# PKI, Encryption, Certificates and You

Nathan Toups
@rojoroboto

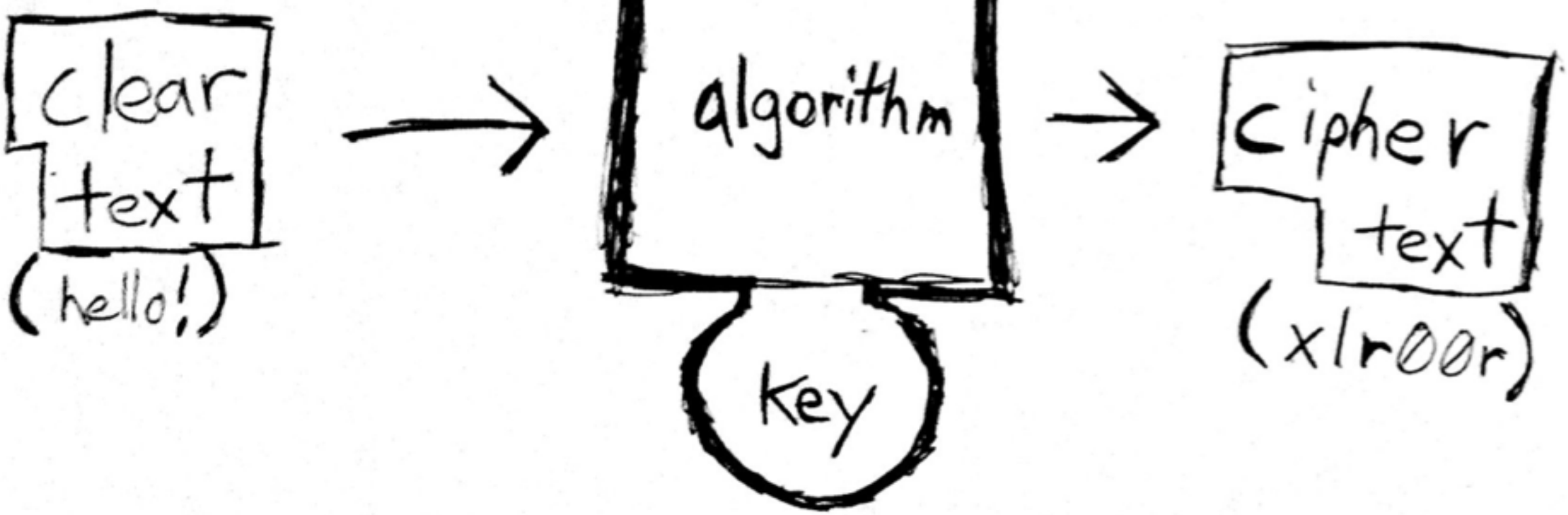
MACTECH

Encryption works. Properly implemented strong crypto systems are one of the few things that you can rely on.

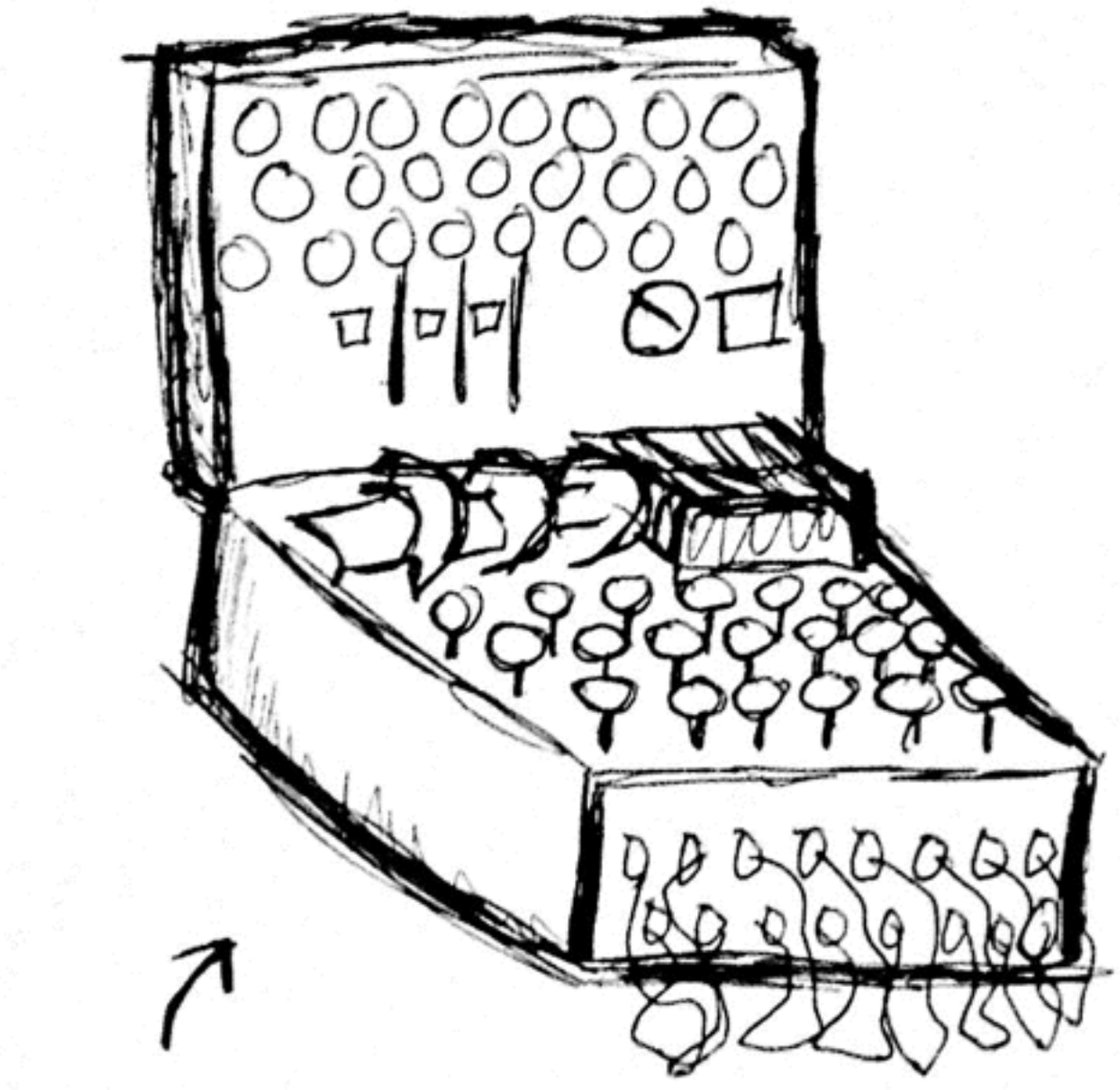
- Edward Snowden

# Fundamentals of encryption

MACTECH


clear
text
(hello!)
algorithm
key
cipher
text
(xlr00r)

# history of cryptography



enigma machine

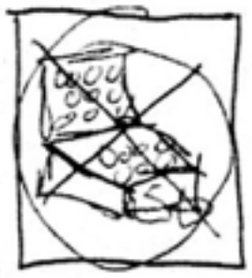
MACTECH

# history of Cryptography

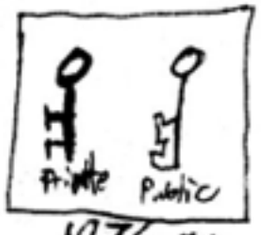
1900 BCE
First evidence of crypto


800 CE
Manuscript for Deciphering Cryptogric Messages

1840 ce
Edgar Allan Poe solves Ciphers



1932 ce
Enigma Crypto deciphered

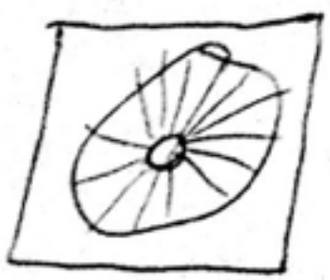



1976 CE
Pub-key introduced

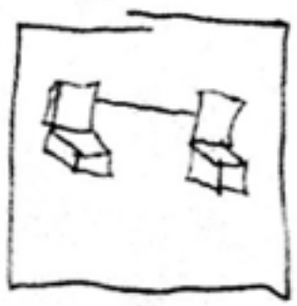
AES

2001 CE
AES Introduced

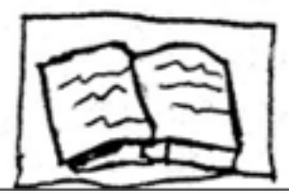
50 BCE
Caesar Cipher



1917 ce
Electro Mechanical Cipher

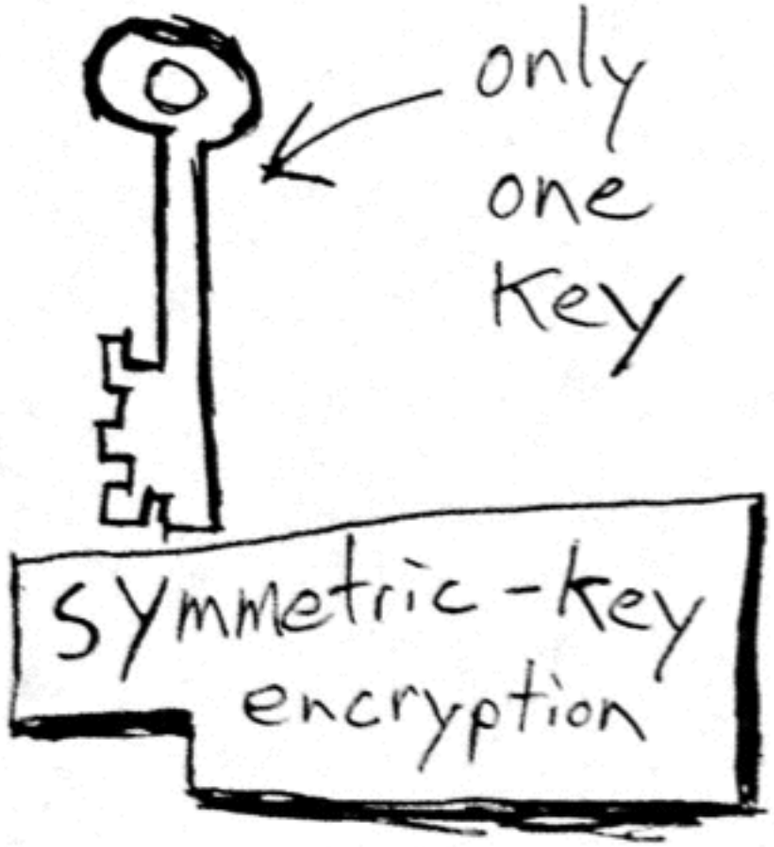


1945 ce
"A mathematical theory of cryptography" Claude Shannon

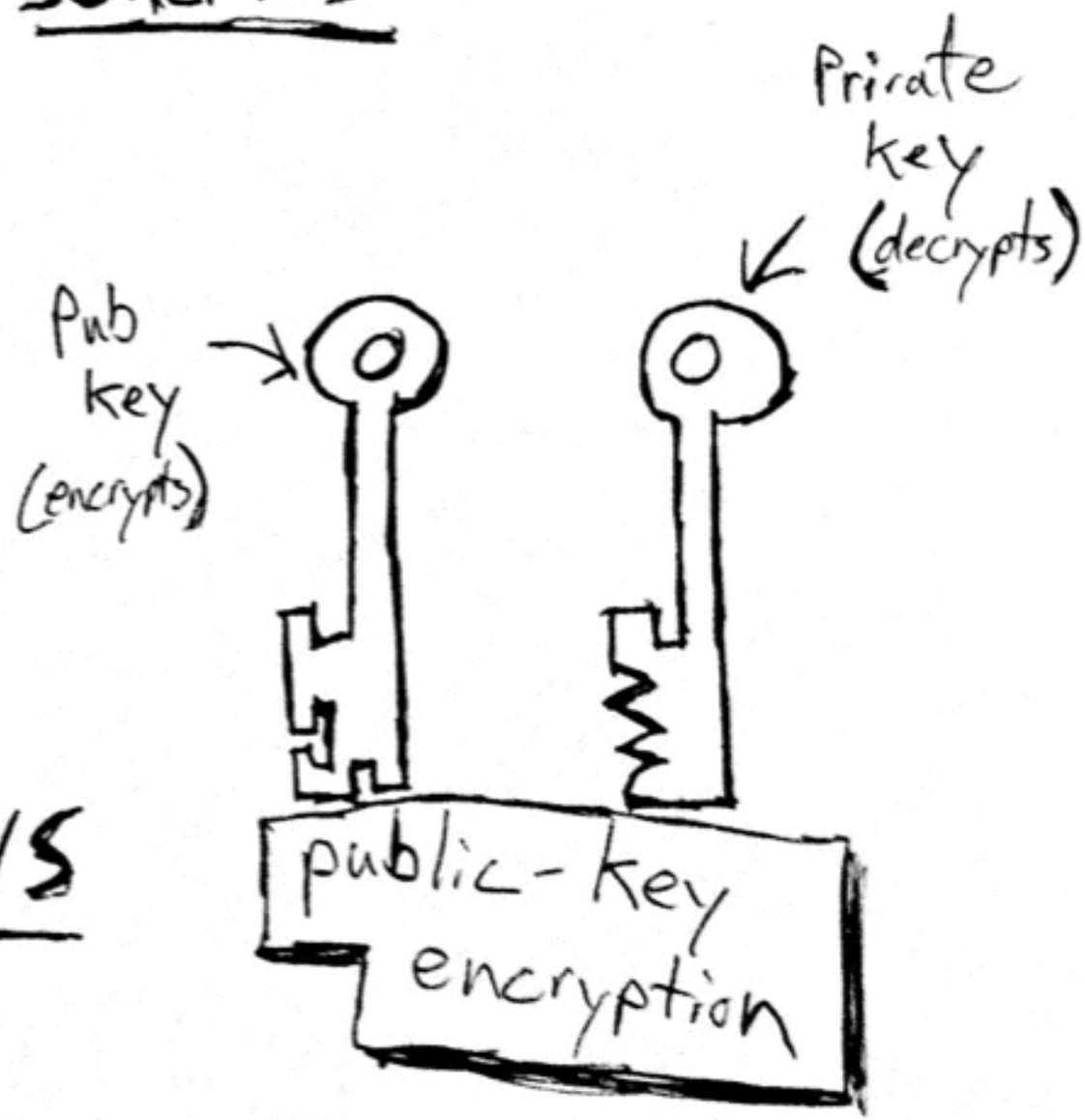


1991 CE
PGP introduced

# encryption schemes




VS



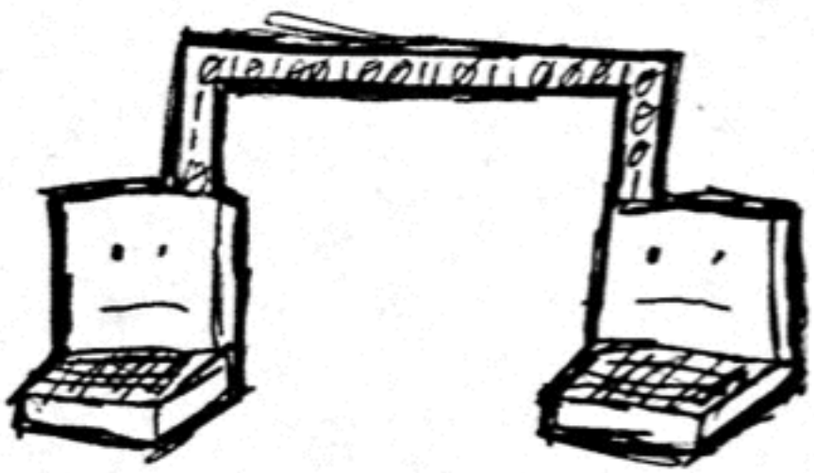
# Symmetric Keys

- Encryption and Decryption keys are the same
- Oldest Type of Encryption
- Computationally simple compared to public-key
- still used all the time for things like session keys

# Public-Key Encryption

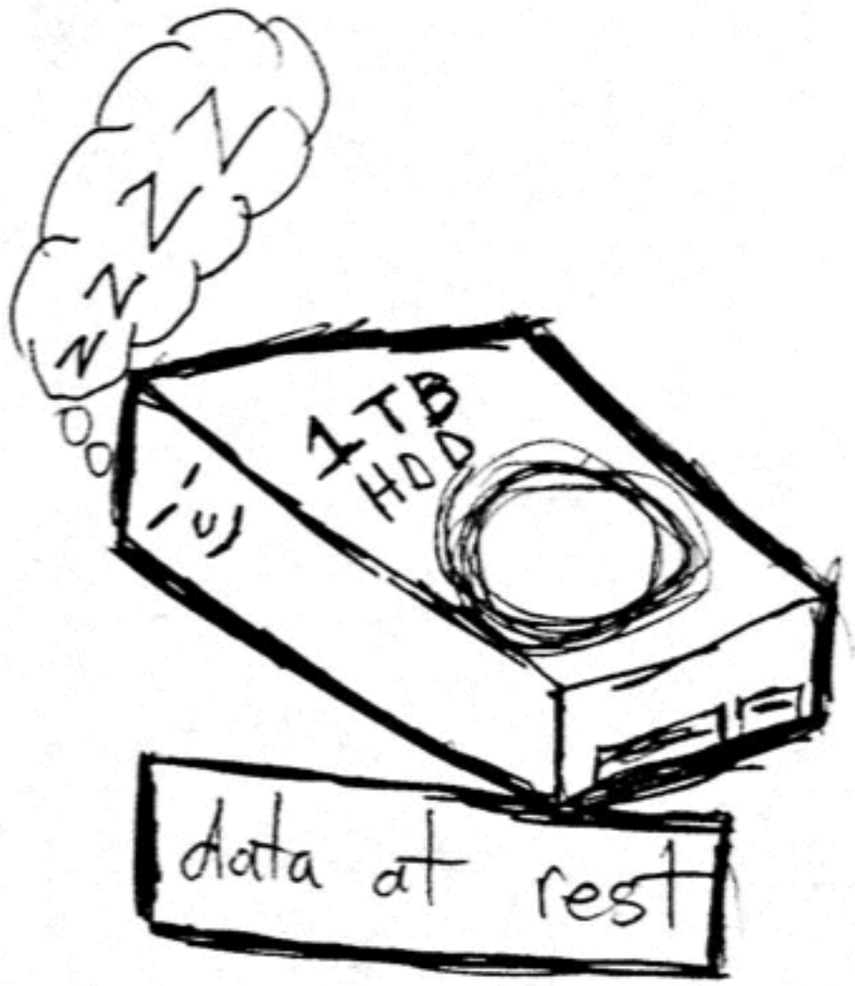
- Cipher text generated with public key
- Only private key is capable of decrypting
- Computationally complex compared to symmetric key
- very young - less than 40 year old

# encryption technology



data in transit

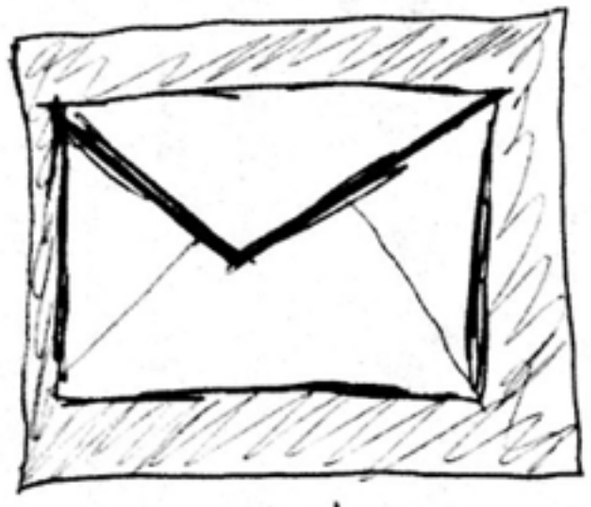
VS



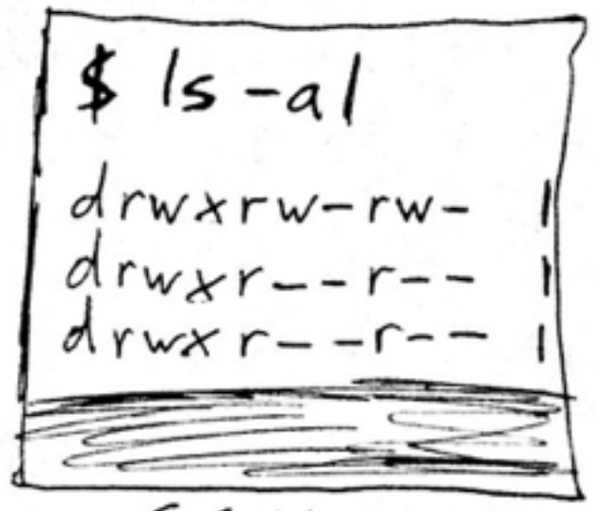

data at rest

MACTECH

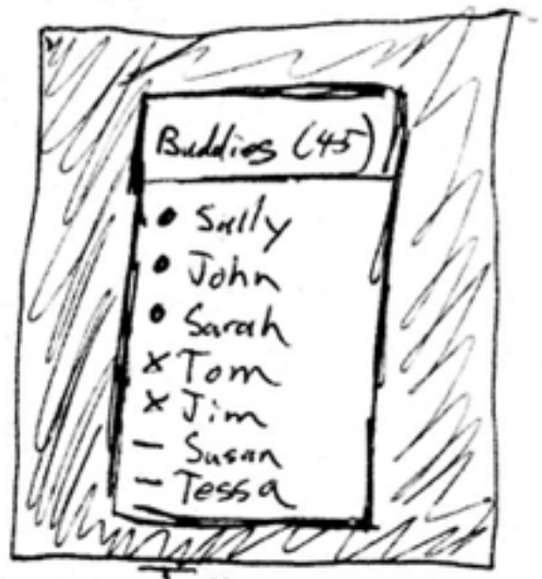
# technologies that use transit encryption



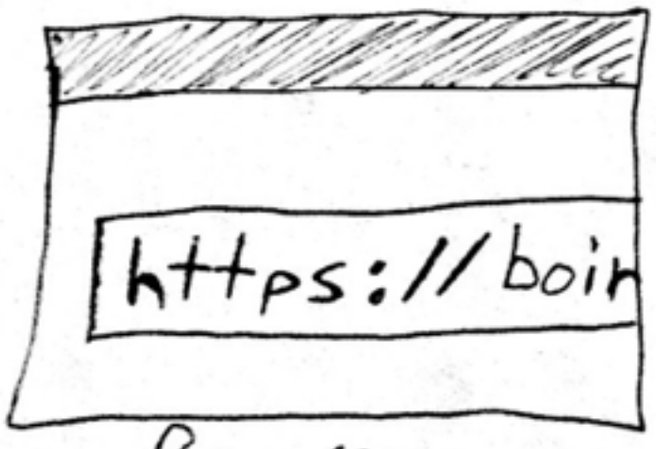
Email
(PGP/GPG)



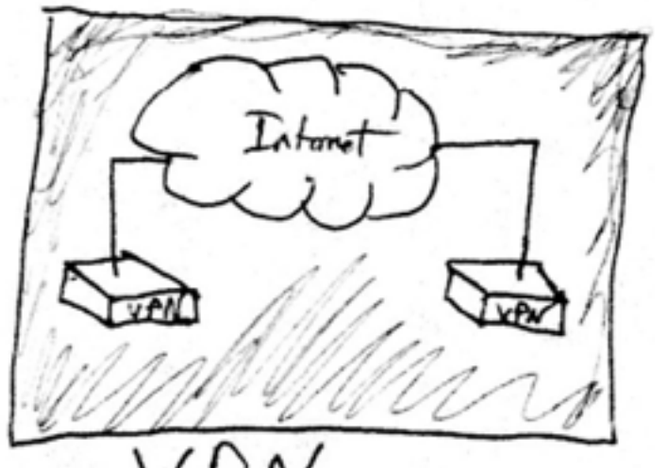

SSH
(Pub-key)




Jabber
(OTR)

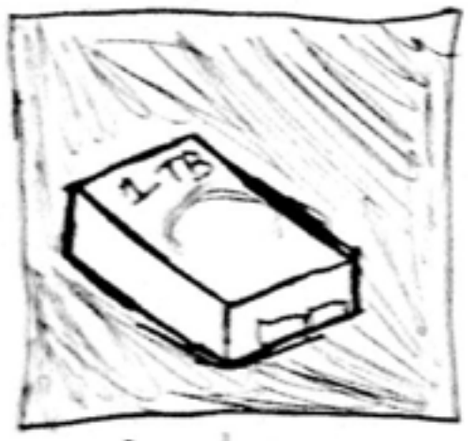



Browsers
(SSL/TLS)



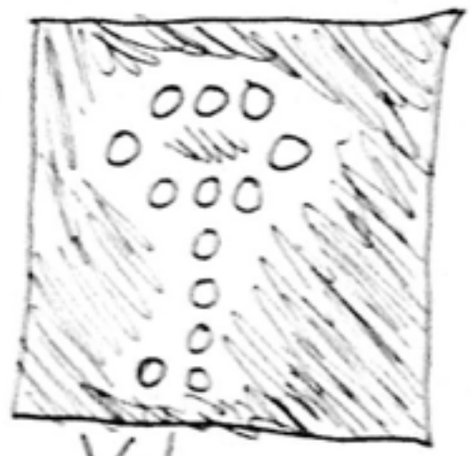

VPN
(Both)

# technologies that use encryption at rest



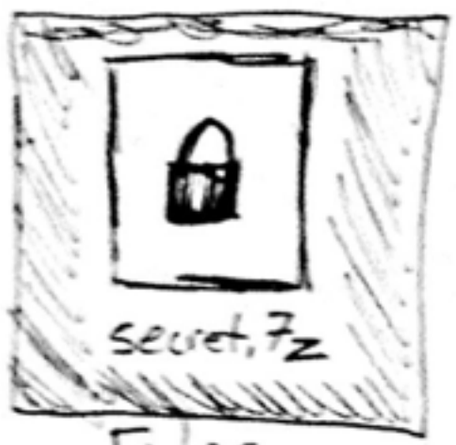

Disks

(WDE)



Volumes

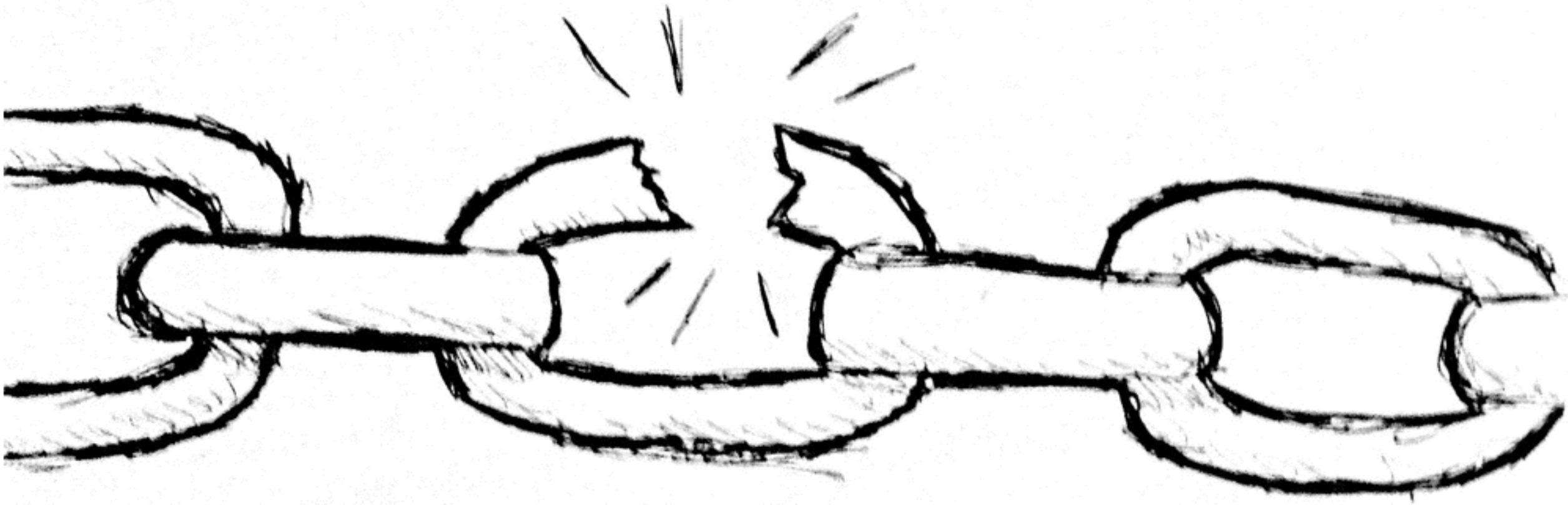
(TrueCrypt)




Files

(various)

# Weaknesses



Encryption works. Properly implemented strong crypto systems are one of the few things that you can rely on…

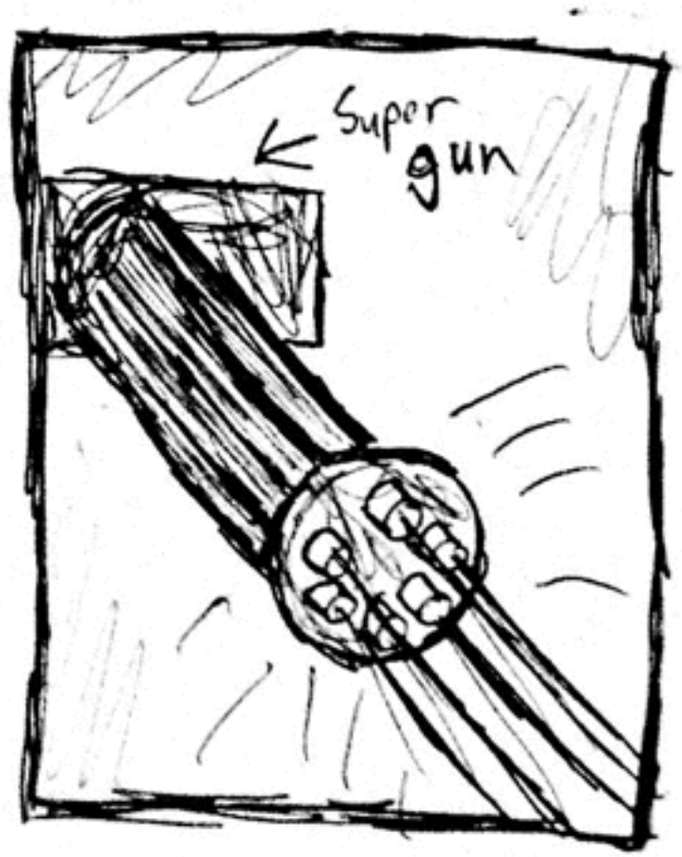
- Edward Snowden

…Unfortunately, endpoint security is so terrifically weak that NSA can frequently find ways around it.

- Edward Snowden

# Weaknesses

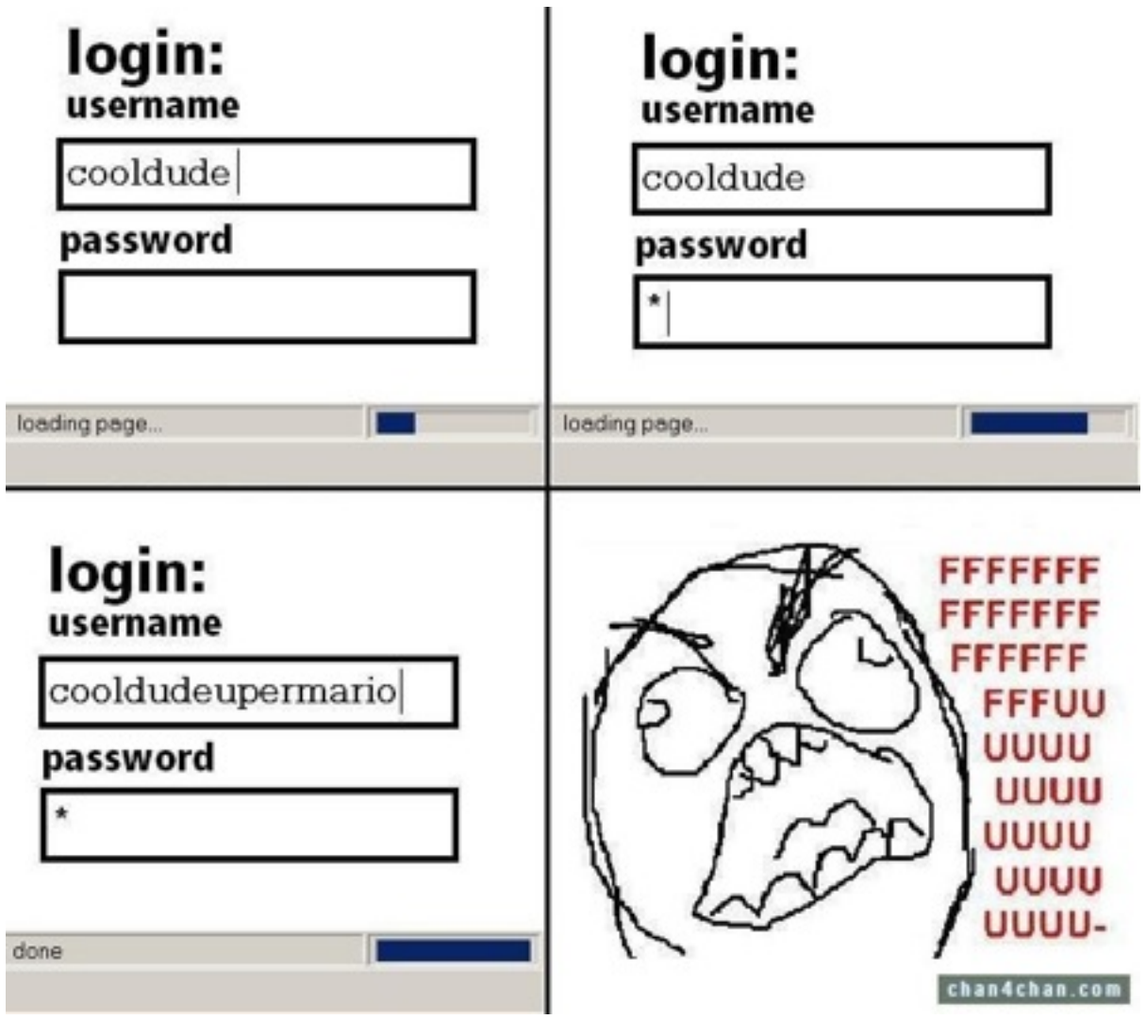
You




Bruteforce


Trust

# you


login:
username
cooldude
password
loading page...
login:
username
cooldude
password
*
loading page...
login:
username
cooldudeupermario
password
*
done
FFFFFFF
FFFFFFF
FFFFFF
FFFUU
UUUU
UUUU
UUUU
UUUU
UUUU-
chan4chan.com

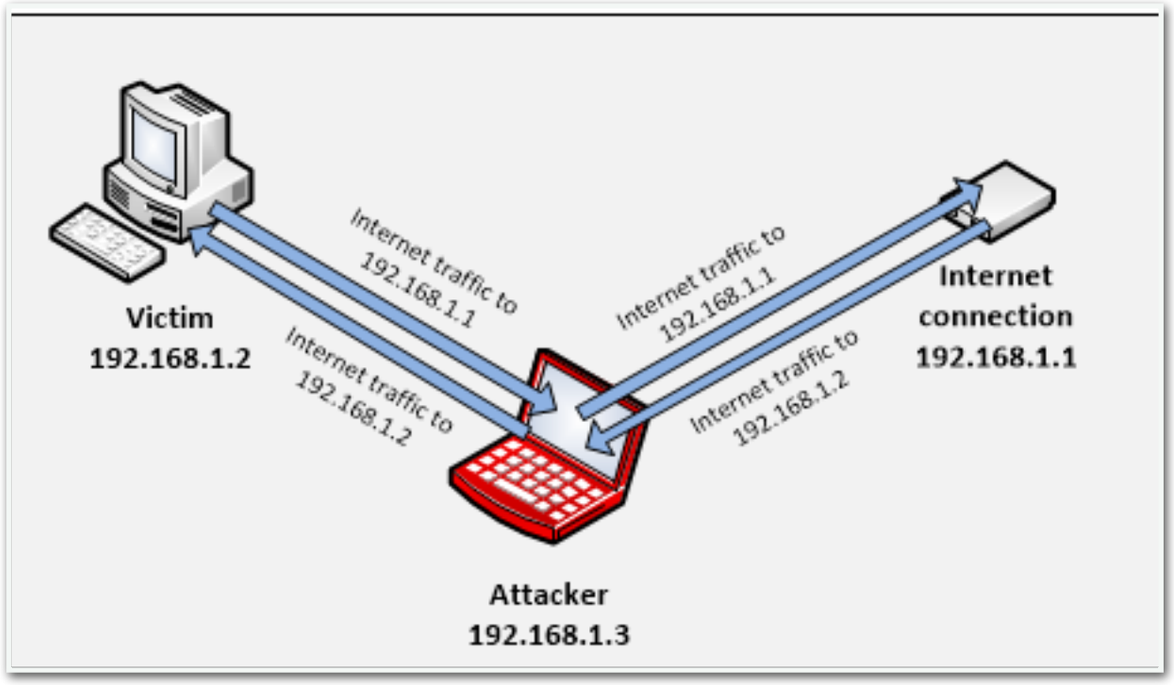
MACTECH

# brute force

```
c.abi.uni2.es [85.62.8.13] failed - POSSIBLE BREAK-IN ATTEMPT!
Sep 17 22:37:25 cerval sshd[27066]: Invalid user julia from 85.62.8.13
Sep 17 22:37:25 cerval sshd[27066]: error: Could not get shadow information for NOUSER
Sep 17 22:37:25 cerval sshd[27066]: Failed password for invalid user julia from 85.62.8.13 po
rt 33067 ssh2
Sep 17 22:37:26 cerval sshd[27068]: reverse mapping checking getaddrinfo for 85.62.8.13.stati
c.abi.uni2.es [85.62.8.13] failed - POSSIBLE BREAK-IN ATTEMPT!
Sep 17 22:37:26 cerval sshd[27068]: Invalid user julia123 from 85.62.8.13
Sep 17 22:37:26 cerval sshd[27068]: error: Could not get shadow information for NOUSER
Sep 17 22:37:26 cerval sshd[27068]: Failed password for invalid user julia123 from 85.62.8.13
 port 33222 ssh2
Sep 17 22:37:27 cerval sshd[27070]: reverse mapping checking getaddrinfo for 85.62.8.13.stati
c.abi.uni2.es [85.62.8.13] failed - POSSIBLE BREAK-IN ATTEMPT!
Sep 17 22:37:27 cerval sshd[27070]: Invalid user a from 85.62.8.13
Sep 17 22:37:27 cerval sshd[27070]: error: Could not get shadow information for NOUSER
Sep 17 22:37:27 cerval sshd[27070]: Failed password for invalid user a from 85.62.8.13 port 3
3365 ssh2
Sep 17 22:37:30 cerval sshd[27072]: reverse mapping checking getaddrinfo for 85.62.8.13.stati
c.abi.uni2.es [85.62.8.13] failed - POSSIBLE BREAK-IN ATTEMPT!
Sep 17 22:37:30 cerval sshd[27072]: Invalid user julie from 85.62.8.13
Sep 17 22:37:30 cerval sshd[27072]: error: Could not get shadow information for NOUSER
Sep 17 22:37:30 cerval sshd[27072]: Failed password for invalid user julie from 85.62.8.13 po
rt 33488 ssh2
Sep 17 22:37:31 cerval sshd[27074]: reverse mapping checking getaddrinfo for 85.62.8.13.stati
c.abi.uni2.es [85.62.8.13] failed - POSSIBLE BREAK-IN ATTEMPT!
Sep 17 22:37:31 cerval sshd[27074]: Invalid user julie123 from 85.62.8.13
Sep 17 22:37:31 cerval sshd[27074]: error: Could not get shadow information for NOUSER
Sep 17 22:37:31 cerval sshd[27074]: Failed password for invalid user julie123 from 85.62.8.13
 port 35041 ssh2
Sep 17 22:37:32 cerval sshd[27076]: reverse mapping checking getaddrinfo for 85.62.8.13.stati
c.abi.uni2.es [85.62.8.13] failed - POSSIBLE BREAK-IN ATTEMPT!
Sep 17 22:37:32 cerval sshd[27076]: Invalid user a from 85.62.8.13
Sep 17 22:37:32 cerval sshd[27076]: error: Could not get shadow information for NOUSER
Sep 17 22:37:32 cerval sshd[27076]: Failed password for invalid user a from 85.62.8.13 port 3
5757 ssh2
Sep 17 22:37:32 cerval sshd[27078]: reverse mapping checking getaddrinfo for 85.62.8.13.stati
c.abi.uni2.es [85.62.8.13] failed - POSSIBLE BREAK-IN ATTEMPT!
Sep 17 22:37:32 cerval sshd[27078]: Invalid user june from 85.62.8.13
Sep 17 22:37:32 cerval sshd[27078]: error: Could not get shadow information for NOUSER
Sep 17 22:37:32 cerval sshd[27078]: Failed password for invalid user june from 85.62.8.13 por
t 35930 ssh2
Sep 17 22:37:33 cerval sshd[27080]: reverse mapping checking getaddrinfo for 85.62.8.13.stati
c.abi.uni2.es [85.62.8.13] failed - POSSIBLE BREAK-IN ATTEMPT!
Sep 17 22:37:33 cerval sshd[27080]: Invalid user june123 from 85.62.8.13
Sep 17 22:37:33 cerval sshd[27080]: error: Could not get shadow information for NOUSER
Sep 17 22:37:33 cerval sshd[27080]: Failed password for invalid user june123 from 85.62.8.13
port 36297 ssh2
Sep 17 22:37:34 cerval sshd[27082]: reverse mapping checking getaddrinfo for 85.62.8.13.stati
c.abi.uni2.es [85.62.8.13] failed - POSSIBLE BREAK-IN ATTEMPT!
Sep 17 22:37:34 cerval sshd[27082]: Invalid user a from 85.62.8.13
```

MACTECH

# trust


Victim
192.168.1.2
Internet traffic to
192.168.1.1
Internet traffic to
192.168.1.2
Internet traffic to
192.168.1.1
Internet traffic to
192.168.1.2
Internet
connection
192.168.1.1
Attacker
192.168.1.3

# trust


MACTECH

# examples

# The Heartbleed Bug

Bug is in the OpenSSL's implementation of the TLS/DTLS (transport layer security protocols) heartbeat extension (RFC6520). When it is exploited it leads to the leak of memory contents from the server to the client and from the client to the server.

# goto fail

```
static OSStatus
SSLVerifySignedServerKeyExchange(SSLContext *ctx, bool isRsa, SSLBuffer signedParams,
                                 uint8_t *signature, UInt16 signatureLen)
{
    OSStatus        err;
    SSLBuffer       hashOut, hashCtx, clientRandom, serverRandom;
    uint8_t         hashes[SSL_SHA1_DIGEST_LEN + SSL_MD5_DIGEST_LEN];
    SSLBuffer       signedHashes;
    uint8_t         *dataToSign;
    size_t          dataToSignLen;

    ...
    if ((err = ReadyHash(&SSLHashSHA1, &hashCtx)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &clientRandom)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
        goto fail;
        goto fail;
    if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
        goto fail;

    err = sslRawVerify(ctx,
                       ctx->peerPubKey,
                       dataToSign,              /* plaintext */
                       dataToSignLen,           /* plaintext length */
                       signature,
                       signatureLen);
    if(err) {
        sslErrorLog("SSLDecodeSignedServerKeyExchange: sslRawVerify "
                    "returned %d\n", (int)err);
        goto fail;
    }
```

MACTECH

# Apple Technology

MACTECH

# Filevault 2

# Filevault 2

- Encrypts the entire partition
- Supports master-key for easy multi-user decrypt
- Has a flexible command line tool ( fdesetup )
- Supports mass-deployable recovery key


```
➜  ~  fdesetup status
FileVault is On.
```

# ShadowHashes

```
➜ ~  dscl localhost -read /Local/Default/Users/nathan AuthenticationAuthority
AuthenticationAuthority: ;ShadowHash;HASHLIST:<SALTED-SHA512-PBKDF2> ;Kerberosv5;;nathan
@LKDC:SHA1.A794B135C0E449B1BB1643699AFBB428ACEEB069;LKDC:SHA1.A794B135C0E449B1BB1643699A
FBB428ACEEB069
➜ ~  dscl localhost -read /Local/Default/Users/rojoadmin AuthenticationAuthority
AuthenticationAuthority: ;ShadowHash;HASHLIST:<SALTED-SHA512-PBKDF2> ;Kerberosv5;;rojoad
min@LKDC:SHA1.EB1AAAEA9F56AA68751754782F316FC3D135DD21;LKDC:SHA1.EB1AAAEA9F56AA687517547
82F316FC3D135DD21
```

```
dscl localhost -read /Local/Defaults/Users/USERNAME AuthenticationAuthority
```

- Hashes are one-way with fixed length output
- OS X password hashes are stored in DSCL
- Shadow Hashes us a Salted SHA512 HASH

# How Certificates Work

2 Are you really https://cool-co.biz ?

cool-co.biz

3


Yes, here is my certificate and public key.

5

Yep

Certificate granted to www.pretendco.com

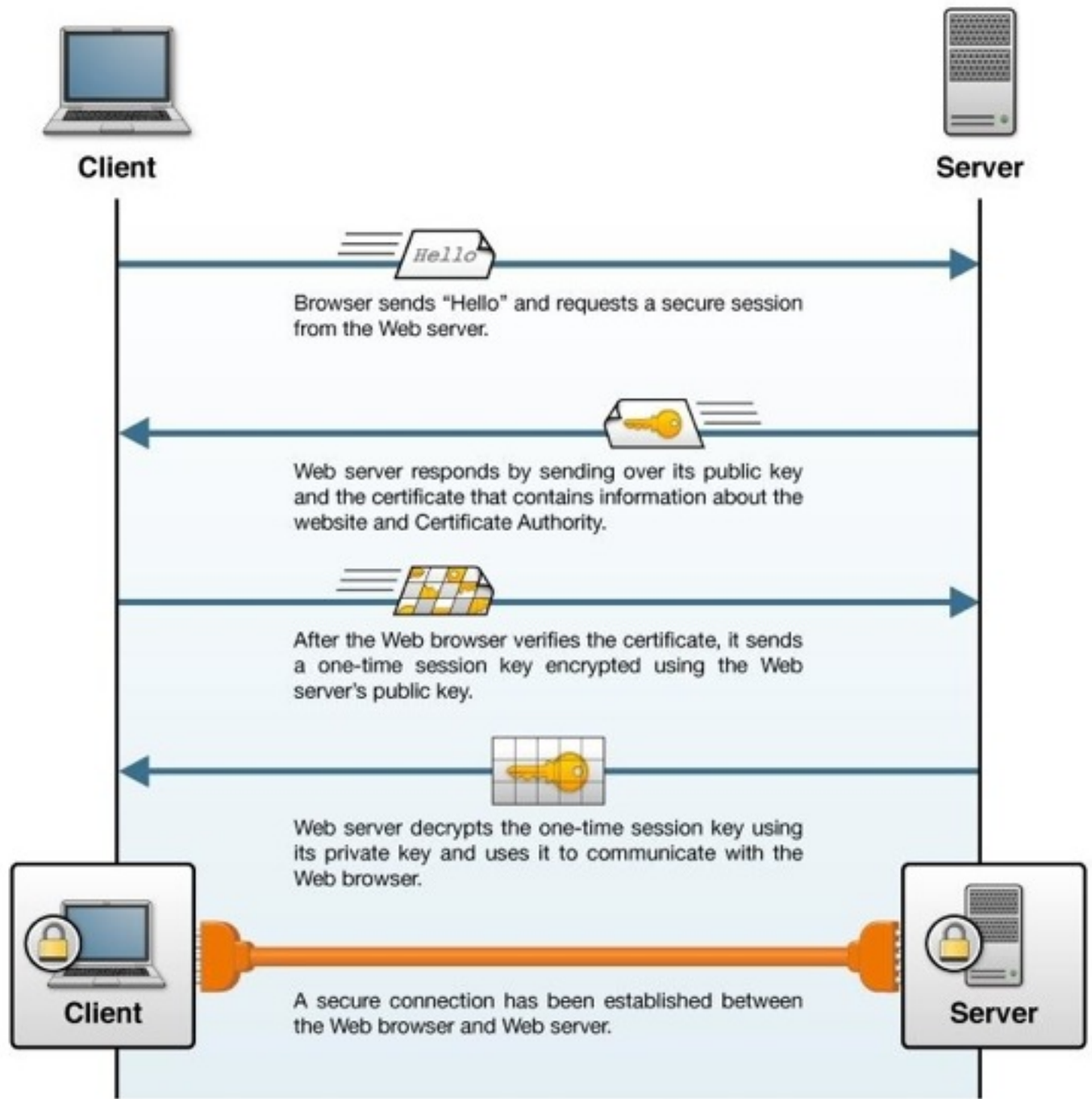
1

4 Is this certificate valid?

http://support.apple.com/kb/HT5012

MACTECH

# How TLS works

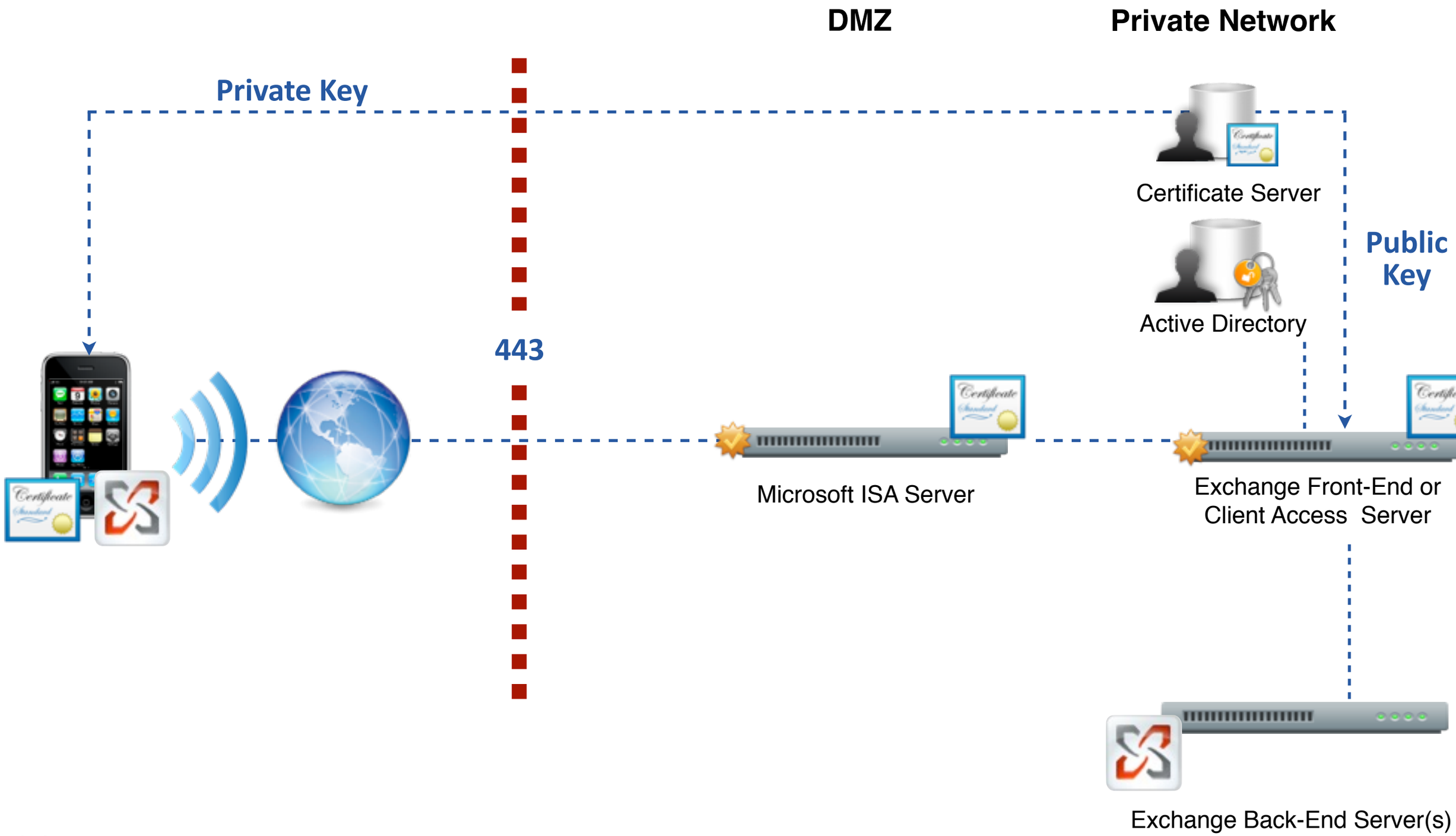


# Certificate Driven Profiles

1. Device generates public/private key pair.
2. Secure communication between device and management server.
3. Certificate signing request incorporating the device-generated public key.
4. CA signs certificate and returns it to enrollment server.
5. Device receives and installs signed certificate.


MACTECH

# Microsoft Exchange ActiveSync




MACTECH

# Web Server

DMZ

Private Network

VPN Auth/Token Server

VPN Server/Concentrator

LDAP Server

Web Server - HR

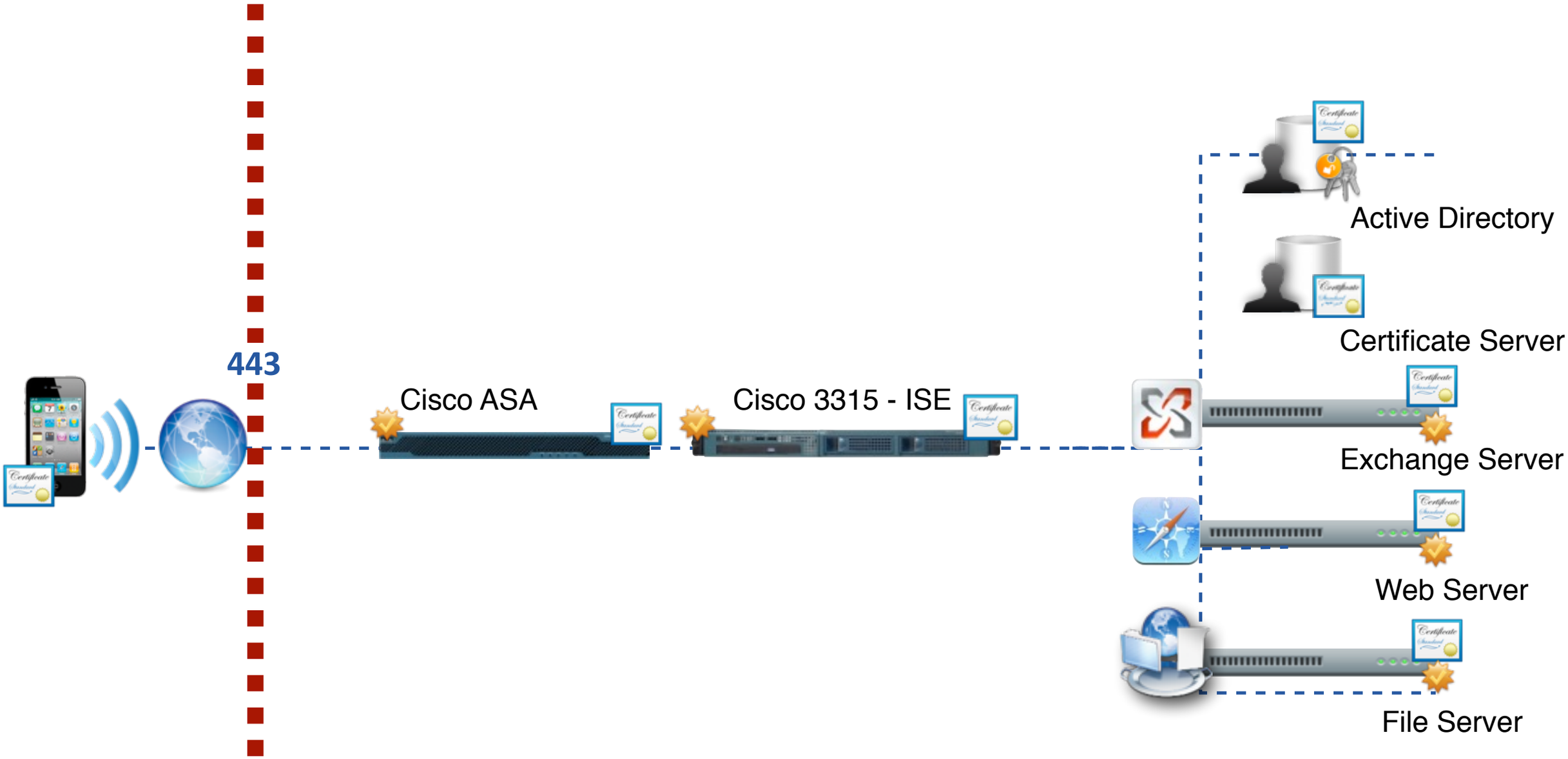
Proxy Server

Web Server - Sales

Web Server - Internal

MACTECH

# How it Works




MACTECH

questions?

MACTECH