INF333 2023-2024 Spring Semester

Burak Arslan

Lab VIII

Apr 04th, 2024
Due: Apr 04th, 10:59:59

1 A Not-so-gentle introduction to PKI (100pts)

This lab will introduce you to the wonderful world of Public Key Cryptography. This is how you get the padlock (🔒) icon next to the web sites that you visit in your browser using the `https` protocol.

Public Key Cryptography

Alice wants to meet with his friend Bob, an engineer at ACME corp, to discuss about the suprise birthday party they are organizing for their friend Charlie. Since Charlie is the owner of an industrial conglomerate that listens to the entire Internet traffic, she has to do this secretly to prevent Charlie from finding out.

Unfortunately, she doesn't have access to a separate secure network, so she needs to use the Internet like the rest of us to set up the meeting.

Here's her secret message:

```
From: alice@example.com
To: bob@example.com
Subject: Charlie's Birthday Party: The Guest List
Date: Wed, Apr 3rd, 2024

I will be waiting for you tomorrow at 14:00 in front of the Burger King
at Istiklal to go over the guest list one last time.

--Alice
```

Now that the message is ready, she has two options to keep this message secret:

Symmetric cryptography; where the same key is used to both for encryption and decryption operations.

Asymmetric cryptography; where the public key is used for encryption and the private key is used for decryption operations.

Using symmetric cryptography is out of question since she has no means to communicate the secret key to Bob. If she performs the encryption and attaches the key to the message so that Bob can decrypt it, it defeats the whole purpose since anyone can decrypt the message. Helpless, she resorts to asymmetric cryptography and opens with a message that asks Bob for his public key:

```
From: alice@example.com
To: bob@example.com
Subject: What is your public key?
Date: Wed, Apr 3rd, 2024

Can you send me your public key so that I can send you a secret message?

--Alice
```

Bob responds promptly:

```
From: bob@example.com
To: alice@example.com
Subject: Re: What is your public key?
Date: Wed, Apr 3rd, 2024

Here it is: <KEY-BOB.pub>

--Bob
```

However, since it's Charlie they are talking about, they need to be sure that Charlie did not intercept and alter Bob's key on the way to Alice. This is called a **Man in the Middle** (MITM) attack. Alice, already aware of this possibility, replies:

```
From: alice@example.com
To: bob@example.com
Subject: Re: What is your public key?
Date: Wed, Apr 3rd, 2024

Did your brain fall out of your head, Bob? Charlie can intercept and
alter your message so you need to send me a signed certificate. I was
in ACME headquarters recently so I downloaded their public key from their
systems personally. Use the certificate with
dn="C=US, O=ACME, CN=Acme Corporation"[1] to sign your public key so I can
verify it's really you.

--Bob
```

The fundametal principle of trust in computer networks is this: **Trust can never be created from nothing**, it can only converted to other forms of trust. Since Alice went to ACME HQ and got the certificate personally, she can trust that she has the authentic ACME certificate.

Public Key Infrastructure (PKI) is a security architecture where trust is conveyed through the signature chain of such Certificate Authorities (CA).

Q1.1. Name a Turkish root CA operator that is trusted by the prominent browser vendors.

Public Key Cryptography

So Bob in turn slaps a Certificate Signing Request (CSR) on top his public key and sends it to the Acme Corp. using his secure company VPN:

```
From: bob@vpn.example.com
To: acme@vpn.example.com
Subject: Please sign my public key
Date: Wed, Apr 3rd, 2024

Here it is: <BOB.csr>

--Bob
```

Since the ACME corp trusts the VPN that they operate, they are sure that this message is really sent by Bob the engineer that works for them.

[1] A distinguished name (DN) is a term that describes the identifying information in a certificate and is part of the certificate itself. Other common fields include: `C`=Country, `O`=Organization, `CN`=Canonical Name

```
From: acme@vpn.example.com
To: bob@vpn.example.com
Subject: Re: Please sign my public key
Date: Wed, Apr 3rd, 2024

Here is your signed public key: <BOB-ACME.pem>

--Dylan from Acme CA Dept
```

Bob forwards his brand new certificate to Alice.

```
From: bob@example.com
To: alice@example.com
Subject: Re: What is your public key?
Date: Wed, Apr 3rd, 2024

Here it is: <BOB-ACME.pem>

--Bob
```

Alice finally sends Bob the original message, using a secure encapsulation scheme named S/MIME, encrypted by bob's certificate.

Now that we made it to the happy ending, we are wondering about the contents of the files in the above message exchange.

To quench our curiosity, implement the following in a bash script named `pki.sh`:

Q1.2. Use the `openssl x509` command to export the details of the certificate from the question 1.1 to a file named `realroot.txt`

Q1.3. Use the `openssl req` command to write the Acme Corp certificate signing request to a file named `acme.csr` and the corresponding private key to a file named `acme.key`

Q1.4. Since ACME is a root CA, use the `openssl ca -selfsign` command to sign the Acme Corp certificate by Acme Corp and create `acme.pem`.

This is called a **self-signed certificate**. You are not supposed to self-sign certificates unless you are the root CA.

Q1.5. Use the `openssl req` command to generate a CSR for Bob and write it to a file named `bob.csr`.

Q1.6. Use the `openssl ca` command to sign Bob's certificate by the ACME certificate and write it to a file named `BOB.pem`.