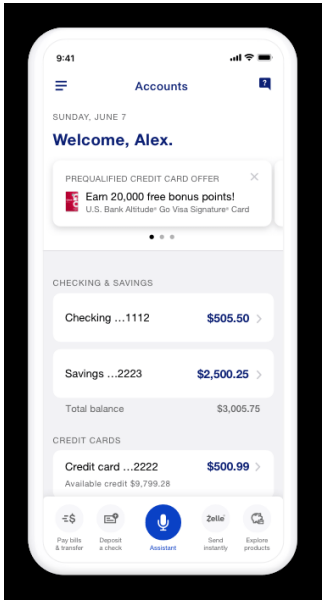
Security in Multi-tier Systems (part 01)

- **Trust**
- **HTTPS handshake**
- **Certificates**
- **Encryption**

- [CS 308 / 350 / 354]

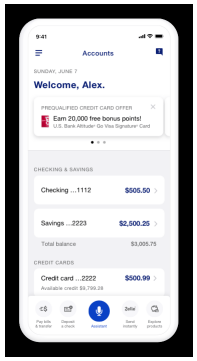
Trust

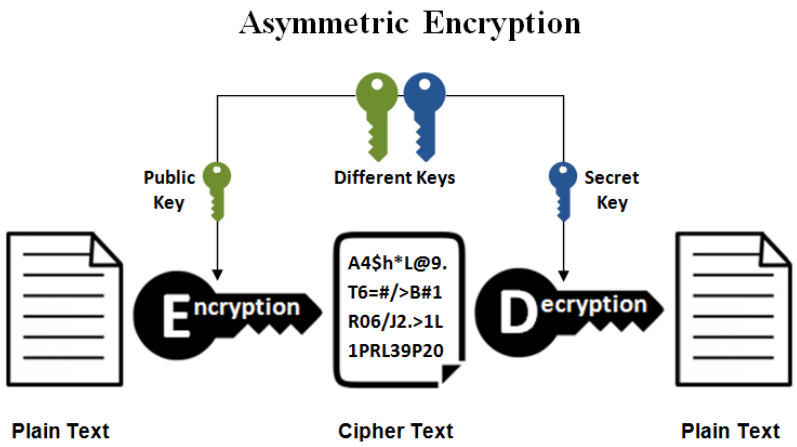
- **How do you know you are communicating with the right server? And how do we know the communication is private and secure?**



Public/private key encryption

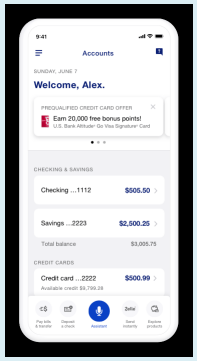
- Encryption is performed using a pair of keys
- If we have US Bank's public key, we can encrypt and send them a message --- and only they can decrypt it because only they have the matching private key

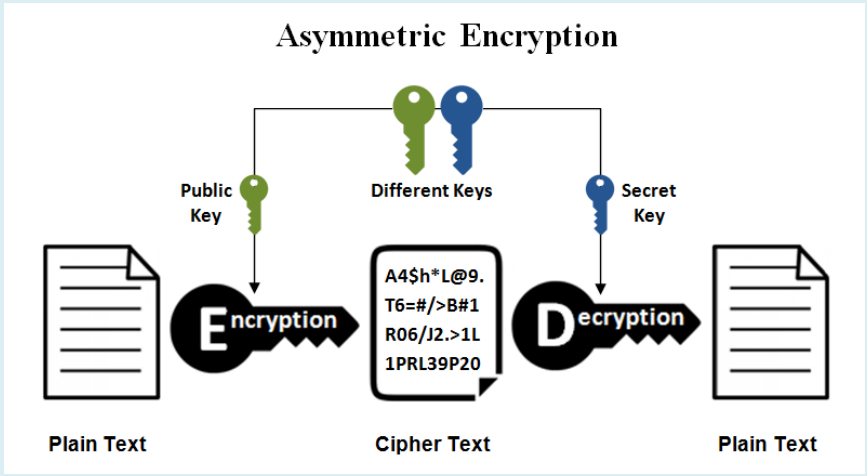




Proof of identity

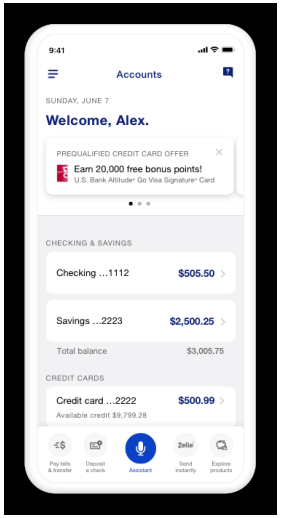
- Also --- if we have US Bank's public key and successfully decrypt their message, we know it's from US Bank because only they have the private key



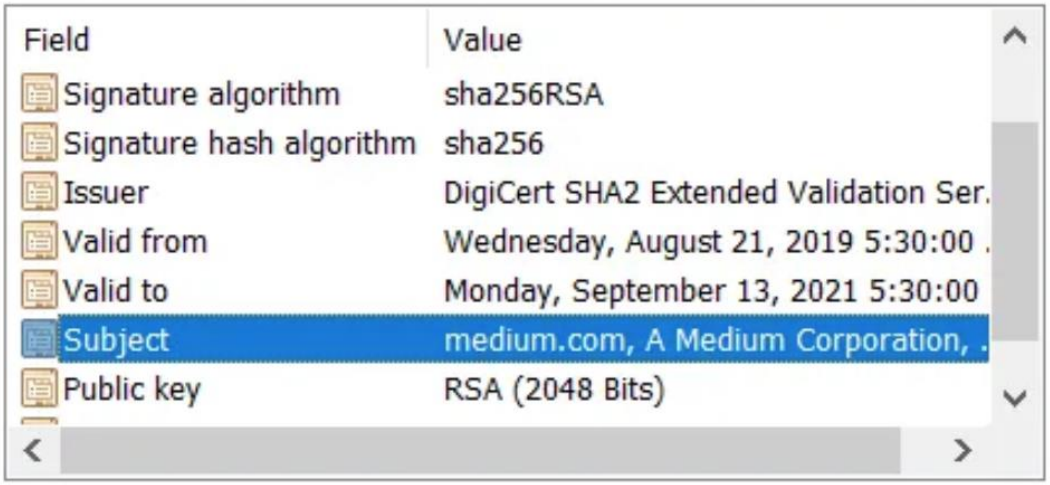


HTTPS "handshake"

- HTTPS is designed to prove identity *and* encrypt by securely getting public key to the client



Example: SSL / TLS certificate



Field	Value
Signature algorithm	sha256RSA
Signature hash algorithm	sha256
Issuer	DigiCert SHA2 Extended Validation Ser.
Valid from	Wednesday, August 21, 2019 5:30:00 .
Valid to	Monday, September 13, 2021 5:30:00
Subject	medium.com, A Medium Corporation, .
Public key	RSA (2048 Bits)

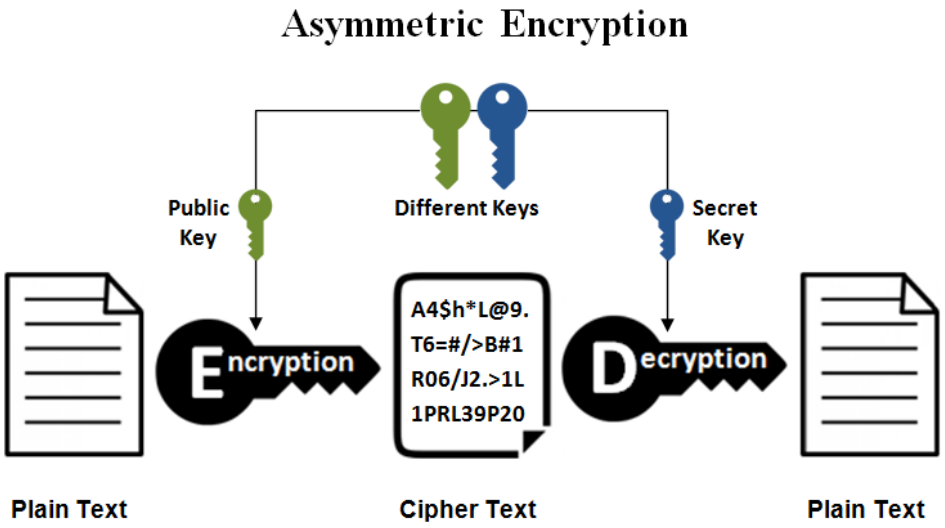
```
CN = medium.com
O = A Medium Corporation
L = San Francisco
S = California
C = US
SERIALNUMBER = 5010624
1.3.6.1.4.1.311.60.2.1.2 = Delaware
1.3.6.1.4.1.311.60.2.1.3 = US
2.5.4.15 = Private Organization
```

Figure 3: https://www.medium.com TLS Certificate Contents at the Time of Writing

https://medium.com/demystifying-security/understanding-tls-certificates-76bdd5815d95

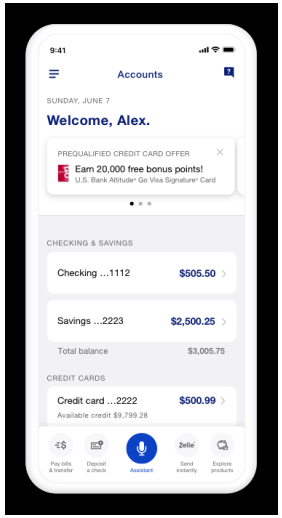
Asymmetric encryption

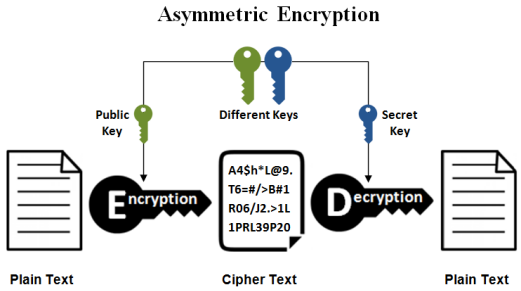
- **Public/private key encryption is an example of asymmetric encryption**
  - *Different keys to encrypt vs. decrypt*
- **Advantage?**
  - *The private key is never shared / transmitted*



Bi-directional

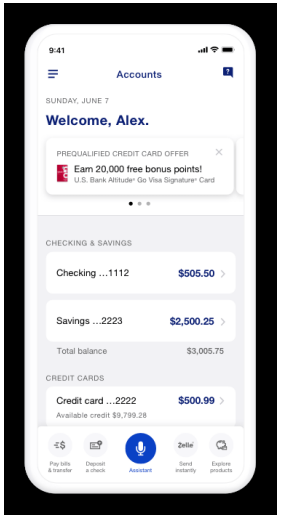
- **Both parties can encrypt and decrypt**
  - *Client encrypts with public, server decrypts with private*
  - *Server encrypts with private, client decrypts with public*





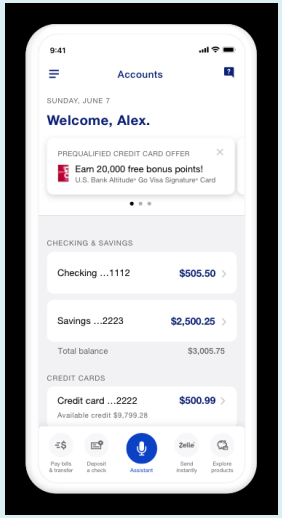
Beware!

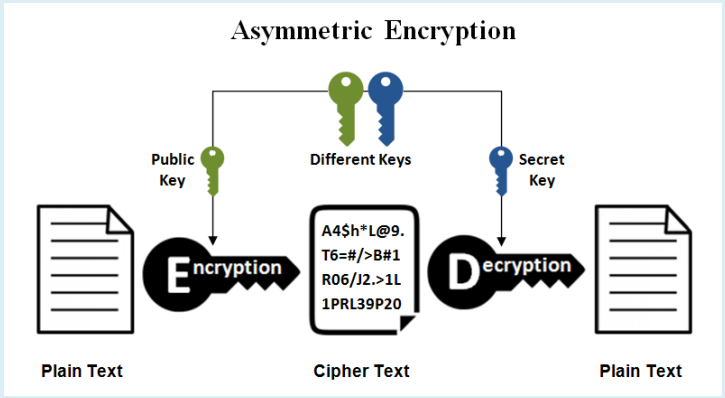
- **It's bi-directional, but…**
  - *Encryption only guarantees privacy in one direction*
  - *Trust is only established in one direction*



Example

- Client has US Bank's public key
- Server (US Bank) has its private key



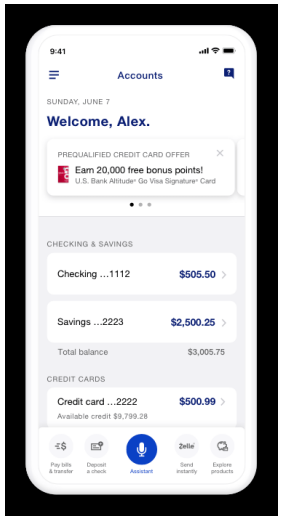


Full security in both directions?

- **Two options:**

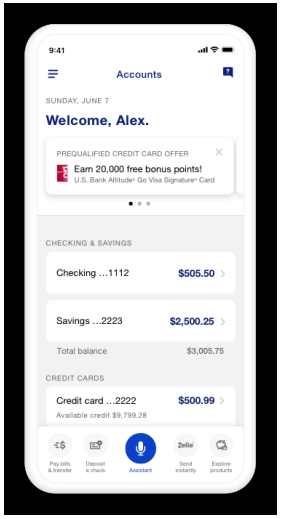
  (1) You need 2 pairs of public/private keys, with each party having public key of the other

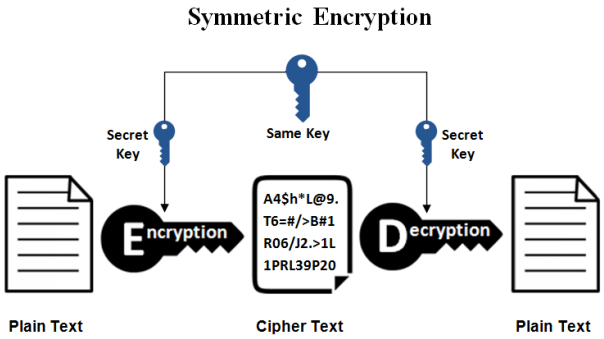
  (2) Use a symmetric key (next slide)



Symmetric encryption

- **After HTTPS handshake, client and server exchange a single key for faster symmetric encryption**
  - *Single key can be used by both parties to encrypt & decrypt*





SSL / TLS certificates

- **The good news?**
- **Easy to obtain, sign, and register certificate so you can secure your app**

https://docs.aws.amazon.com/acm/latest/userguide/gs-acm-request-public.html

https://www.verisign.com/en_US/website-presence/online/ssl-certificates/index.xhtml

SSL = secure sockets layer (deprecated)
TLS = transport layer security (new std)

That's it, thank you!