Author: rupp,scd
Project:Security.MS
Date: 09/11/13

Security.MS specifications

Table of Contents

1.introduction....................1
2.Description....................2
3.Services....................2
  3.1.Interface....................2
  3.2.VOIP....................3
    a)sip....................4
    b)WebRtc....................4
  3.3.Chat....................5
  3.4.Email....................6
4.Cryptographic Protocol....................7
  4.1.Key Exchange....................8
  4.2.Sessions keys....................11
5.Front-end....................12
6.Back-end....................13
  6.1.Session Secure Channel management....................13
  6.2.User authentication....................14
7.Database....................14

Illustration Index

Illustration 1: Tabbed Interface for services....................3
Illustration 2: VOIP Unavailable when WebRtC not detectedc....................4
Illustration 3: sample WebRTC Voip/Video communication....................5
Illustration 4: Ajax Chat....................6
Illustration 5: Cubusmail preferences....................7
Illustration 6: Communication Channel computer1-computer2....................8
Illustration 7: Key Exchange and Session Key computation for Security.MS....................9
Illustration 8: PGP Certificate Submission....................10
Illustration 9: PGP Certificate Information....................11
Illustration 10: Status Bar....................12
Illustration 11: Front-end components....................13

1. introduction

This document describes the technique and design used to develop a secure service for chat, voip and emails., security.MS

2. Description

Security.MS is a 100% web service that do not require any plugin or third party software installation which provides cryptographic services to ensure confidentiality and integrity of chat. Voip and emails.

The service works with most standard modern browsers and only require javascript and HTML5 (no java/flash/silverlight etc...)

It is based on several components:

- ***The Client / Front-end which is pure Javascript and HTML5;***
- ***The backend which is an application server connected to a database;***
- ***The HSM, which goal is to store keys;***

The key aspect of security.MS is to provide a secure channel inside a browser for communications between two computers. This secure channel is setup using GNUPG key exchange protocol (PGP Key format). This keys will generate session keys that will be used to cipher communications between the computers and all this will be done entirely in javascript/HTML5.The Javascript code will run in a pop-up windows (or any child windows spawn by the parent windows).

The main goal of this system is to prevent interceptions and man-in-the-middle attack so that it must be resistant to several known techniques such as:

- ***SSL Striping***
- ***Man-in-the-Browser attack***
- ***Proxy SSL Server***
- ***Man-in-the-Middle (general) attack***
- ***Xss***

What must be prevented here is the interception of the communication by an unauthorized third-party so that only the two computers can decipher and read the content of the communication.

3. Services

3.1. Interface

The interface presents the services as tabbed pages: Chat/Voip/Email

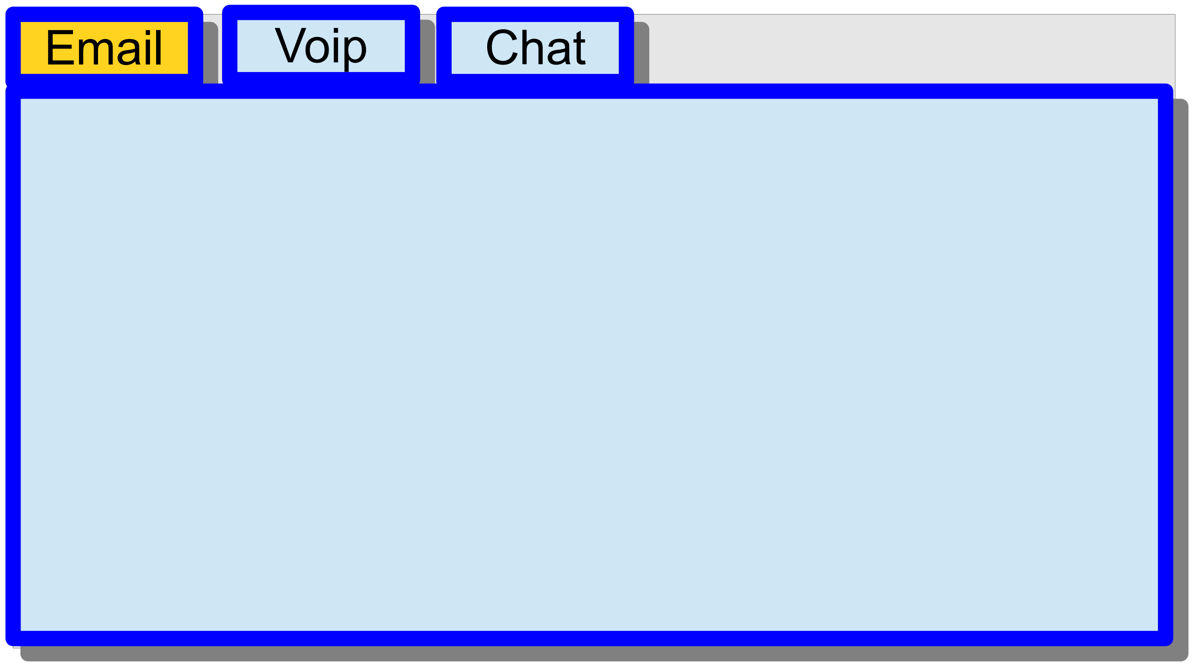


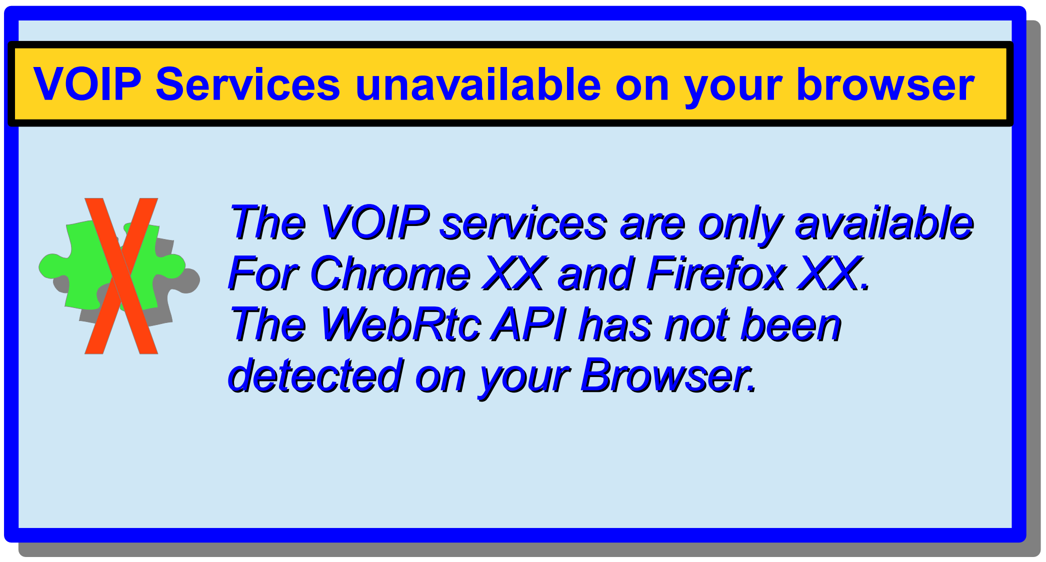
Illustration 1: Tabbed Interface for services

3.2. VOIP

The VOIP services are based on SIP services and WebRtc services.

Since Sip API is also based on WebRTC- this imply the presence of the WebRTC API in the browser. Currently only Chrome and Firefox are supporting this API while in the near future it should be implemented on all other browsers (IE,Opera, …)

If the user is not using Chrome or Firefox,the VOIP services won't be available.



VOIP Services unavailable on your browser

The VOIP services are only available
For Chrome XX and Firefox XX.
The WebRtc API has not been
detected on your Browser.

Illustration 2: VOIP Unavailable when WebRtC not detected

a) sip

The VOIP sip client is based on a javascript/HTML5 library such as:

- ***http://sipml5.org/*** ***(SimMl)***
- ***https://code.google.com/p/sip-js/*** ***(Sip-js)***

b) WebRtc

See [1] for an introduction about WebRtc and Sip in HTML5.

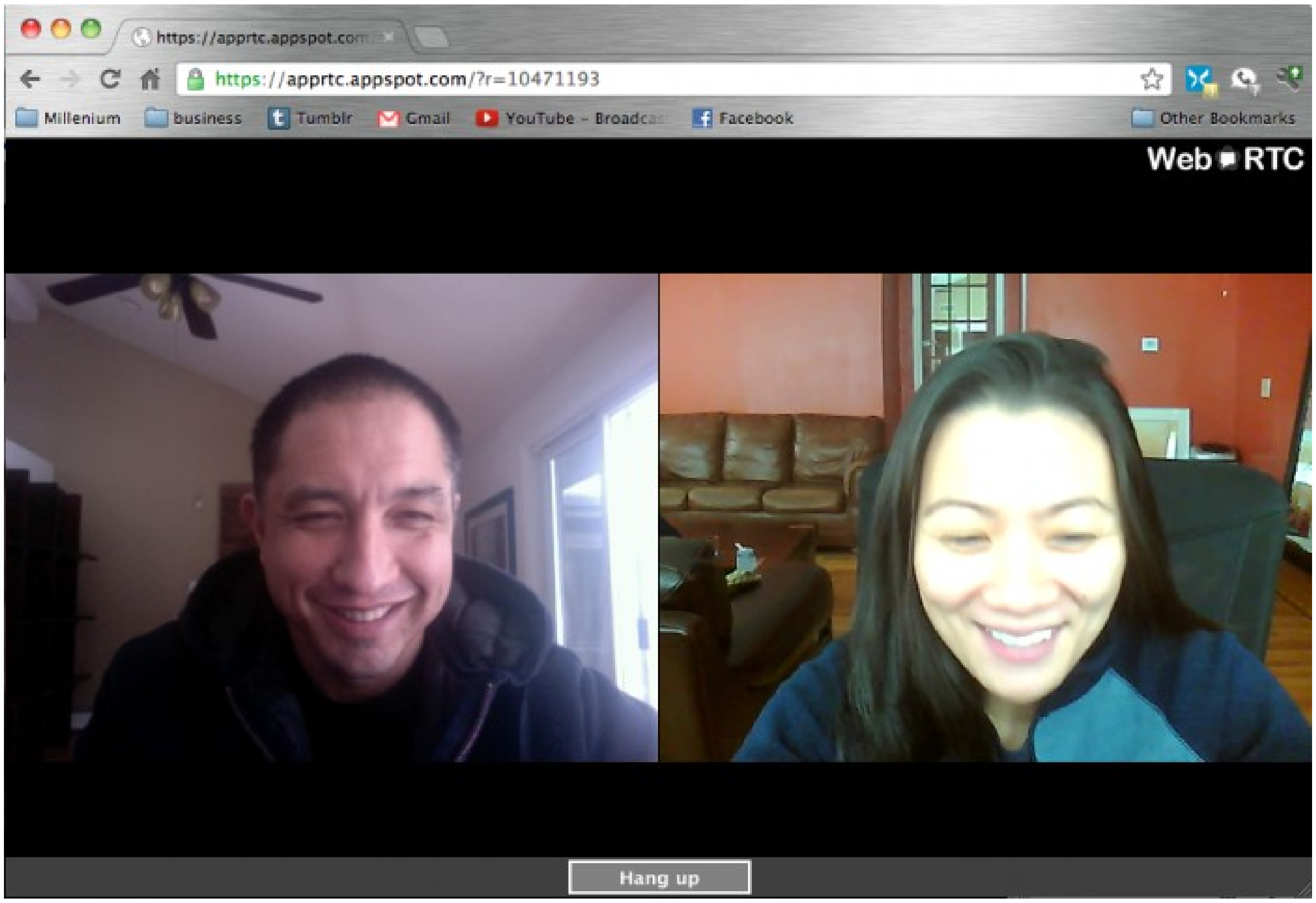


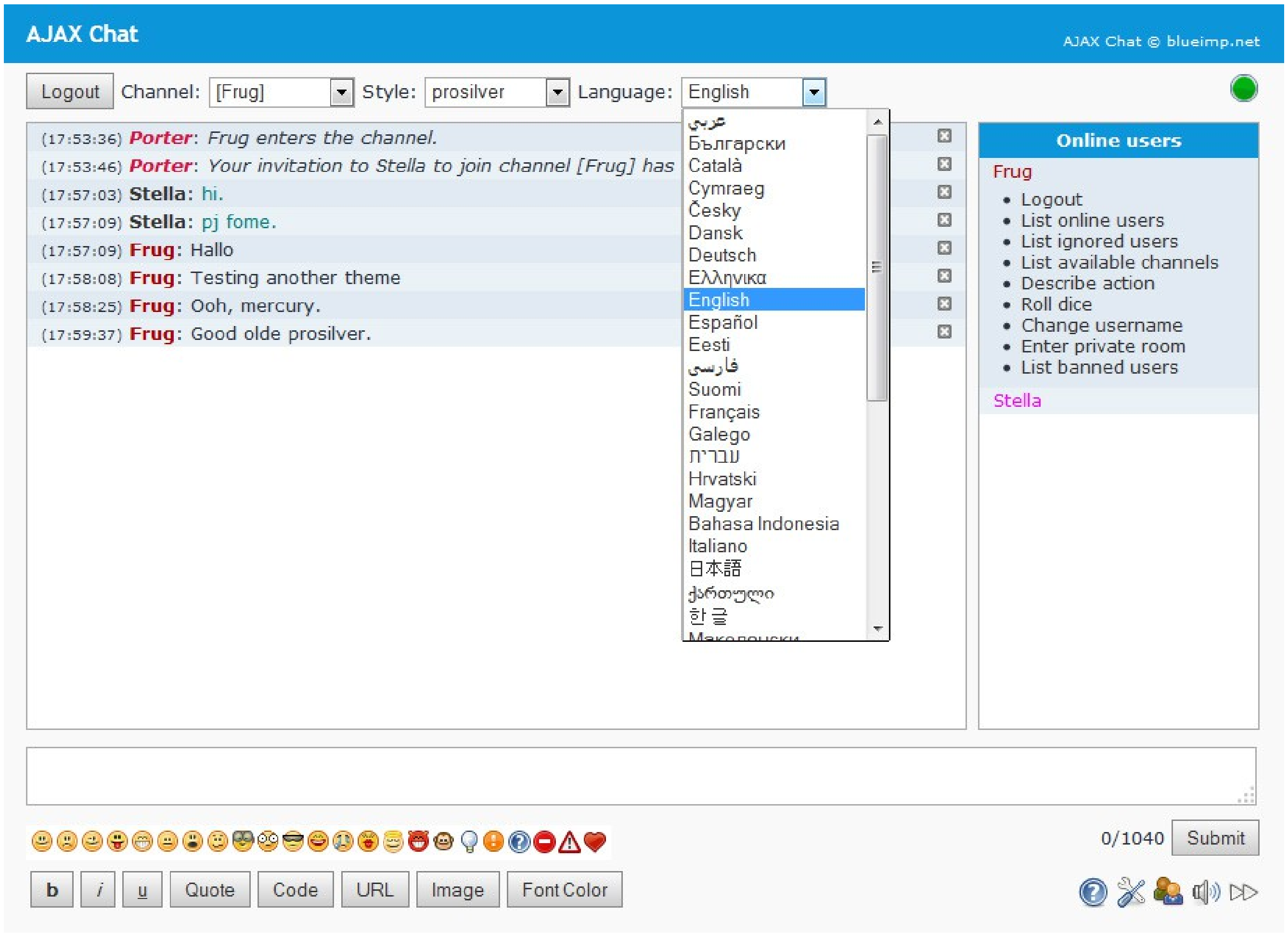
Illustration 3: sample WebRTC Voip/Video communication

WebRtc stands for 'Web Real-Time Communications', this is a new API that is coded in the browsers and that mainly allows to capture the video and sound input. WebRTC allows full VOIP communications without any plugins such as Java,Flash or Silverlight at all. The video and sound are totally fluid and high-quality (see [2])

3.3. Chat

The Chat is based on Ajax Chat (see http://frug.github.io/AJAX-Chat/)

This is a smooth, fluid powerful Ajax Chat with the main function of a modern chat system.

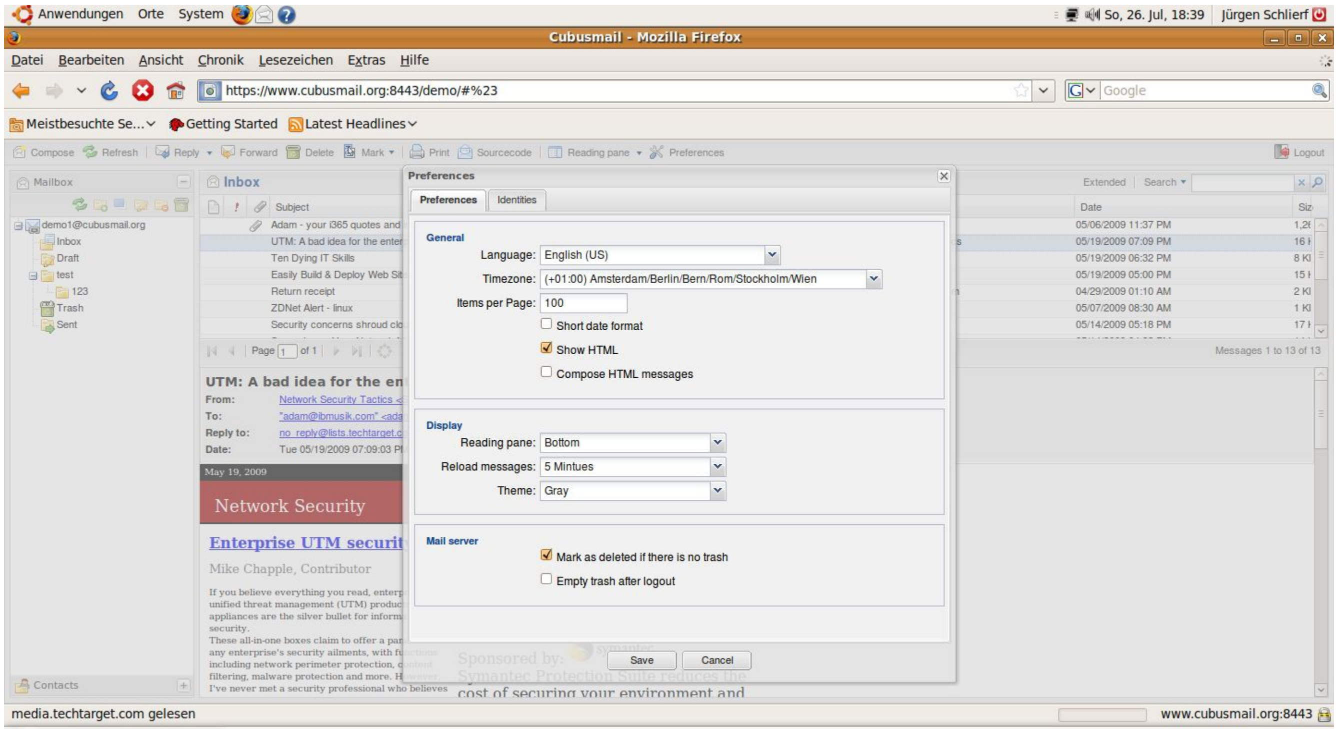


(17:53:36) *Porter*: *Frug enters the channel.*
(17:53:46) *Porter*: *Your invitation to Stella to join channel [Frug] has*
(17:57:03) **Stella**: hi.
(17:57:09) **Stella**: pj fome.
(17:57:09) **Frug**: Hallo
(17:58:08) **Frug**: Testing another theme
(17:58:25) **Frug**: Ooh, mercury.
(17:59:37) **Frug**: Good olde prosilver.

Illustration 4: Ajax Chat

3.4. Email

The email based on cubusmail (https://code.google.com/p/cubusmail/)



Preferences

Preferences | Identities

General

Language: English (US)

Timezone: (+01:00) Amsterdam/Berlin/Bern/Rom/Stockholm/Wien

Items per Page: 100

☐ Short date format

☑ Show HTML

☐ Compose HTML messages

Display

Reading pane: Bottom

Reload messages: 5 Mintues

Theme: Gray

Mail server

☑ Mark as deleted if there is no trash

☐ Empty trash after logout

Save | Cancel

Illustration 5: Cubusmail preferences

4. Cryptographic Protocol

The cryptographic protocol is based on PGP key exchange system.

We create a local javascript proxy on the client that cipher and decipher on the fly the communications with the server. Between two proxies the data are ciphered.

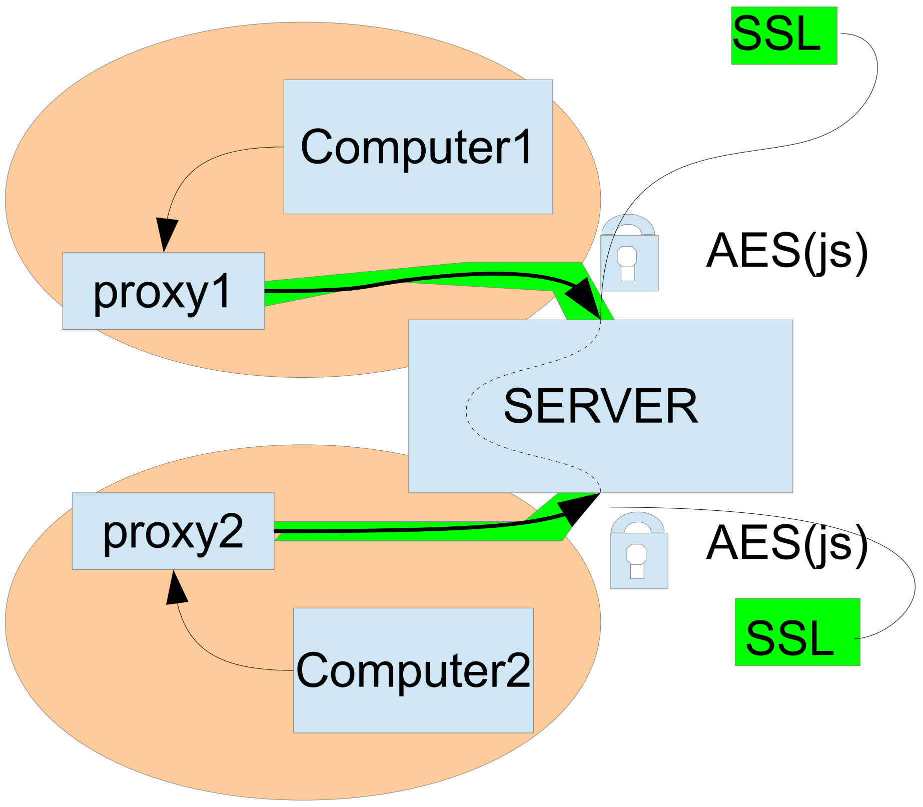


Illustration 6: Communication Channel computer1-computer2

Javascript based encryption will use the library crypto-js. (https://code.google.com/p/crypto-js/)

Concretely internet data between the two proxies are ciphered/deciphered on-the-fly by AES using AES session keys. It has to be noted that communications are secured by an SSL channel.

The server connects the two computers and relay the ciphered traffic. It has no knowledge of the session keys (which are sent ciphered by PGP keys computer-to-computer)

It has to be noticed that only a symmetric algorithm in javascript – like AES - can be used for traffic on-the-fly encryption/decryption , an assymetric algorithm like RSA would be far too slow tio be of any use.

4.1. Key Exchange

Keys are exchanged by mean of the browser.The key exchange protocol is PGP.

Validity and identity of the keys is ensure by PGP PKI system (which not CA based).

The browser will display the informations related to the key and will ask the other user of it wants to proceed.

PGP interest is that it is combining symmetric and assymetric encryption in the following way:

- *asymetric encryption is used to exchange the session keys and for data communication issues*
- *symmetric encryption (which is 1000 time faster than asymetric encryption) is used ro cipher/decipher the data*

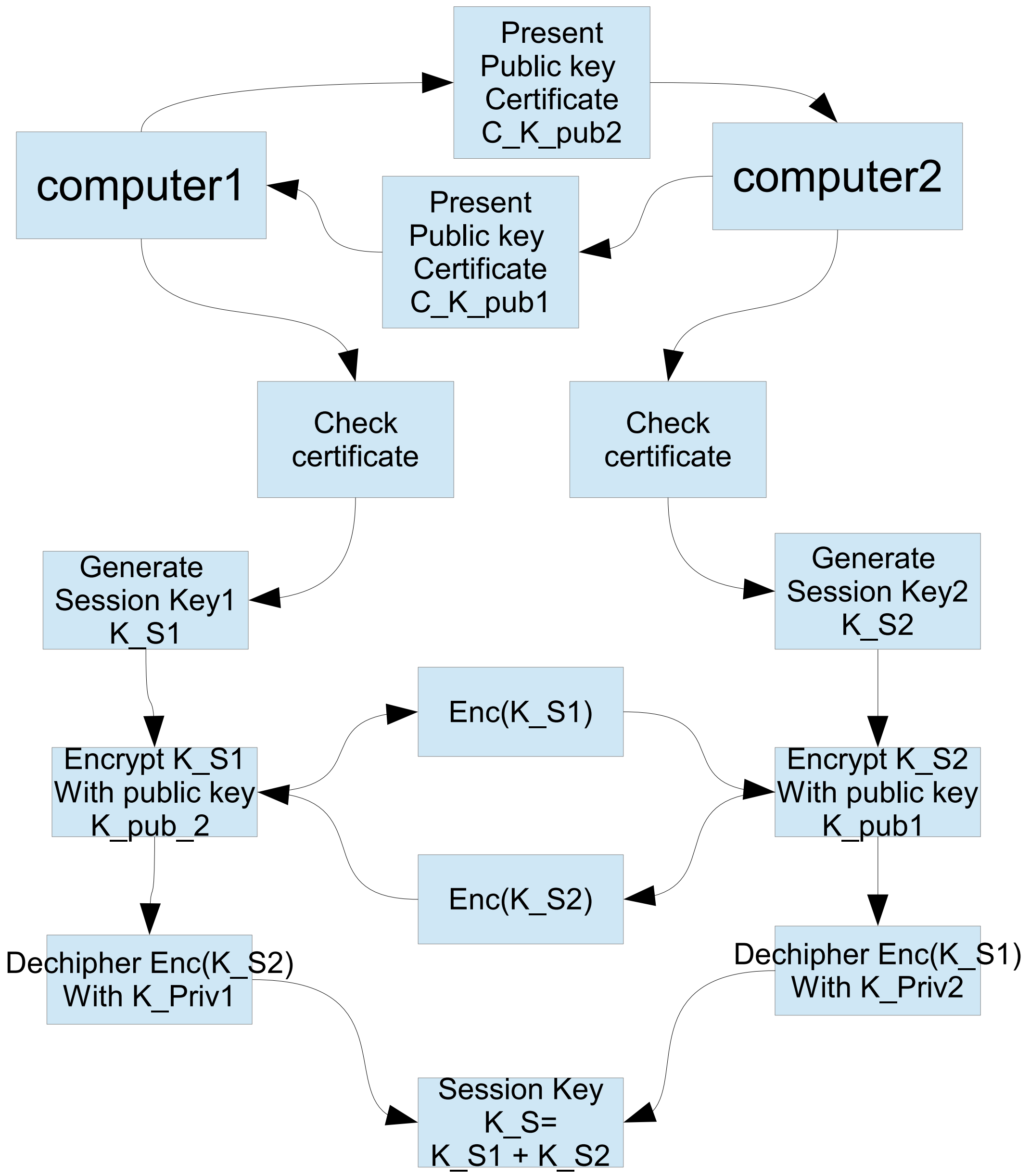

Present Public key Certificate C_K_pub2
computer1
computer2
Present Public key Certificate C_K_pub1
Check certificate
Check certificate
Generate Session Key1 K_S1
Generate Session Key2 K_S2
Enc(K_S1)
Encrypt K_S1 With public key K_pub_2
Encrypt K_S2 With public key K_pub1
Enc(K_S2)
Dechipher Enc(K_S2) With K_Priv1
Dechipher Enc(K_S1) With K_Priv2
Session Key K_S= K_S1 + K_S2

Illustration 7: Key Exchange and Session Key computation for Security.MS

The Key Exchange system is a variant of the PGP exchange system in the sense that both computers creates their part of the sesion key.

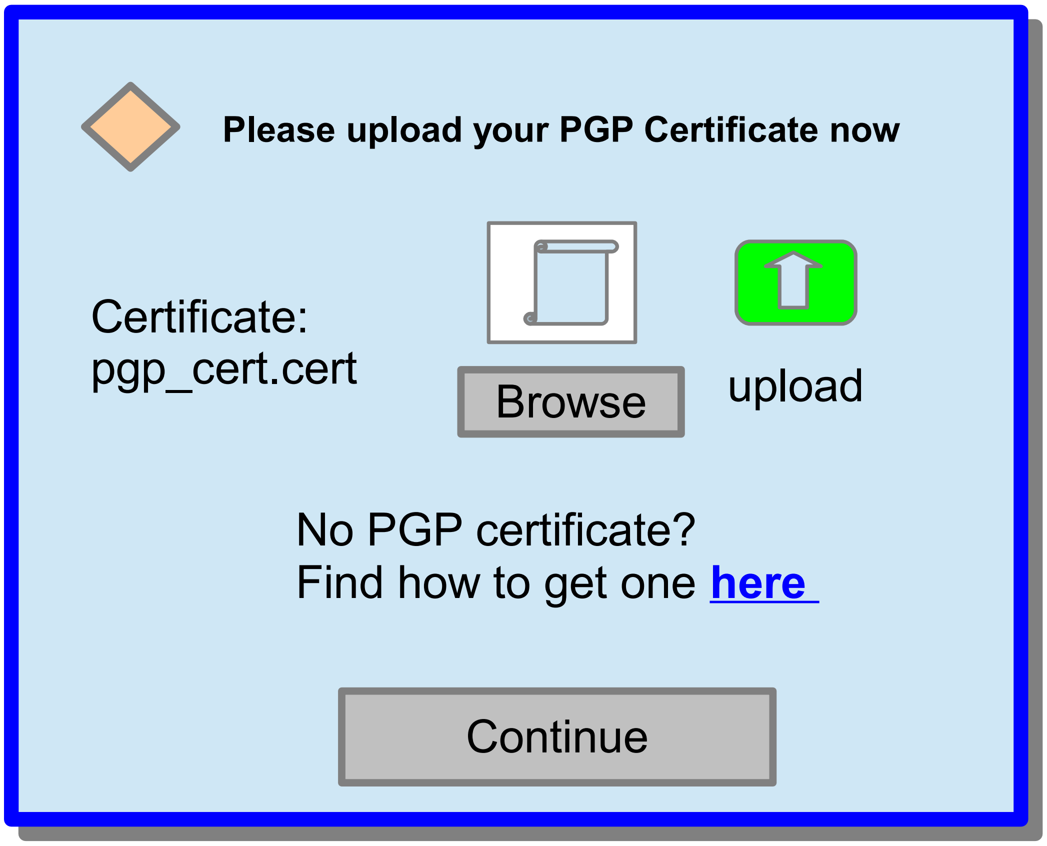
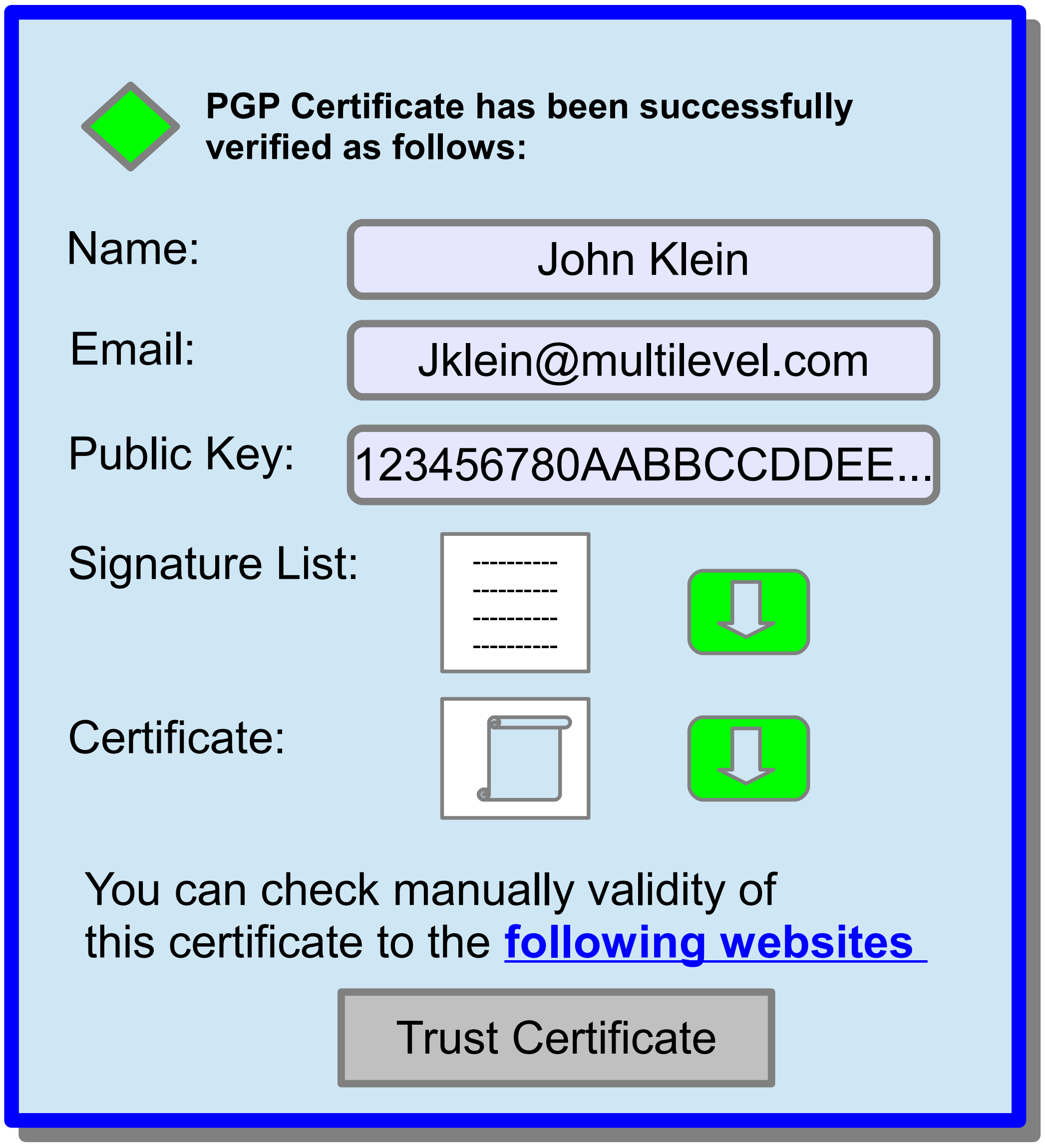


Illustration 8: PGP Certificate Submission

User is required to submit a PGP certificate that will be processed and authentificated by the website automatically.And the *information will be displayed to the two parties*.

- *Computer1 will have to accept (trust) Computer2 certificate*
- *Computer2 will have to accept (trust) Computer1 certificate*



PGP Certificate has been successfully verified as follows:

Name: John Klein

Email: Jklein@multilevel.com

Public Key: 123456780AABBCCDDEE...

Signature List:
Certificate:
You can check manually validity of this certificate to the following websites
Trust Certificate

Illustration 9: PGP Certificate Information

Once Computer1 and Computer2 trust each others, the session key computation can start and the secure channel is ready. K_S=K_S1+K_S2 is used for AES encryption/decryption on both sides.

The State of the secure channel is displayed on a top status bar.

4.2. Sessions keys

A mutual authentication scheme is created using PGP system: the secret AES session keys are ciphered by PGP keys and sent to the other computer.

The session key will be generated using Diffie Hellman/DSS scheme.

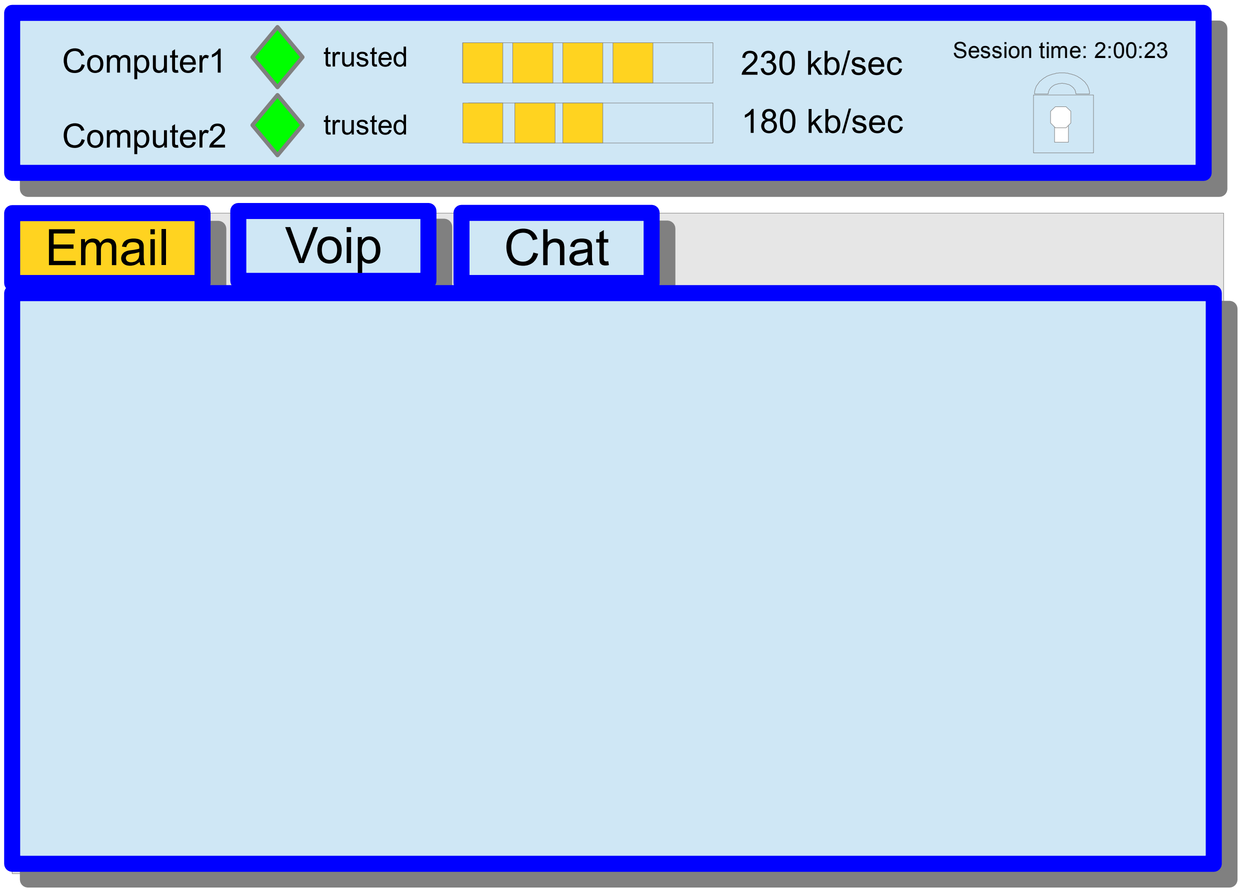


Illustration 10: Status Bar

5. Front-end

The front-end is entirely done in Javascript and HTML5 and based on the components described in the previous sections.

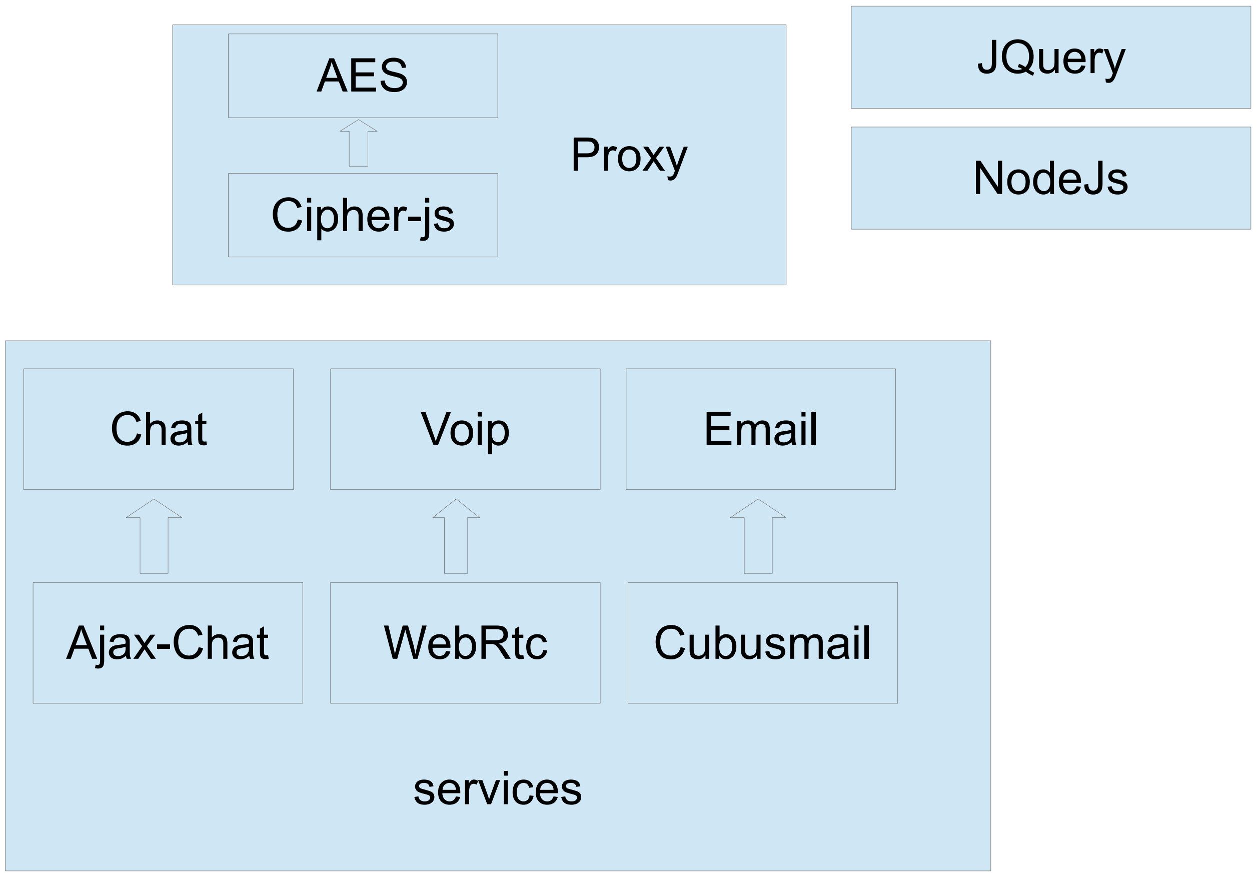


Illustration 11: Front-end components

6. Back-end

The Backend is a web server wich run php.The server must be able to connect two users togethers and relay the traffic for a given session, therefore the server is mostly a relay system since 90% of the intelligence is done in the front end.

6.1. Session Secure Channel management

The backend will create,run and terminate sessions/secure channels. A session will be defined by several parameters:

- *userid of the two parties*
- *start_time*
- *end_time*

Sessions are volatiles, they are not stored in the backend and are destroyed once stopped. Besides, the Sessions data are totally wiped on the server.

No cache is forced on the browser by using the following tags:

```
<meta http-equiv='cache-control' content='no-cache'>
<meta http-equiv='expires' content='0'>
<meta http-equiv='pragma' content='no-cache'>
```

Also features such as autofill, etc... are prevented on the browser side using several techniques.

There is no way anyway to clear the cache on the browser and to wipe data , so the users must be warned that some data may reside on the cache somehow.

6.2. User authentication

The website provided user creation and user password management. Before using the services the user must log-in on the server.

7. Database

The database contains the userid of the registered users. No data about PGP keys or secure sessions is stored.

[1]http://www.onsip.com/blog/2013/01/23/webrtc-sip-and-html5-brief-introduction

[2]https://sites.google.com/site/webrtc/demo