RAIDERS of the LOST CERTIFICATE

Paul Suh
paul.suh@ps-enable.com
http://ps-enable.com

$ ps | Enable

What is a Certificate?

```
-----BEGIN CERTIFICATE-----
MIIDNDCCAp2gAwIBAgIDDG3kMA0GCSqGSIb3DQEBBQUAME4xCzAJBgNVBAYTAlVT
MRAwDgYDVQQKEwdFcXVpZmF4MS0wKwYDVQQLEyRFcXVpZmF4IFNlY3VyZSBDZXJ0
aWZpY2F0ZSBBdXRob3JpdHkwHhcNMDkwODE0MTIyODI1WhcNMTAwOTE1MDgzNjU0
WjCBvjELMAkGA1UEBhMCVVMxGjAYBgNVBAoTEW1haWwuZ29vZGVhc3QuY29tMRMw
EQYDVQQLEwpHVDE1MjczNTkzMTEwLwYDVQQLEyhTZWUgd3d3LnJhcGlkc3NsLmNv
bS9yZXNvdXJjZXMvY3BzIChjKTA5MS8wLQYDVQQLEyZEb21haW4gQ29udHJvbCBW
YWxpZGF0ZWQgLSBSYXBpZFNTTChSKTEaMBgGA1UEAxMRbWFpbC5nb29kZWFzdC5j
b20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBALfkfK1/GXjZ9ElME5FBRAic
ELomSkAyLSf7lJkoizNx9TjmQxvhKOOOY4BZha7Ppu65gf561MpUPmpnE+NvJCyP
h0jdZOLniovAAVJAyy6gCb7XnzPYPXR7ei8OVqX+NSxl4Wvl1GD2Cda4Uvg7A949
3s5Dpo8ufWd9A+Lmz8RdAgMBAAGjga4wgaswDgYDVR0PAQH/BAQDAgTwMB0GA1Ud
DgQWBBRdSbSgosLIWuz1Yk48krPNNaMa9zA6BgNVHR8EMzAxMC+gLaArhilodHRw
Oi8vY3JsLmdlb3RydXN0LmNvbS9jcmxzL3NlY3VyZWNhLmNybDAfBgNVHSMEGDAW
gBRI5mj5K9KylddH2CMgEE8zmJCf1DAdBgNVHSUEFjAUBggrBgEFBQcDAQYIKwYB
BQUHAwIwDQYJKoZIhvcNAQEFBQADgYEAb83ueDKHAUQ2kKx85OjkZJLm7fI5Ah59
z+Qe3uO+2bXQmjfTKXZvFspNN03ffBYsroqrKF6PnJOGRSDaqX5E6OINbG23hoiu
phCk7Clcq6JFMGwXPFJIdJEP3g3/8bJQLMgsODNCEOKyNWlAwEJFw33lJ4+suXHK
```

No, really…

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 814564 (0xc6de4)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=US, O=Equifax, OU=Equifax Secure Certificate Authority
        Validity
            Not Before: Aug 14 12:28:25 2009 GMT
            Not After : Sep 15 08:36:54 2010 GMT
        Subject: C=US, O=mail.goodeast.com, OU=GT15273593, OU=See
www.rapidssl.com/resources/cps (c)09, OU=Domain Control Validated -
RapidSSL(R), CN=mail.goodeast.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:b7:e4:7c:ad:7f:19:78:d9:f4:49:4c:13:91:41:
                    44:08:9c:10:ba:26:4a:40:32:2d:27:fb:94:99:28:
```

A Little More Basic, Please?

1. Choose two distinct prime numbers p and q.
   Compute $n = pq$.
   Compute $\phi(n) = (p - 1)(q - 1)$, where ϕ is Euler's totient function.
   Choose an integer e such that $1 < e < \phi(n)$ and $\gcd(e,\phi(n)) = 1$ (i.e., e and $\phi(n)$ are coprime).
   Determine $d = e^{-1} \pmod{\phi(n)}$. (i.e., d is the multiplicative inverse of $e \pmod{\phi(n)}$).

An alternative, used by PKCS#1, is to choose d matching with , where is the least common multiple. Using λ instead of $\phi(n)$ allows more choices for d. λ can also be defined using the Carmichael function, $\lambda(n)$.

OK, You've Really Lost Me

1. Allows two sides to communicate securely without exchanging secret codes beforehand
2. Assures the identity of the certificate holder

Symmetric Ciphers

Both sides must have the same secret key

Keys can be simple or complex

Scytale

Enigma

JN-25

One-time pad

DES, 3DES, AES

Use of a Symmetric Key

The Trouble with Symmetric Ciphers

Making sure all of the users of a code have the same key

A.k.a., the “Key Distribution Problem”

More complex ciphers are more secure but make the Key Distribution Problem worse

Submarine I-1 and JN-25

- Submarine I-1 sunk with copies of codes
- Salvaged by Allied forces
- All Japanese Naval codes considered compromised in 1943

$ ps | Enable

VENONA Project

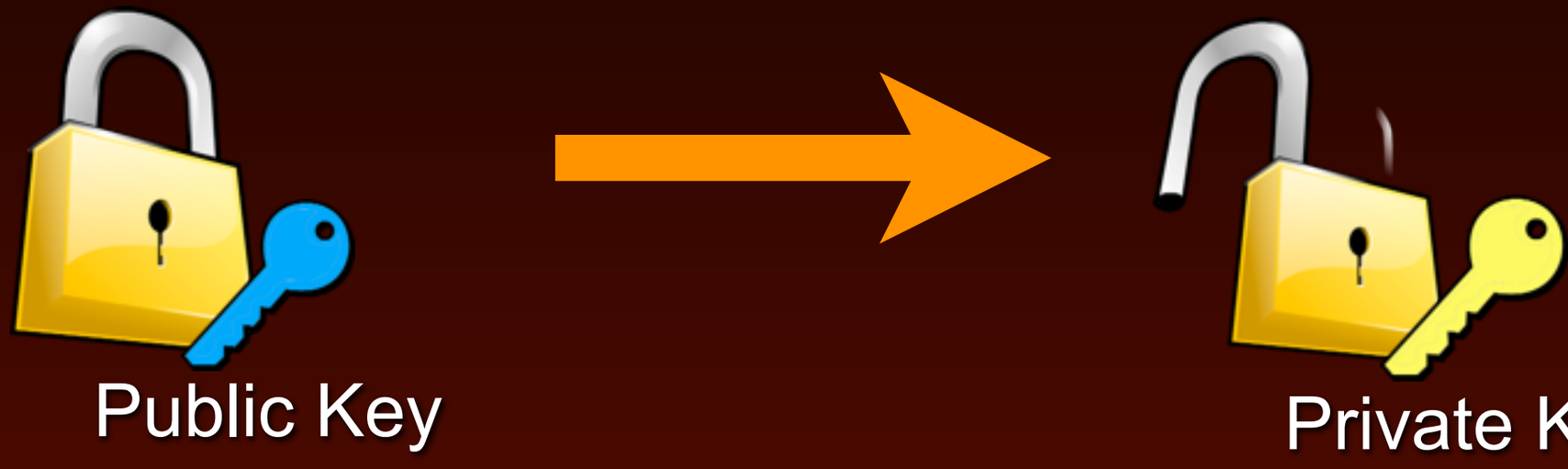
- High demand for code pads caused Soviets to re-use some one-time pads
- US was able to read some Soviet message traffic encrypted with the re-used pads

Ron Rivest, Adi Shamir, Leonard Adleman

R S A

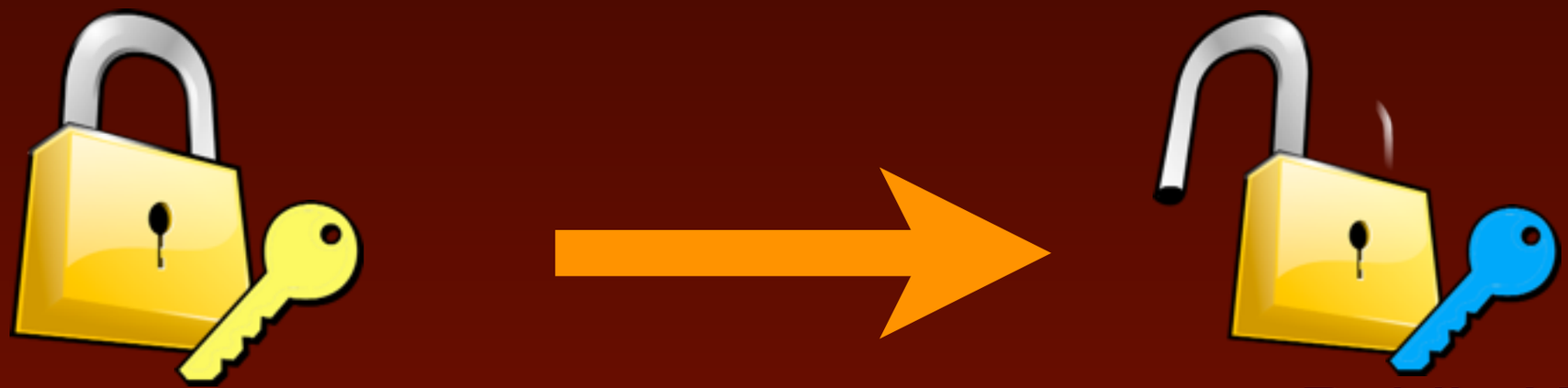
- Originally discovered by James H. Ellis, Clifford Cocks, and Malcolm Williamson at GCHQ in the UK in 1973

How Does RSA Work?



Public Key

Private Key



Private Key

Public Key

Public Key Encryption

Public Key
Alice

Private Key
Alice

Public Key
Alice

Digital Signature

Man in the Middle

Public Key
“Alice”

Obtaining a Certificate

Public Key
Private Key

CSR

Private Key
CA

Certificate Chains

Root Certificate Authority

Intermediate Certificate Authority

Leaf Certificate

A Certificate is a Public Key Plus Identification Info Digitally Signed by Some Entity That You Trust

Which Certificate Authority?

- Commercial or government CA
  - Verisign, StartSSL, DoD, ...
- Internal CA
  - Active Directory
  - Open Directory
  - MDM

Hash Functions

Advanced checksum

Easy to determine if a message was changed

One-way function

~~MD-5 – 128 bits~~

~~SHA-1 – 160 bits~~

SHA-2 – 256, 384, 512 bits

SHA-3 – in the works at NIST

Uses of Hash Functions

- Hash-based Message Authentication Code
  - Prevent an attacker from making changes
  - HMAC-MD5
  - HMAC-SHA1
- Password-Based Key Derivation Function
  - PBKDF2
  - bcrypt
  - scrypt

Certificate Elements

Version

Serial Number

Algorithm ID

Issuer

Not Valid Before

Not Valid After

Subject

Subject Public Key Info

Public Key Algorithm

Subject Public Key

Extensions (Optional)

Certificate Signature Algorithm

Certificate Signature

Certificate Elements: Subject

Servers: LDAP-style identifier

```
o=Company,ou=Department,cn=www.example.com
```

E-mail: E-mail address extension

```
o=Company,ou=Department,cn=Alice Doe/
emailAddress=alice.doe@example.com
```

Subject Alternative Name Certificates

Also called Unified Communications Certificate (UCC)

Common with Microsoft Exchange

Has Subject Alternative Name attribute

```
DNS Name=www.example.com

DNS Name=wiki.example.com
```

Certificate Revocation

Certificate Revocation List (CRL)

Online Certificate Status Protocol (OCSP)

Certificate Types

TLS Server

Key encipherment, key agreement, digital signature

TLS Client

Digital signature, client authentication

S/MIME

Digital signature, e-mail protection

Code signing

Digital signature, code signing

PKCS File Types

PKCS#1: RSA private key (.key)

PCKS#7: S/MIME E-mail (.p7b, .p7c)

PKCS#8: Private key (.p8)

PKCS#10: Certificate signing request (.csr, .req)

PKCS#12: Encrypted private key and certificates (.p12, .pfx)

DER vs. PEM encoding

http://ps-enable.com/articles/Certificate_file_types.html

Certificate and Key Handling

Do NOT expose private key!

Store key and certificate in PKCS#12 or encrypted disk image

Mark files with purpose and expiration date

Keep backups

SSL and TLS

Secure Sockets Layer 1.0, 2.0, 3.0

Transport Layer Security 1.0, 1.1, 1.2

Protocol for negotiating certificate and key exchange

TLS vs. STARTTLS

Server Name Indication

TLS

TLS version
Server Name Indication
Allowed ciphers
Random number

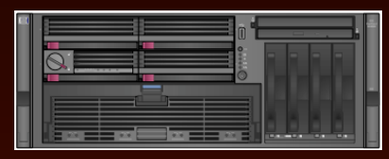
Certificate chain
Allowed ciphers
Random number

Selected cipher
Pre-master key

Both sides generate master key from Pre-master + two random numbers

Both sides generate master key from Pre-master + two random numbers

STARTTLS



Connection request

Capabilites info

STARTTLS command

TLS negotiation
as previous example

TLS negotiation
as previous example

Test Certificate Usage

```
openssl s_client -connect host:port
     <-servername name>

openssl s_client -connect host:port
     -starttls smtp
```

Server Name Indication

Gets around requirement of one IP address per TLS server

Browsers

Safari 3.0 (10.5.6), Mobile Safari (iOS 4.0), Firefox 2.0, Chrome 5.0.342.1 (6 on Win XP), IE 7 (Vista or higher)

Servers

Apache 2.2.12, IIS 8, Nginx 0.5.32 (with correct OpenSSL), Tomcat with Java 7

Simple Certificate Enrollment Protocol

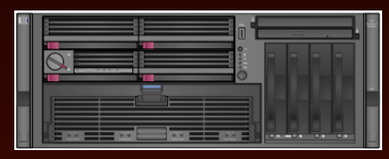
Allows end users to request certificates

Used by devices to request client authentication certificates from MDM

Port 1640/tcp

SCEP Exchange

This is why you have to install the trust profile before you can enroll a device!



CA certificate request

Client validates CA certificate
Client generates key pair

CA certificate

CSR inside PKCS#7
Authentication data

Polling request

Status: pending

Device certificate inside PKCS#7

Sources of Trust

/System/Library/Keychains/

SystemRootCertificates.keychain

SystemCACertificates.keychain (intermediates)

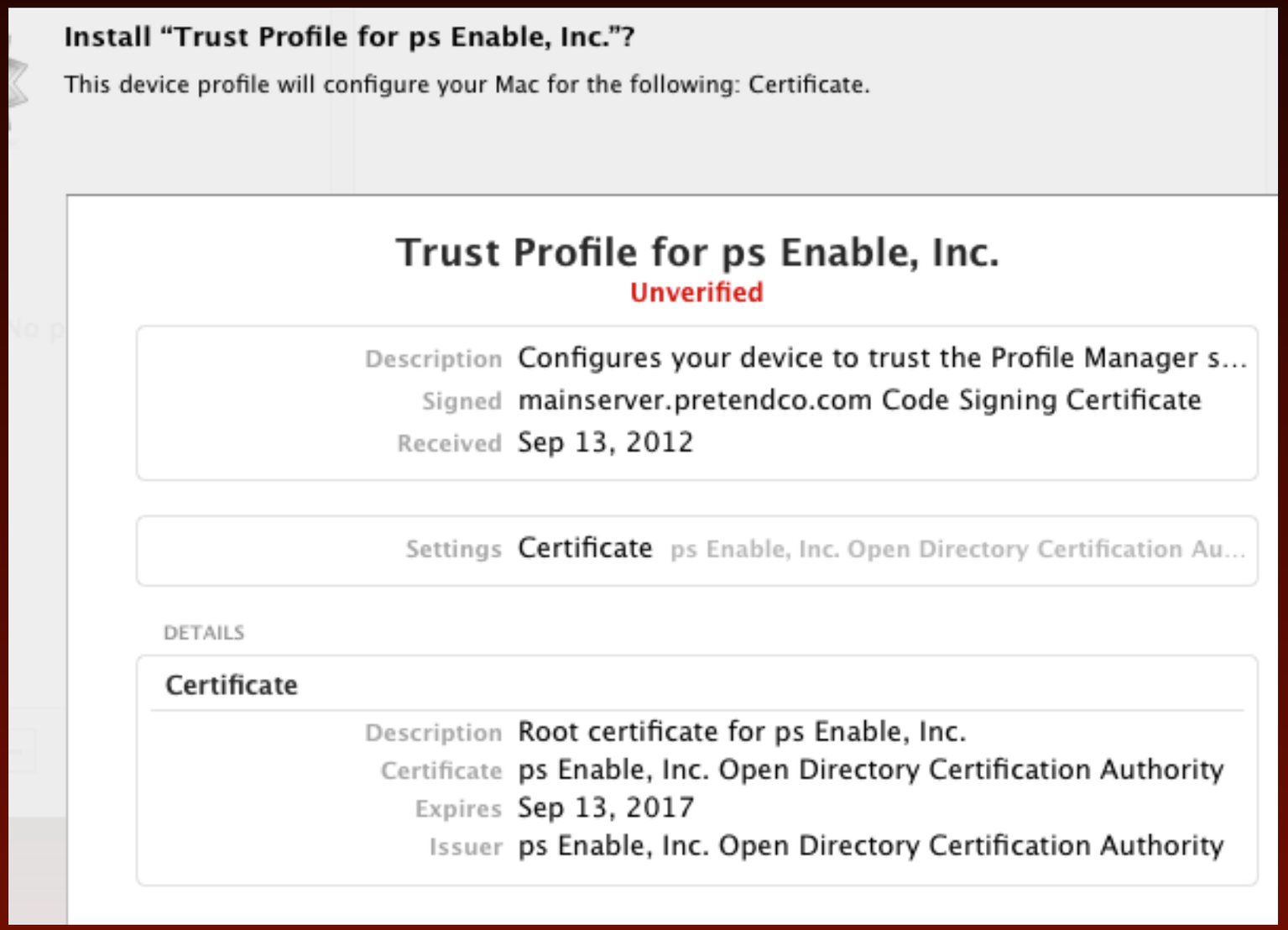
EVRoots.plist

X509Anchors

/Library/Keychains/System.keychain

~/Library/Keychains/login.keychain

Trust Profiles



Browsers and Trust

Safari and Chrome use Keychain

Firefox does not!

Problems with PKI

How many Certificate Authorities in System Roots?

- Over 200
- Google for “Diginotar”

Certificate revocation

- OCSP off due to privacy leaks
- CRL is too big

Problems with PKI

Proliferation of private roots

- Profile Manager / MDM
- Active Directory PKI
- Deep packet inspection appliances

Domain validation vs. Extended validation

Problems with PKI

- Recent attacks on implementations
  - Null-terminated strings
  - BEAST / CRIME / BREACH
- Weaknesses in underlying crypto
  - MD5 is dead
  - SHA-1 is fading
  - Weak PRNGs

RAIDERS of the LOST CERTIFICATE

Paul Suh
paul.suh@ps-enable.com
http://ps-enable.com

$ ps | Enable