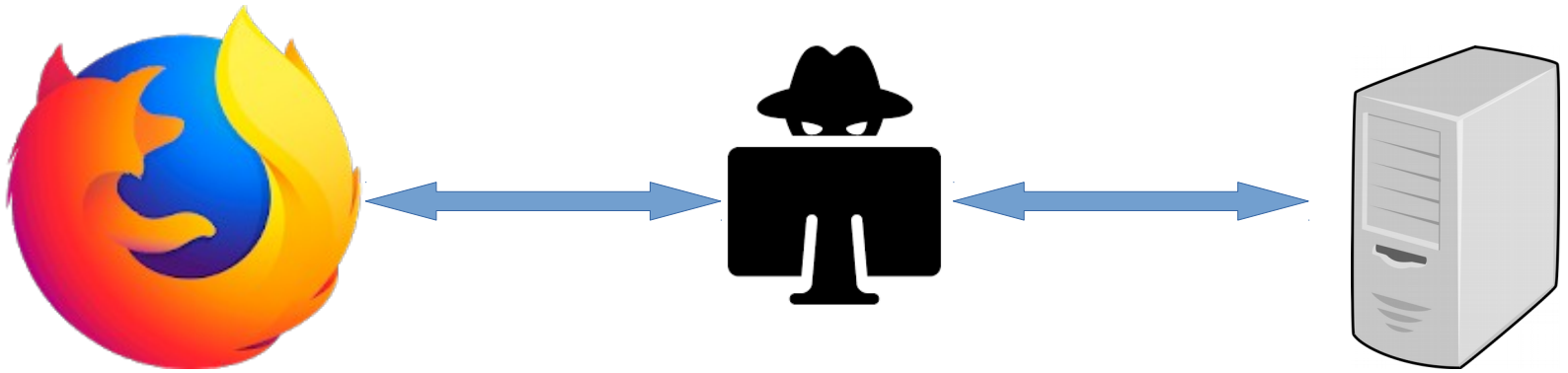
Diffie–Hellman key exchange

- Allows secure key exchange between two or more parties in secure way
- Is based on public key cryptography
- Math is different for RSA and Elliptic curves

ECDH

1. Alice gets Bob public key
2. Alice multiply her private key with Bobs public key
3. Bob gets Alice public key
4. Bob multiply his private key with Alice public key
5. Because A*B=B*A both parties have same result
6. They start to use this result as private key for symmetric cryptography


TLS 1.2 ECDHE
Client
Server
Client Hello
Supported cipher suites
Server Hello
Chosen cipher suite
Key share
Certificate & signature
Key share
Finished
Finished
HTTP GET
HTTP Answer



X.509

- Digital certificate that holds additional information like name, organization, location
- Holds public key of the owner
- May have additional attributes

Self signed certificate

- Useless for security
- Useful for development and testing
- Every one can create it with any data and parameters

Signing a certificate

1. Create public and private key
2. Create Certificate signing request (CSR)
3. Send the CSR to Certificate Authority
4. CA will validate that content of the CSR is valid
5. If is valid, CA will issue a certificate with provided parameters in CSR that is includes CA digital signature

Issued To

Common Name (CN)	*.wikipedia.org
Organization (O)	Wikimedia Foundation, Inc.
Organizational Unit (OU)	<Not Part Of Certificate>
Serial Number	08:30:94:62:D1:FE:A6:0A:E0:BA:BF:F5:EF:8B:C5:45

Issued By

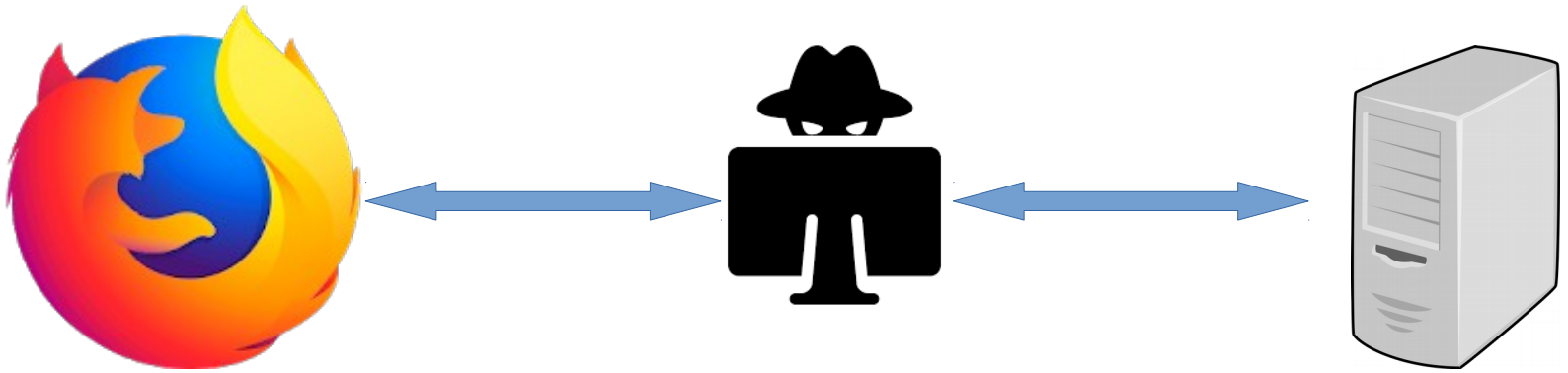
Common Name (CN)	DigiCert SHA2 High Assurance Server CA
Organization (O)	DigiCert Inc
Organizational Unit (OU)	www.digicert.com

Period of Validity

Begins On	December 21, 2017
Expires On	January 24, 2019

Fingerprints

SHA-256 Fingerprint	68:55:49:46:13:AC:3A:18:6E:8A:16:5C:BD:79:12:B7: F1:99:BC:8E:25:F6:1B:60:78:71:B0:8B:06:EC:A6:C9
SHA1 Fingerprint	0F:FB:95:52:F3:B1:3E:CF:AB:6E:82:8C:60:88:A2:0F:D0:04:4E:4E



Thank you