KTH Computer Science
and Communication

ACME Scandinavia

Secure IT environment

Final Report: Requirements and System Design

Group 1

ADAM JOHANSSON	adajoh@kth.se
ERIK NORELL	eriknore@kth.se
IGNACIO QUEZADA CRUZ	piqc@kth.se
MICKE SÖDERQVIST	mickeso@kth.se

Royal Institute of Technology
EP2520 - Building Networked Systems Security (BNSS)
Instructor: Panos Papadimitratos (papadim@kth.se)
Main Teaching Assistant: Stylianos Gisdakis (gisdakis@kth.se)

March 26, 2015

Requirements

Two separate networks are required, one in Stockholm and one in London. These are to be connected over the insecure internet in a secure manner, meaning no information is exposed to any outsiders, no information is lost and it is not possible for any outsider to insert false information.

Wired connection to the network at each office is required and only accessible using company desktop workstations. These can not be used by unauthorized personnel.

Wireless connections to the networks at each office is required as well. Wireless connection will only be possible using company supplied laptops and company supplied mobile phones with authentication of users using digital identities and certificates. Employees from Stockholm visiting in London shall be able to access the Stockholm network using their company supplied laptops on the London wireless network.

It shall be possible to transfer files between two company supplied mobile phones in a secure manner. It shall not be possible to transfer a file from an employee to a non-employee.

Specifics to Stockholm office

There shall be an intrusion detection system (IDS) deployed in the Stockholm network, monitoring traffic in the network and raising an alarm if an intrusion is detected as well as logging information about the intrusion.

A web server with access to company data shall be accessible to company employees from outside the network (e.g. from home with a private computer), only by using two factor authentication. This authentication must be performed using an application on the company supplied mobile phone. All traffic to and from this server shall be logged.

Finally, the complete security solution presented shall be scalable to accommodate for further expansions of the company's IT infrastructure.

eo

Design

The heart of our secure solution is an ACME public key infrastructure (PKI), meaning most of the security is based around the PKI. Many of the services involves a client/server relation where both server and client are authenticated using ACME certificates before data is encrypted using SSL/TLS. The PKI is based on a Root Certificate Authority (CA) using OpenSSL. The Root CA credentials should be kept secure, preferably never used on a computer connected to the internet and stored on a USB-key or similar - if these are compromised the whole network should be considered insecure! A Signing CA with a certificate signed by the Root CA is used to issue certificates to all employees, company devices, servers, etc.

All certificates are issued with a 2048-bit RSA key pair and SHA256 hashed message authentication code (HMAC). Certificate Revocation Lists (CRL) are regenerated daily and distributed where needed, if a certificate is revoked the CRL is regenerated and distributed immediately.

The main advantage of using a PKI instead of e.g. a Kerberos based system is that services do not depend on a central machine being online and available to the users. A PKI avoids a single point of failure, if e.g. the authentication server becomes unavailable the whole network is unavailable. A PKI on the other hand is not dependent on any machine. If all certificates are distributed and the Root certificate is available and trusted, then everything works. Another advantage with a PKI is that it's not dependent on passwords (other than when exporting public and private keys). This means greater transparency for users and avoids most of the risks with a password based system (choice of bad passwords, dicitionary attacks, etc.).

External security

All traffic between the London and Stockholm network is protected through a Virtual Private Network (VPN) in tunnel mode (using TLS/SSL), the software used is OpenVPN. Both London (client) and Stockholm (server) entry points are authenticated using ACME certificates.

Employees outside of the ACME network can access sensitive data on an Apache webserver. The webserver is only accessible using HTTPS, i.e. encrypted with SSL/TLS using ACME PKI credentials. It is configured to only allow strong ciphers and TLSv1.0 protocol or higher. All traffic is logged and saved daily.

To access the secure data on the webserver an employee is authenticated using two or three factors. Our suggestion is that an employee is required to install his or her ACME certificate on the device used to access the server, the requirement of certificates can be considered a strong factor. The second factor is one time 6-digit pincodes (OTP) using Google Authenticator which are regenerated every 30 seconds. An optional third

factor is a personal pincode or password used in conjunction with the OTP. After 5 bad login attempts a user account is locked. If using personal certificates are considered too impractical a two factor authentication is possible using only personal passwords and OTPs.

Internal Security

It is assumed that the internal network is physically secure from outsiders, i.e. that an outsider does not have physical access to the offices of ACME. In other words there is no need to encrypt the internal traffic. We also assume that proper measures are taken at ACME offices to prevent unauthorized personnel from accessing the workstations.

A Firewall is placed at the entry point of each network using stateful packet filters. This will prevent unauthorized outsiders on the internet to get access to the internal network. It will strictly allow communication to those services we use, with a default DROP policy, we wrote every rule with as few wildcards as possible. The firewall is implemented using iptables and will also be used for routing.

An IDS is installed at the Stockholm office monitoring all traffic going in and out of the network. The software used is Snort. The IDS can be configured to send a summary to an administrator daily automatically by email. It *does need* proper and time consuming configuration of the rules to work optimally, this should be performed as soon as the network is deployed. The IDS is currently configured to update its rules automatically on a daily basis and can be administrated using a web based GUI.

The wireless network is available to company devices using ACME certificates. The network is secured by WPA2 Enterprise using EAP-TLS protocol, meaning all devices and an internal RADIUS server are authenticated by ACME certificates and all communication is encrypted. The software used for the RADIUS server is FreeRADIUS.

File sharing service

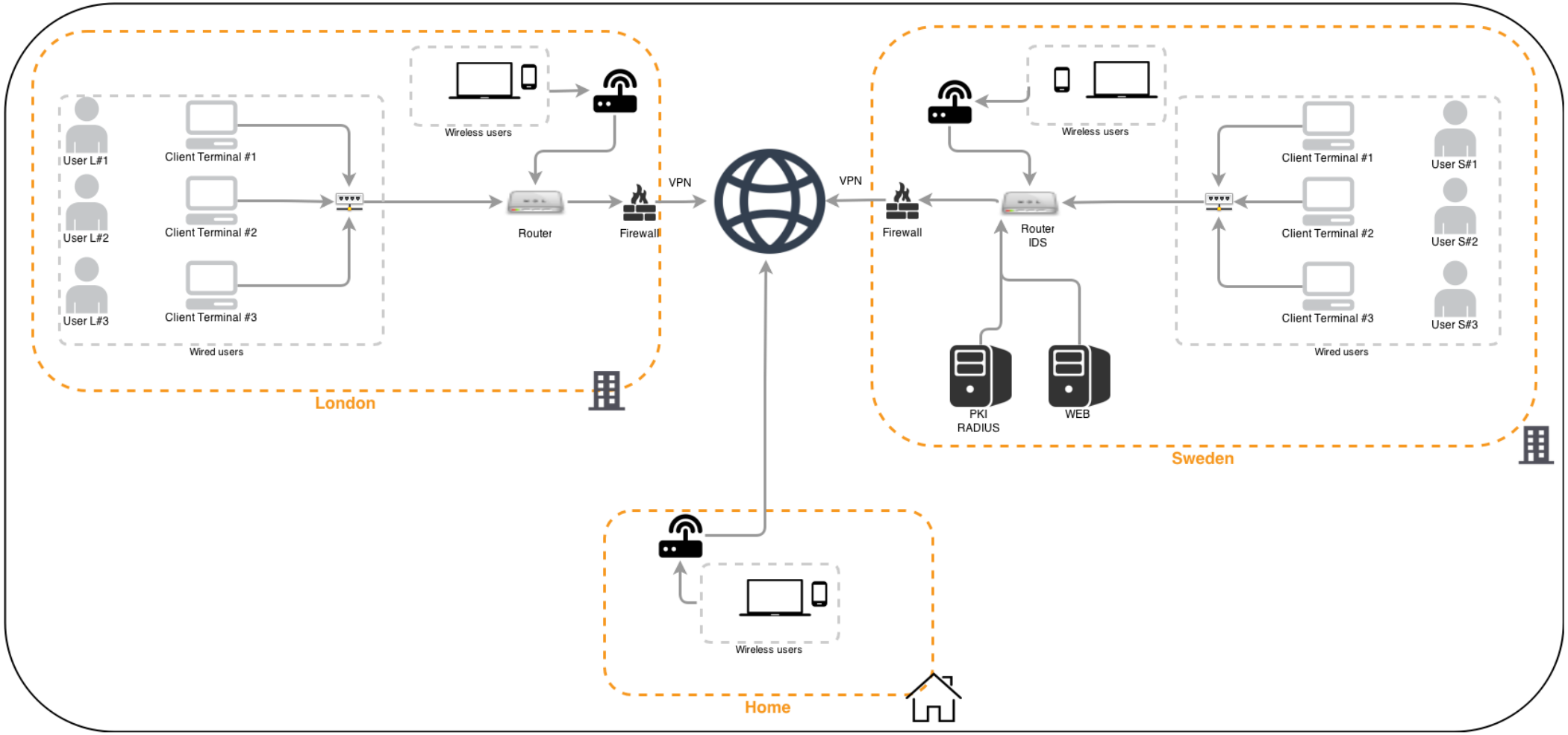
Regarding the file sharing service, we have assumed that we are *not required* to restrict company phones from unauthorized file sharing with other devices. In other words we are only supplying an application for which secure file sharing is possible. The first case would mean restricting USB transfer, use of SD-cards, use of other applications, etc. Which might not even be possible using commercially available mobile phone operating systems.

An Android application has been developed with the purpose of transferring files securely between employees' ACME phones. All the encryption and decryption is performed automatically and totally transparent to the user. A small API is hosted on the webserver which is used in the transfer of files and files are temporarily stored on the webserver.

Employee PKI credentials are required on the phone as only the intended recipient should be able to access the file once transferred. A downside to this requirement is that in the event of a phone getting lost or stolen the employee credentials must be revoked and new ones issued and reinstalled, this includes all other devices used by the employee, if any. The upside is stronger security as files stored on the server cannot be accessed by others than the recipient, e.g. if the server would get compromised or server administrators by accident access files.

The communication between the application and the webserver is secured over HTTPS where both parties are authenticated using ACME certificates. An employee can only send files to other employees as the possible recipients are supplied from the webserver using an automatically updated database of current employees. Once a file and recipient has been chosen a symmetric 256-bit key is randomly generated and used to encrypt the file using AES-CBC encryption. The symmetric key is then encrypted using the recipient's public key from the ACME credentials, meaning only the recipient having the matching private key can decrypt the file. The encrypted symmetric key, IV (for CBC) and file are sent to the server and stored there awaiting delivery.

Each time an employee starts the application it checks if there are any files to retrieve from the webserver. If there is then the file, IV and paired symmetric key are delivered and decrypted so that the employee can access the file.


User L#1
User L#2
User L#3
Client Terminal #1
Client Terminal #2
Client Terminal #3
Wired users
Wireless users
Router
Firewall
VPN
VPN
London
Firewall
Router
IDS
Wireless users
Client Terminal #1
Client Terminal #2
Client Terminal #3
Wired users
User S#1
User S#2
User S#3
PKI
RADIUS
WEB
Sweden
Wireless users
Home

Appendix

There are some implementation specific details concerning the network and application we thought could be good to know. Also some shortcomings to our design which we didn't feel was necessary as this is more a proof of concept, still we are aware of them.

The VMs contain

VM1: iptables, OpenVPN, Snort (incl GUI through Apache server)

VM2: iptables, OpenVPN

VM3: Apache web server, ACME PKI Signing CA

VM4: RADIUS server, ACME PKI Root CA

Web server

The Web server is currently stateless. We have not enabled cookies on the website as it is only a proof of concept. If we would have, we would have enabled SecureCookies on the Apache server preventing the client from sending cookies in an unencrypted channel - i.e. avoiding a possible type of redirection attack.

The secure area of the web server could be used as a convenient secure area for the whole internal network. The secure area can be setup to only authenticate users from outside the internal network with Google Authenticator. However, an issue of revoked users and the wireless network (see below) must be resolved first.

IDS - Snort

The IDS is mostly setup as a proof of concept, meaning we have everything set up but not configured fully (web interface is ACIDBASE, or snortreport for a more summarized version and oinkmaster is used for automatic updates). The rules of which the IDS uses should be selected more carefully and possibly some custom rules should be added. We have tools (snort-stat) to produce a short summary report which could be sent by email daily to an administrator.

PKI, CAs and authentication

The Root CA and Signing CA are symbolically stored on different VMs, so if the signing CA gets compromised we can revoke it meaning all issued certificates become useless.

We wrote scripts for creating and revoking users, so input of a new user is automatically inserted into the database when a user is created and CRLs are immediatly distributed to the other servers which are also restarted where necessary when a certificate is revoked. However the error handling of these scripts is not perfect, e.g. entering the wrong password for the sigining CA when creating a new certificate results in files being created but cannot be used (although insertion in database only happens if user is added sucessfully).

The OTP account is created in a separate process. We have a script which, given a user name, generates a paired hash and QR-code to use with Google Authenticator. The hash and user name is manually entered into a list of users. The phone to be used as authenticator scans the QR-code. If someone gets hold of the QR-code he or she can authenticate using the paired username, so the QR-code should not be saved once it has been scanned.

xPKI

With the use of KTH VeSPA, we managed to get certificates signed by their CA using a custom made script. It allows the user to use the certificate to make use of the VESPA services (have not gone through them as the site with the information was incomplete). The same way, the certificates issued by VeSPA could be used to make use of our own services (could be as we did not really add the VeSPA certificates to ours).

Wireless network

We have found that the D-Link AP caches a user connection, meaning if a user is revoked he or she is still able to connect a certain time after revocation. In this situation the AP doesn't check credentials - however when credentials are checked the user is denied. We believe we can setup FreeRADIUS to demand that the AP doesn't cache sessions at all. We only found this out at the end so we didn't have time to investigate this further.

Secure File Transfer

The way it works now, the following data is stored in a MySQL database: A list of all common names and corresponding Public Keys from the certificates. A table with common name, file name, the file, encrypted symmetric key and IV. To better secure the data in case the database is compromised, at least file name should also be encrypted with the recipients Public Key. Privacy can be strengthened by encrypting Common Names, but for ACME it might be more worth to have better accountability. That can be achieved for example by forcing the sender to sign a hash of the file sent. Having the file name in plaintext might also give an attacker patterns (known data) that help him break the symmetric key.

The app also could be more user-friendly. Right now, in case it cannot contact the server, it crashes. When files are downloaded, all feedback the user gets is *file downloaded*, which is not very helpful.

The client certificate is now stored in Androids KeyStore. It is possible to instead save it in an app-specific KeyStore, but this way we can use the same certificate for other uses.

Another thing that we do not show in the demo but that a real life solution should have is a separate certificate for the cell phones. That way the cell phone certificate can be separately revoked upon theft or loss.

The API can better be strengthened against SQL injections and also possible CSRF/XSS attacks.

API

To enable the Secure File Transfer to talk with the server we have implemented an API which (after SSL authentication) enables the Android application to use the webserver as a backbone. The API has four tasks:

1. Get users - Names and Public keys
2. Get information about file(s) sent to the user and the corresponding encrypted symmetric key
3. Get the encrypted file
4. Send file to another user

These are called by setting a request parameter ”req_status” to the corresponding number (1-4), and depending on which task a number of additional parameters might have to be set. By doing this we could at a later stage have other applications connected to the same functionality.

This rather small API uses a MySQL database to store users and transfers in transit and this is accessed through PHP using MySQLi connection/queries. As HTTPS/SSL connections are handled in the Apache webserver rather than in PHP there is not to much authentication going on in the API, but using some of the $_SERVER variables we check that the SSL connection has been verified successfully and that the certificate is issued by our CA (but no one should be able to reach tha API if this wasn't correct). For the SQL queries we use prepared statements to prevent SQL injection.