SAGE2™

Installation guide, V3

November 2017

Introduction

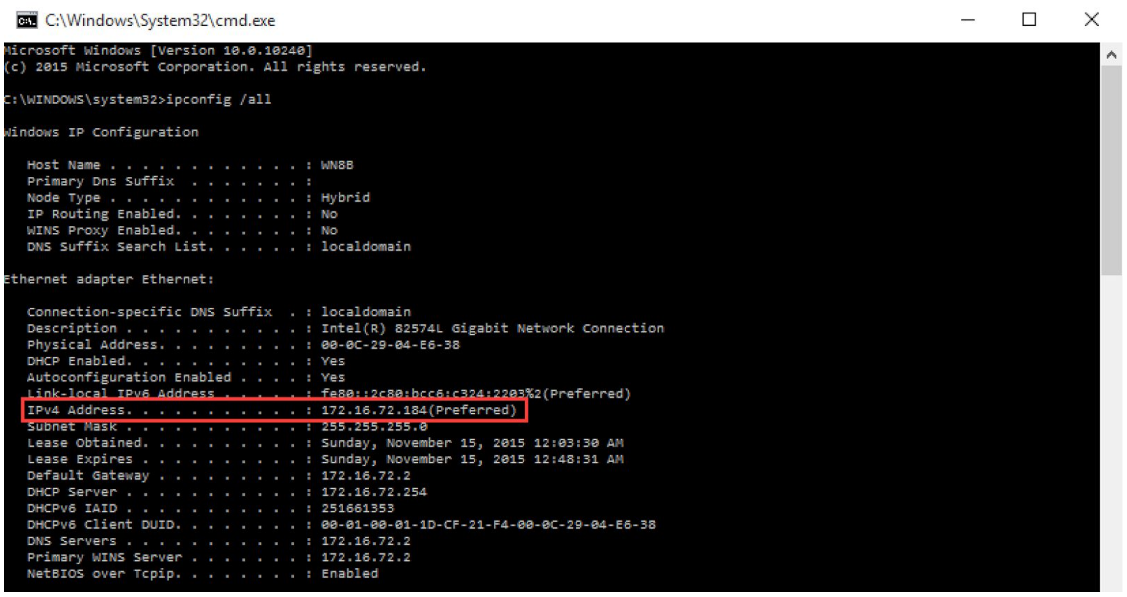
This document describes first the installation and setup for Windows (supported Windows 8, 8.1, and 10, all for 64-bit). Windows 10 is preferred (most tested). It should work similarly for Home, Pro, and Enterprise editions of Windows. The installation focuses on a single machine setup, which is highly recommended for a first setup.

Suggested configuration:

- **Windows machine** connected to a large monitor or TV
    - will run a launcher service to control the SAGE2 setup,
    - will run the SAGE2 server, one SAGE2 display client, and
- **Laptop**
    - will connect to the launcher and the SAGE2 server
    - will run one SAGE2 client interface

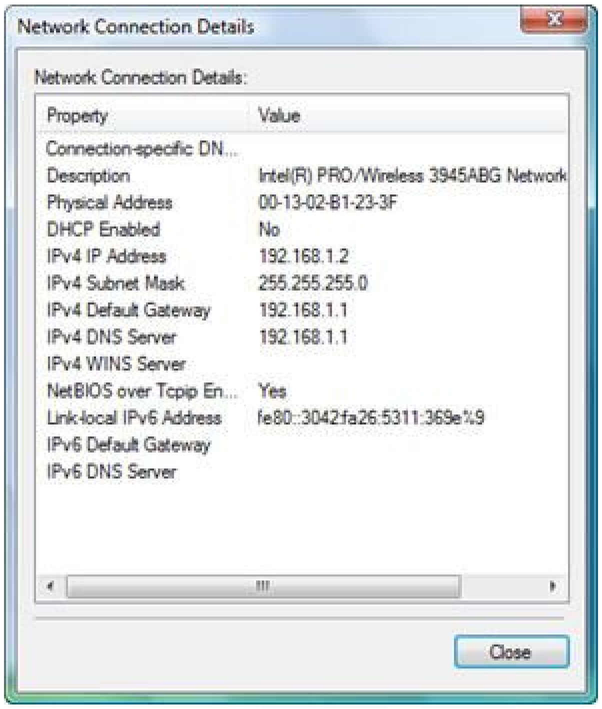
Prerequisites

- **Google Chrome 64-bit** should be installed
    - See: https://www.google.com/chrome/browser/desktop/
    - Make sure to select the 64-bit version (you might have to select ‘Download Chrome for another platform’).
- Knowing the **IP address (xxx.xxx.xxx.xxx) or hostname (myserver.test.com)** of your SAGE2 machine
    - to find how the IP address, either:
        - open a command terminal by running ‘cmd’
        - run: *ipconfig /all*
        - find your IPv4 address

- 

  ```
  C:\Windows\System32\cmd.exe
  Microsoft Windows [Version 10.0.10240]
  (c) 2015 Microsoft Corporation. All rights reserved.

  C:\WINDOWS\system32>ipconfig /all

  Windows IP Configuration

     Host Name . . . . . . . . . . . . : WN8B
     Primary Dns Suffix  . . . . . . . :
     Node Type . . . . . . . . . . . . : Hybrid
     IP Routing Enabled. . . . . . . . : No
     WINS Proxy Enabled. . . . . . . . : No
     DNS Suffix Search List. . . . . . : localdomain

  Ethernet adapter Ethernet:

     Connection-specific DNS Suffix  . : localdomain
     Description . . . . . . . . . . . : Intel(R) 82574L Gigabit Network Connection
     Physical Address. . . . . . . . . : 00-0C-29-04-E6-38
     DHCP Enabled. . . . . . . . . . . : Yes
     Autoconfiguration Enabled . . . . : Yes
     Link-local IPv6 Address . . . . . : fe80::2c80:bcc6:c324:2203%2(Preferred)
     IPv4 Address. . . . . . . . . . . : 172.16.72.184(Preferred)
     Subnet Mask . . . . . . . . . . . : 255.255.255.0
     Lease Obtained. . . . . . . . . . : Sunday, November 15, 2015 12:03:30 AM
     Lease Expires . . . . . . . . . . : Sunday, November 15, 2015 12:48:31 AM
     Default Gateway . . . . . . . . . : 172.16.72.2
     DHCP Server . . . . . . . . . . . : 172.16.72.254
     DHCPv6 IAID . . . . . . . . . . . : 251661353
     DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-1D-CF-21-F4-00-0C-29-04-E6-38
     DNS Servers . . . . . . . . . . . : 172.16.72.2
     Primary WINS Server . . . . . . . : 172.16.72.2
     NetBIOS over Tcpip. . . . . . . . : Enabled
  ```

- or
  - Open Network Connections by clicking the Start button Picture of the Start button, and then clicking Control Panel. In the search box, type adapter, and then, under Network and Sharing Center, click View network connections.
  - Select an active network connection, and then, in the toolbar, click View status of this connection. (You might need to click the chevron icon to find this command.)
  - Click Details.
  - Your computer's IP address appears in the Value column, next to IPv4 Address:

- 

Network Connection Details

Network Connection Details:

Property	Value
Connection-specific DN...	
Description	Intel(R) PRO/Wireless 3945ABG Network
Physical Address	00-13-02-B1-23-3F
DHCP Enabled	No
IPv4 IP Address	192.168.1.2
IPv4 Subnet Mask	255.255.255.0
IPv4 Default Gateway	192.168.1.1
IPv4 DNS Server	192.168.1.1
IPv4 WINS Server	
NetBIOS over Tcpip En...	Yes
Link-local IPv6 Address	fe80::3042:fa26:5311:369e%9
IPv6 Default Gateway	
IPv6 DNS Server	

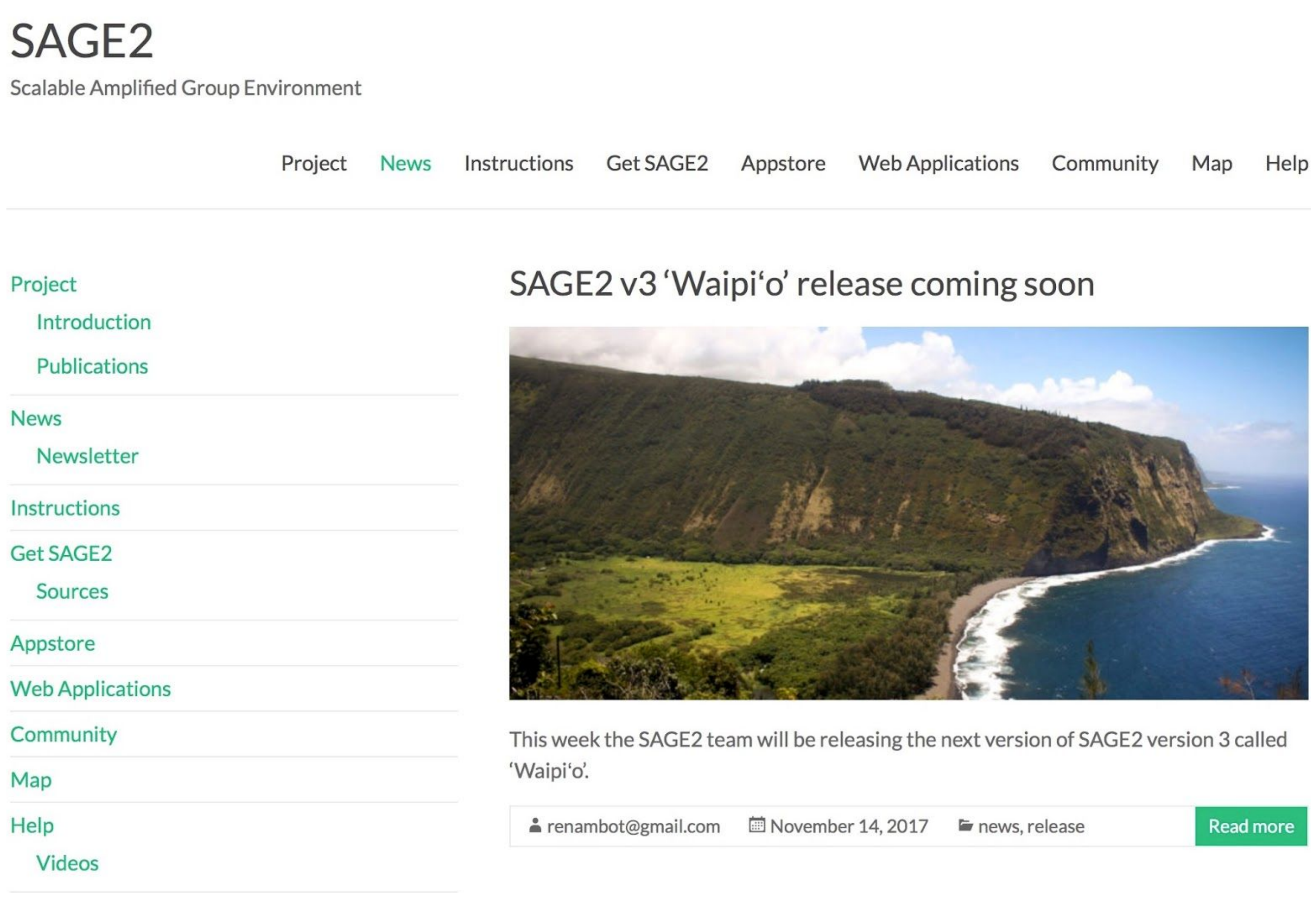
Remark

You can install SAGE2 in any folder, however, If this is the first time the computer is launching SAGE2, a folder called SAGE2_Media will be created within your Documents folder (for the current user). Once installed, Chrome will open the launcher and initial configuration setup page for SAGE2.

SAGE2 v3 is currently built with **Electron 1.7.9** and **Node.js v8.9.1**.

Binary Installation

Get SAGE2


SAGE2

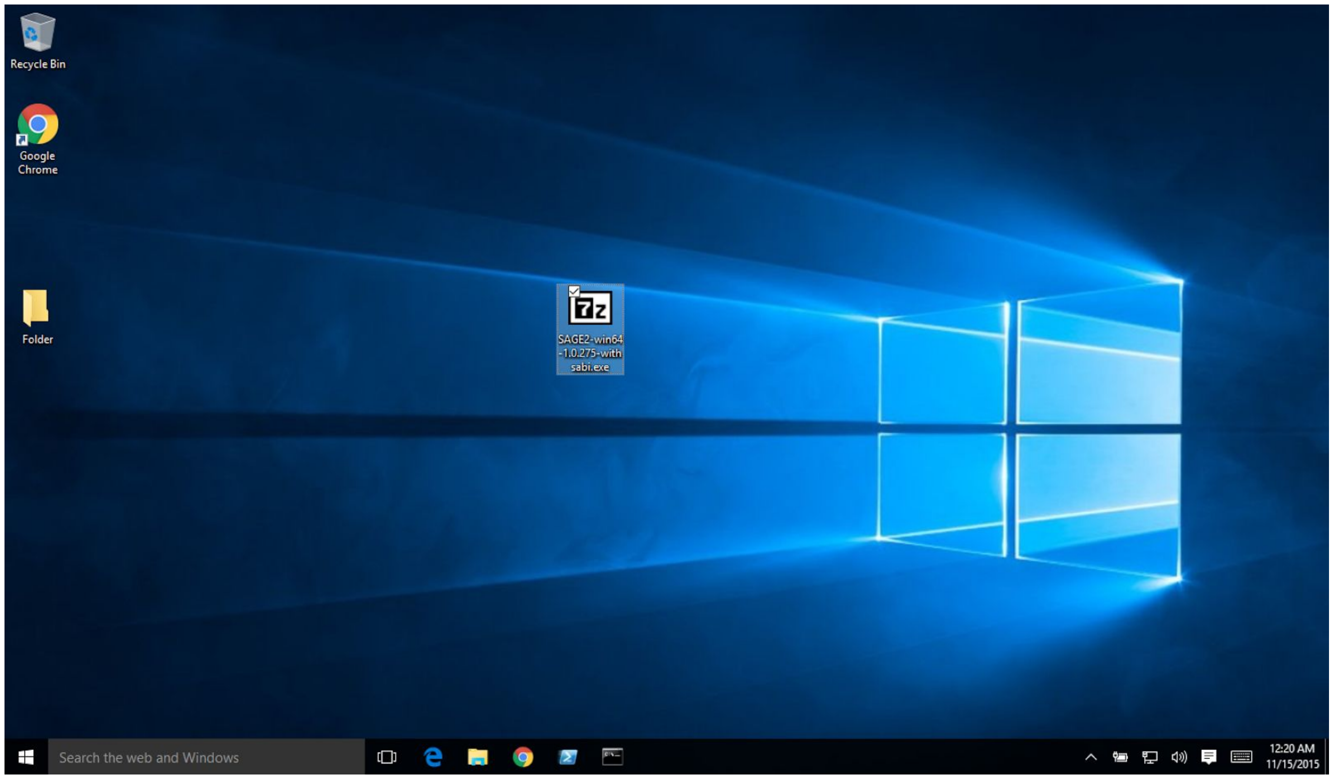
Scalable Amplified Group Environment

SAGE2 v3 ‘Waipi‘o’ release coming soon

This week the SAGE2 team will be releasing the next version of SAGE2 version 3 called ‘Waipi‘o’.

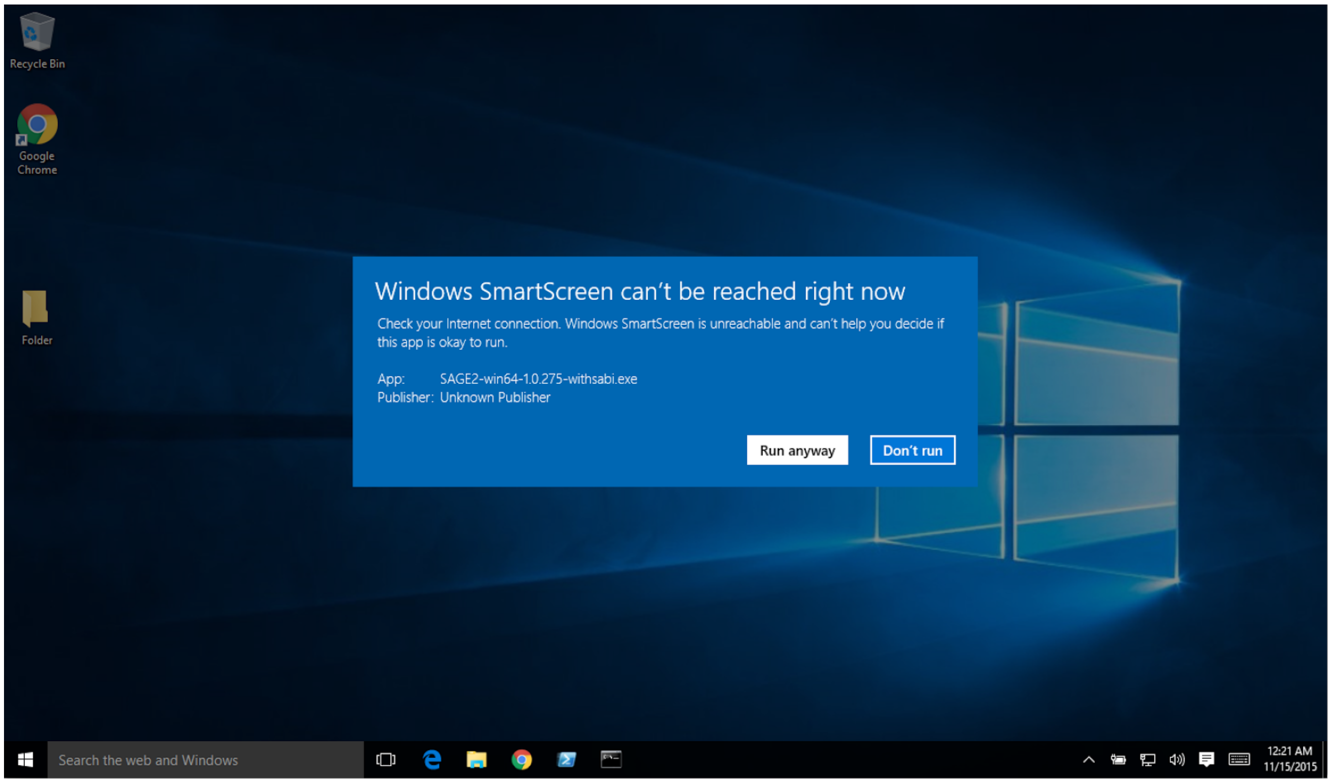
renambot@gmail.com November 14, 2017 news, release

Navigate to the SAGE2 website: *sage2.sagecommons.org.* The latest binaries are available in the *Get SAGE2* page.



Installation

Select and download the Windows binary. It comes as self-extracting archive file, it contains all the files needed to run SAGE2.

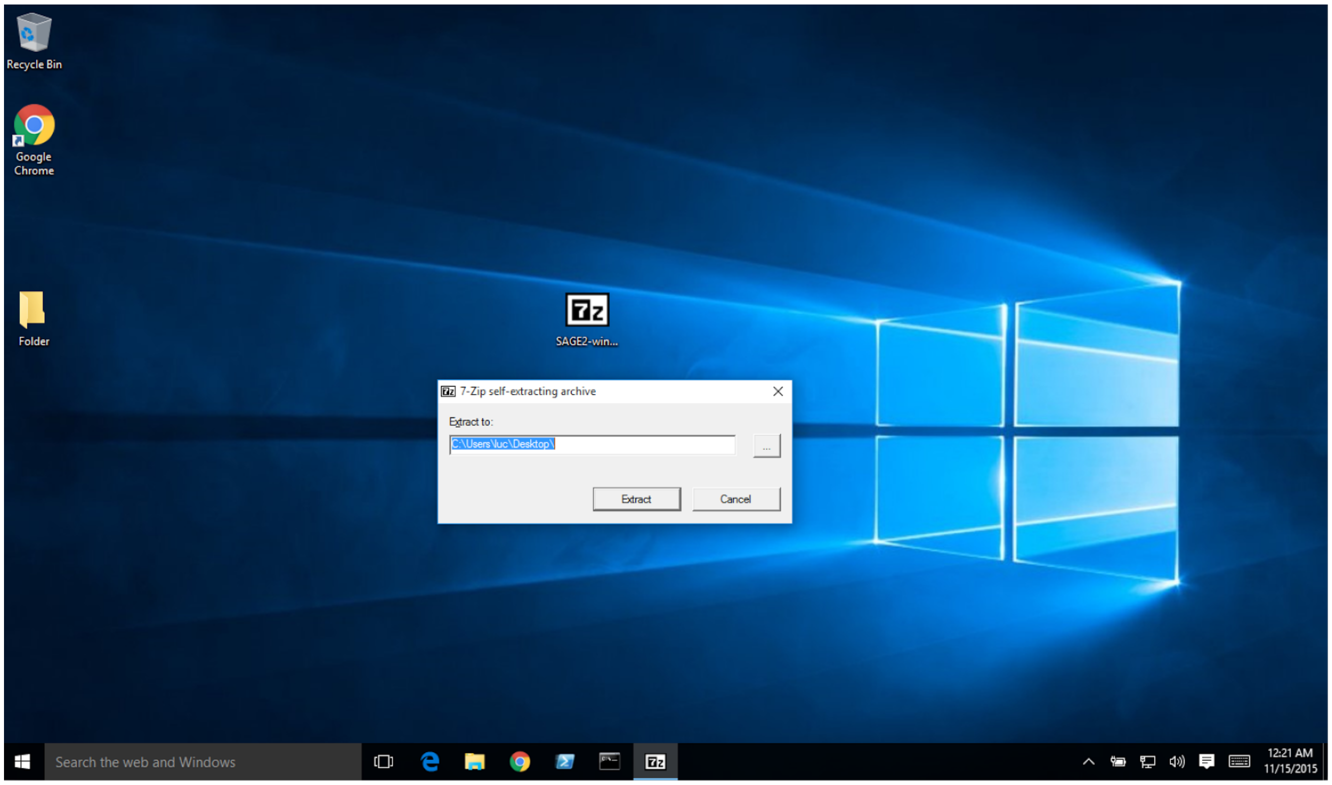


Windows SmartScreen can't be reached right now

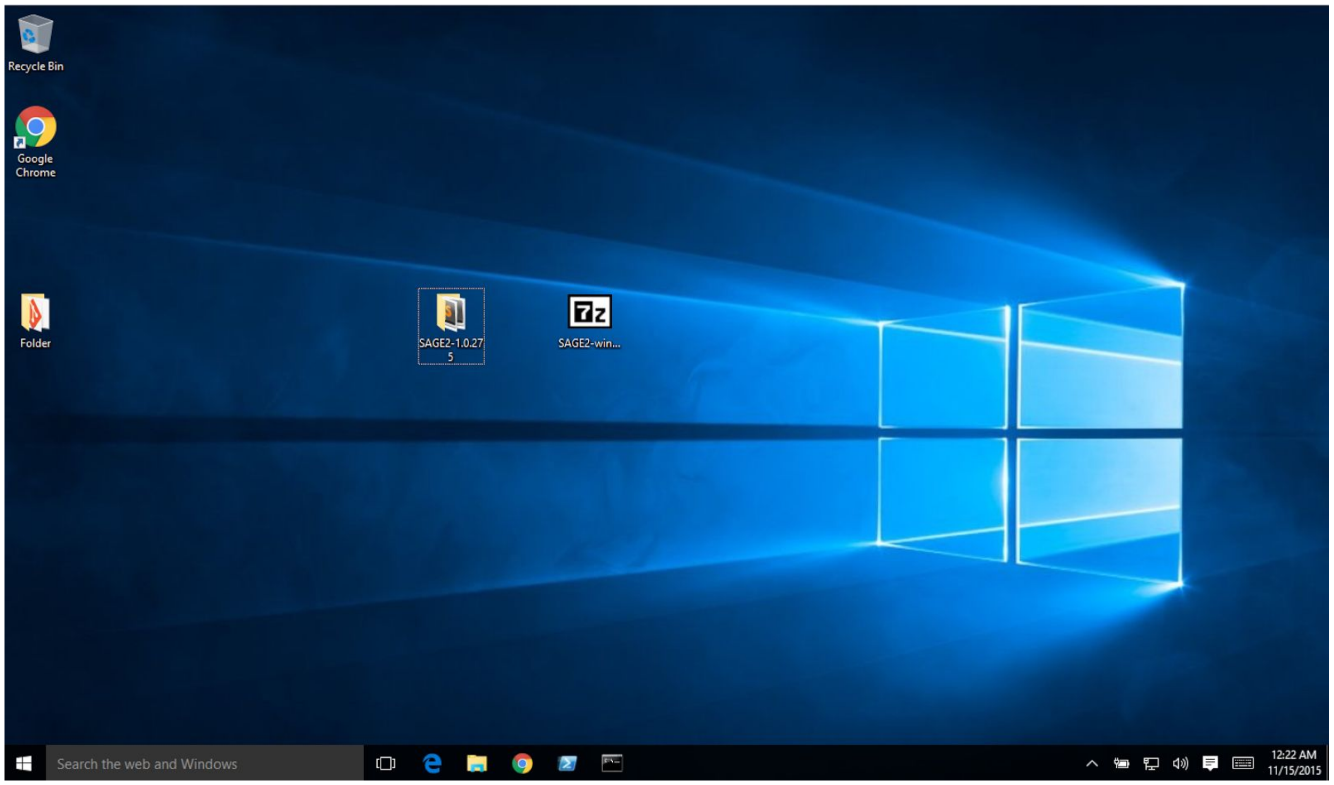
Check your Internet connection. Windows SmartScreen is unreachable and can't help you decide if this app is okay to run.

App: SAGE2-win64-1.0.275-withsabi.exe
Publisher: Unknown Publisher

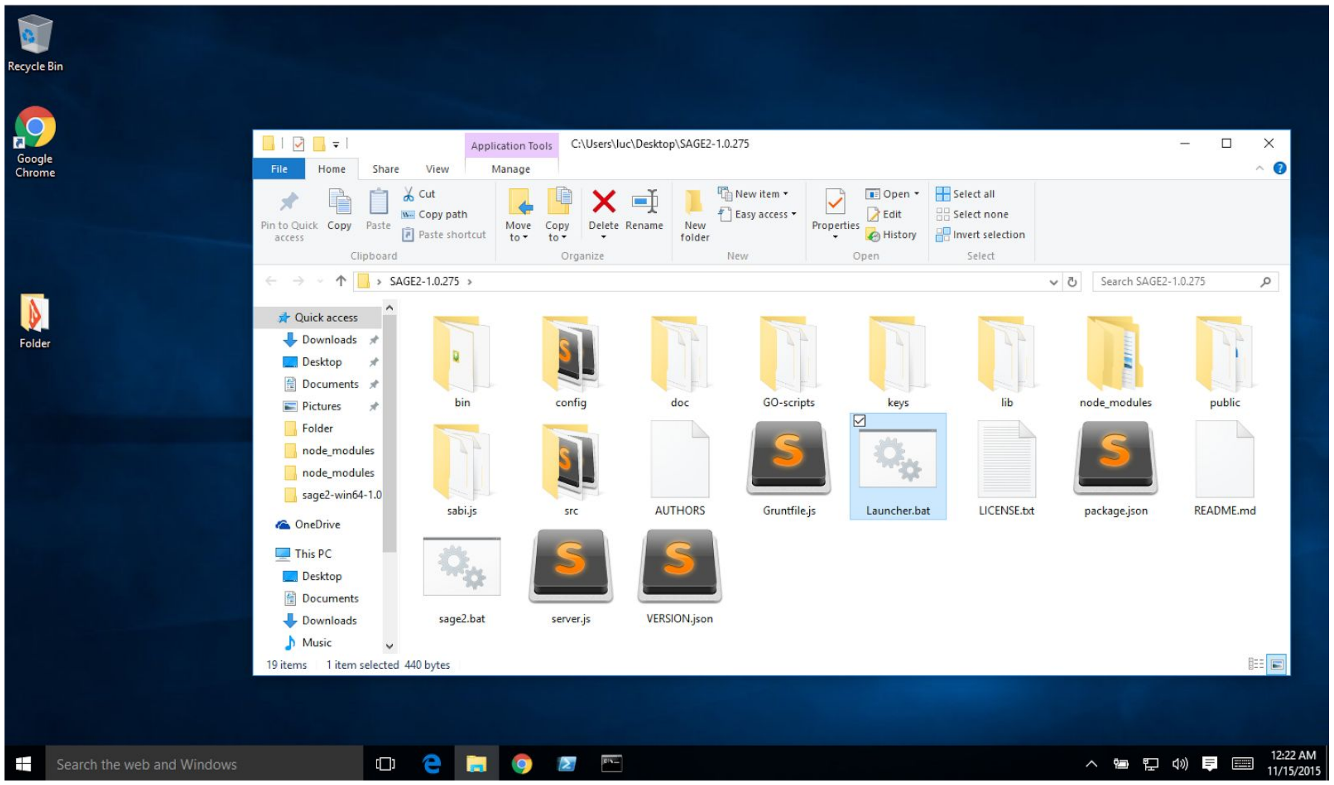
Windows might not recognize the downloaded file or its origin. Click ‘Run anyway’.



Select a destination folder to extract SAGE. For instance the desktop. SAGE will extract within a sub-folder similar to SAGE2-3.0.xxx.

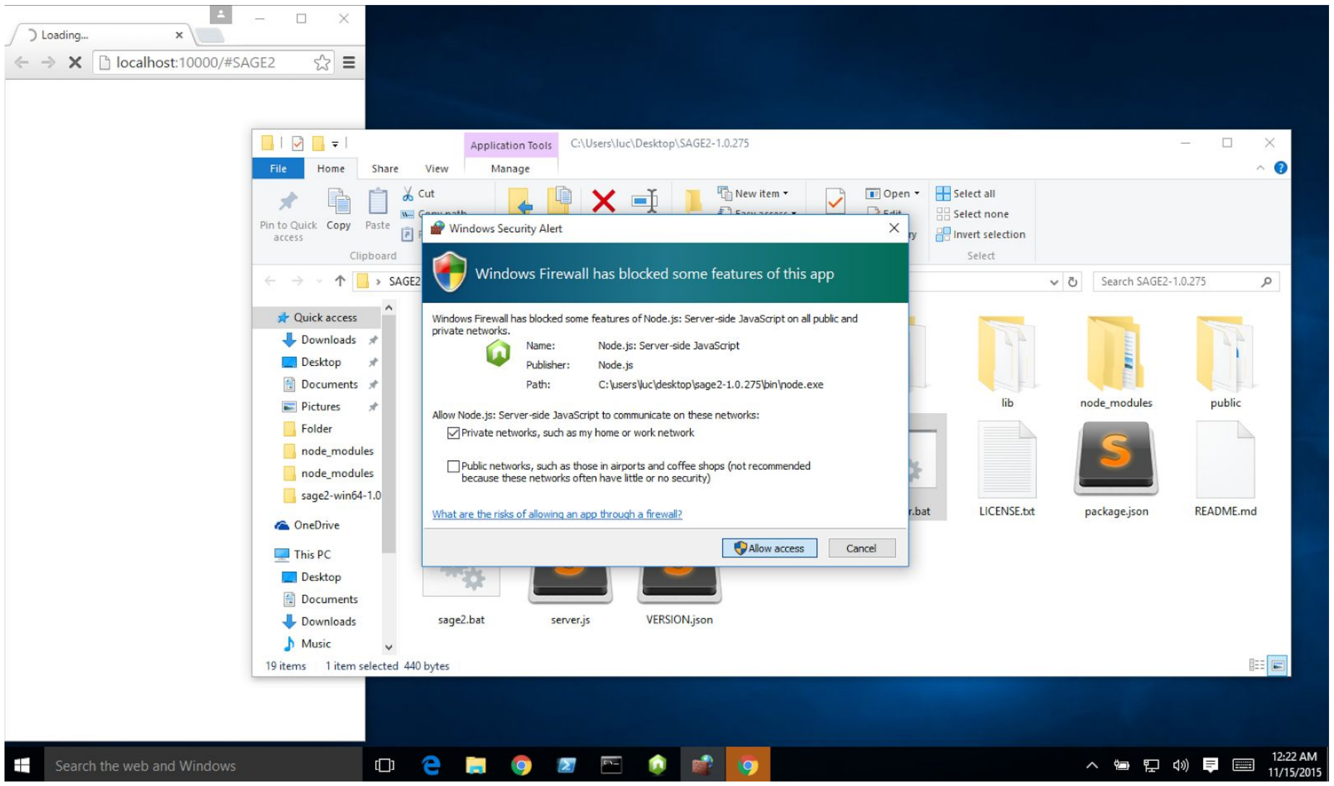


Once uncompressed, you can discard the .exe file.



The execution of SAGE2 is controlled by a 'launcher' process (called *sabi.js*). To start the launcher, double-click *'Launcher.bat'.* The launcher is itself a web server, which can be accessed remotely for remote management.

As part of the launch process it will open Chrome.
If Chrome has not been installed yet, please do so now.


Windows Security Alert

Windows Firewall has blocked some features of this app

Windows Firewall has blocked some features of Node.js: Server-side JavaScript on all public and private networks.

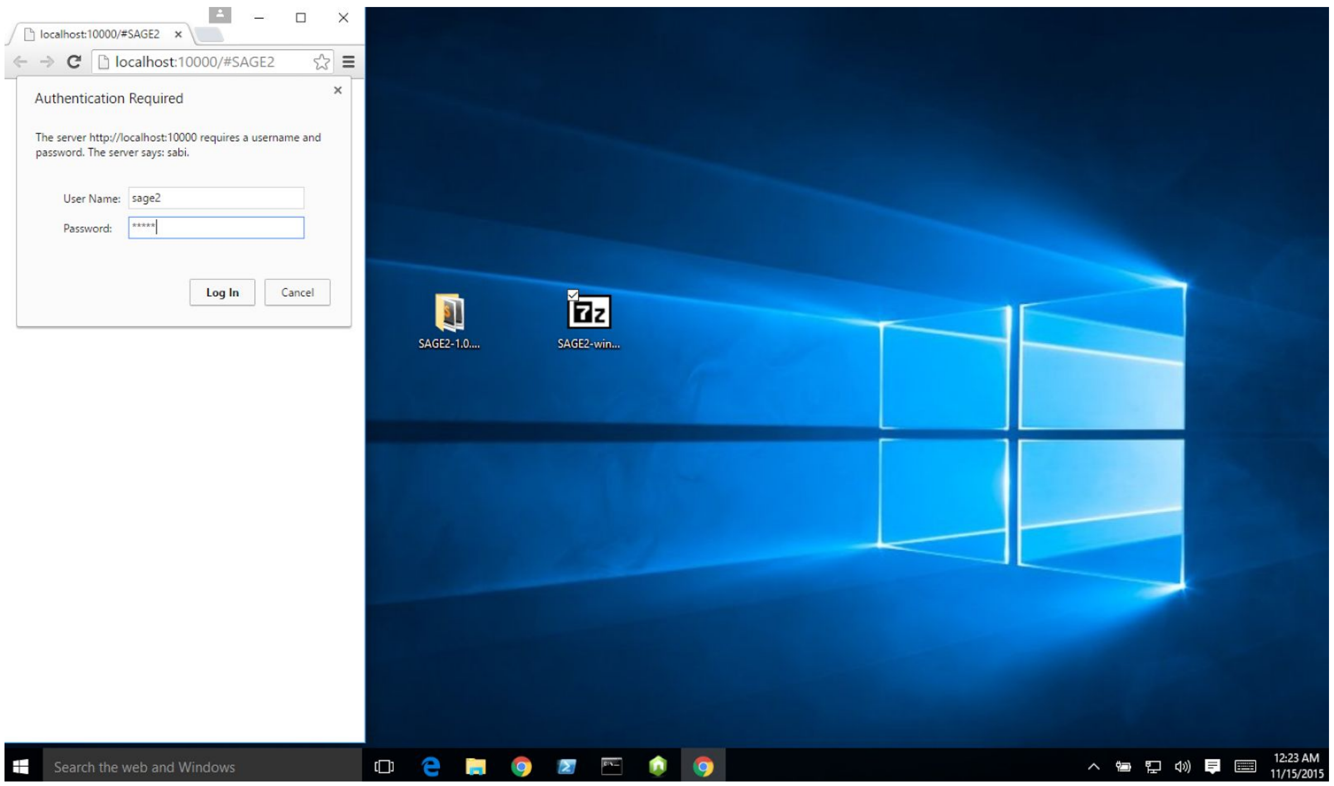
Name:	Node.js: Server-side JavaScript
Publisher:	Node.js
Path:	C:\users\luc\desktop\sage2-1.0.275\bin\node.exe

Allow Node.js: Server-side JavaScript to communicate on these networks:

- [✓] Private networks, such as my home or work network
- [ ] Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

What are the risks of allowing an app through a firewall?

If a firewall is running, a popup might appear to ask you to allow ‘node.exe’ (javascript runtime) to access the network. Click *‘Allow access’.*
In case the computer which SAGE2 is being installed on may move to a different network, it may be simpler to check both Private and Public options, allow access when on either.



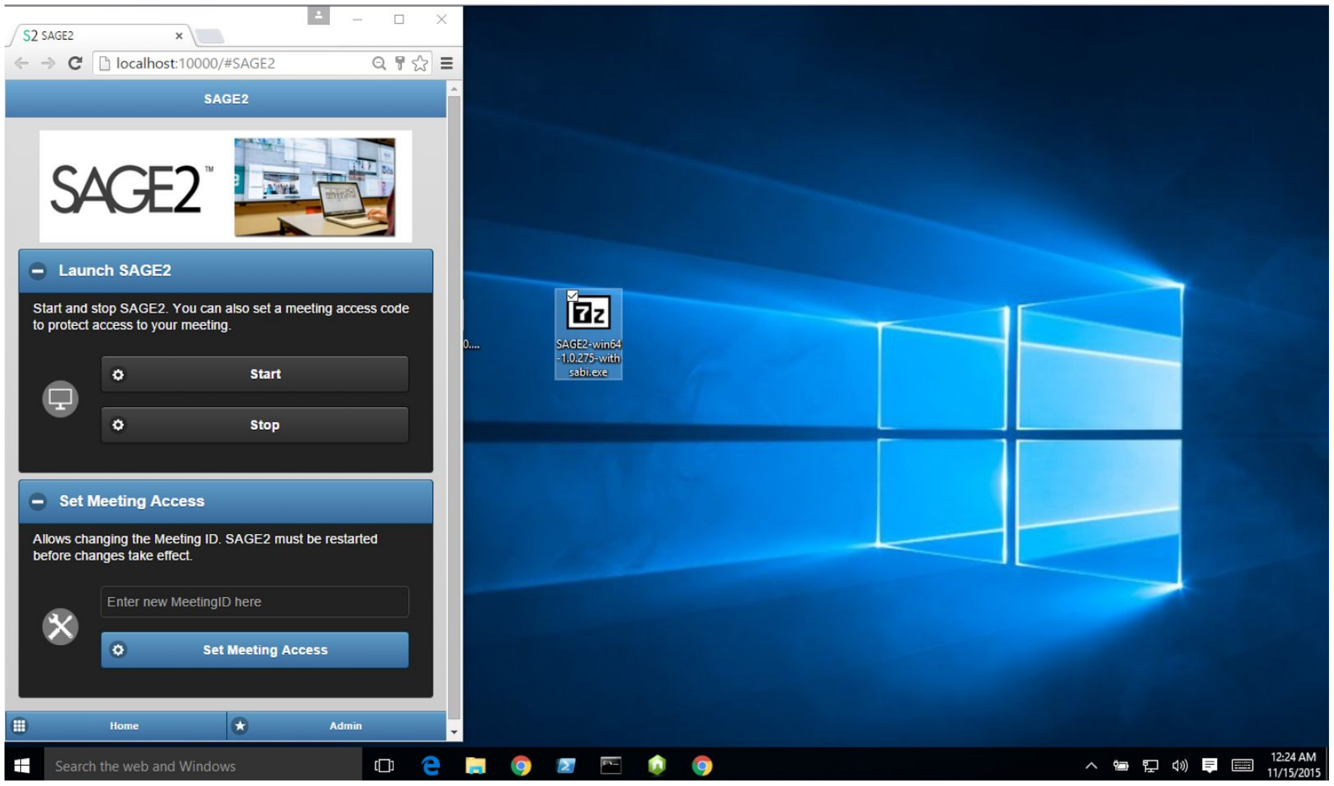
Authentication Required

The server http://localhost:10000 requires a username and password. The server says: sabi.

User Name: sage2

Password: *****

The launcher is now running at the following URL: http://localhost:10000. A chrome window is opened automatically to the launcher. The basic functions are accessible at http://localhost:1000/#SAGE2
The launcher is protected by a password. **Default values are, name: *sage2* and password: *sage2*.** The password can be changed in the *Admin* page.


SAGE2

Launch SAGE2

Start and stop SAGE2. You can also set a meeting access code to protect access to your meeting.

Start

Stop

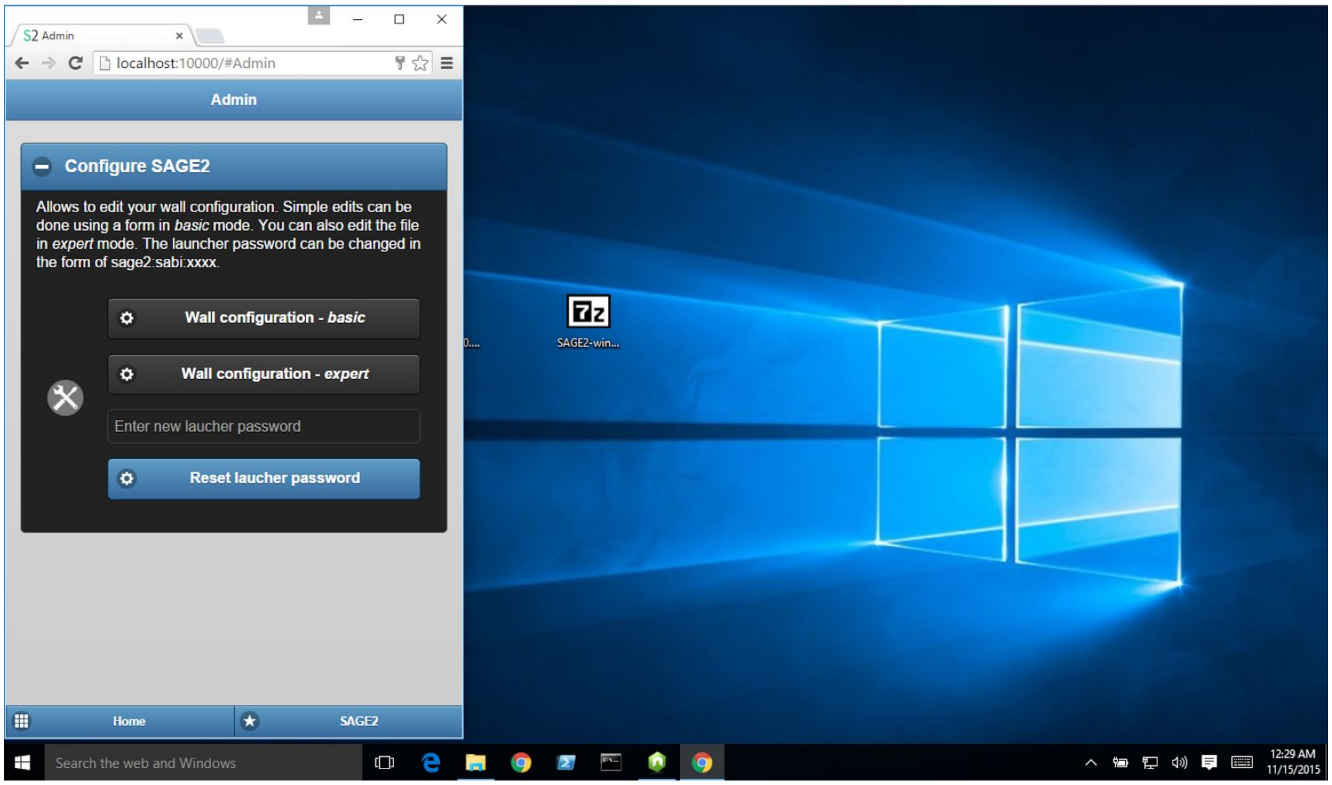
Set Meeting Access

Allows changing the Meeting ID. SAGE2 must be restarted before changes take effect.

Enter new MeetingID here

Set Meeting Access

The basic functions are accessible at http://localhost:10000/#SAGE2 : Start a SAGE2 session, stop a SAGE2 sessions, and set a meeting access ID (password for end-users to access the SAGE2 session).



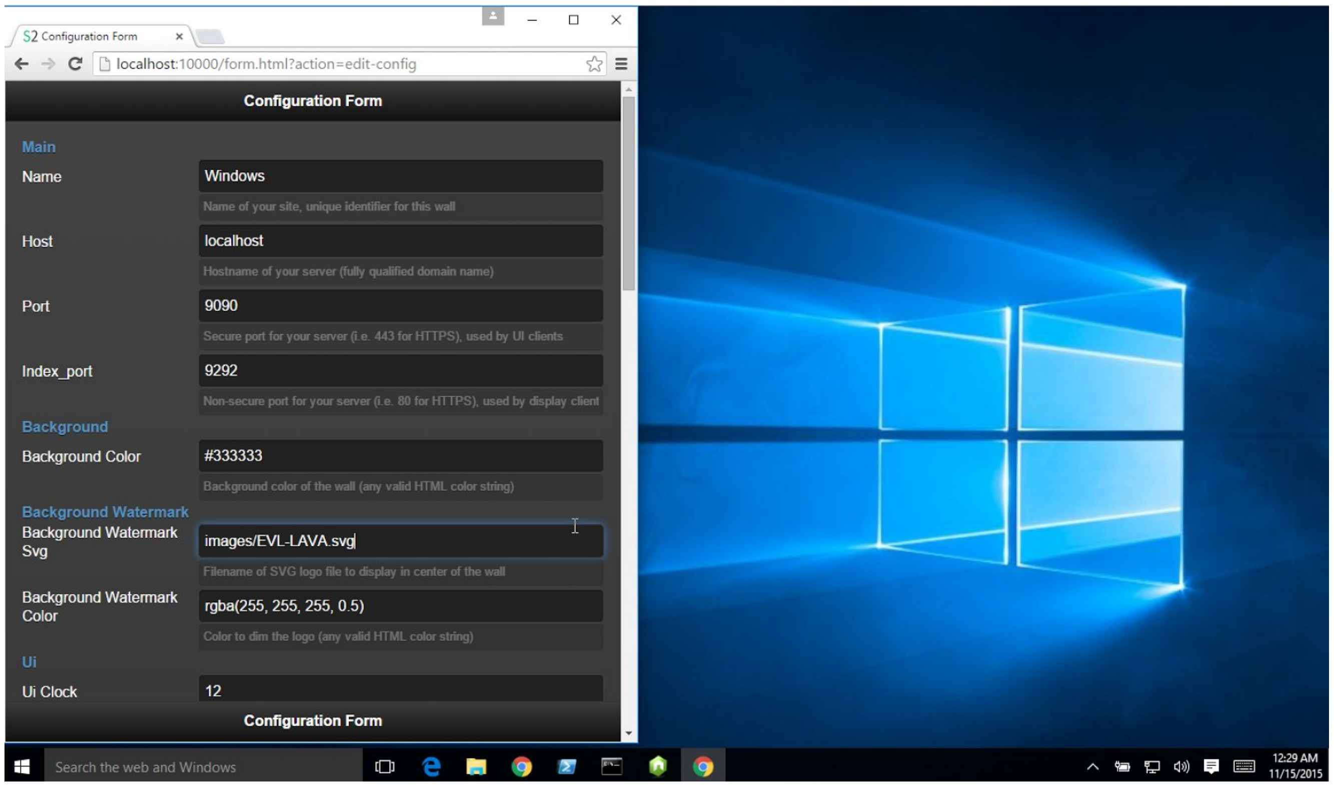
Allows to edit your wall configuration. Simple edits can be done using a form in *basic* mode. You can also edit the file in *expert* mode. The launcher password can be changed in the form of sage2:sabi:xxxx.

Wall configuration - *basic*
Wall configuration - *expert*
Enter new laucher password
Reset laucher password
Home
SAGE2
SAGE2-win...
Search the web and Windows
12:29 AM
11/15/2015

The admin page (http://localhost:10000/#Admin) is accessible by pressing the button at the bottom of the page. One can edit the SAGE2 wall configuration using either an assisted web form, basic web form, or a JSON file editor.

- Assisted - The assisted form as recommended for new users, as it will attempt to detect system settings and recommend values based on what it found. There are also buttons to easily add alternative hosts and remote sites.
- Basic - The basic form includes the most commonly entered fields
- Expert - The JSON file editor is recommended for those are are familiar with the process and JSON structure. This provides an interface to write the JSON file containing the configuration settings the server will use. The file must be syntactically correct in order for the server to work.

To change the launcher password, enter your new password in the text box (in the form *sage2:sabi:newpassword*) and click ‘Reset launcher password’. The password will be activated at the next start of the launcher.


S2 Configuration Form

localhost:10000/form.html?action=edit-config

Configuration Form

Main

Name	Windows
	Name of your site, unique identifier for this wall
Host	localhost
	Hostname of your server (fully qualified domain name)
Port	9090
	Secure port for your server (i.e. 443 for HTTPS), used by UI clients
Index_port	9292
	Non-secure port for your server (i.e. 80 for HTTPS), used by display client

Background

Background Color	#333333
	Background color of the wall (any valid HTML color string)

Background Watermark

Background Watermark Svg	images/EVL-LAVA.svg
	Filename of SVG logo file to display in center of the wall
Background Watermark Color	rgba(255, 255, 255, 0.5)
	Color to dim the logo (any valid HTML color string)

Ui

Ui Clock	12

Configuration Form

The SAGE2 wall assisted configuration form lets the user quickly edit existing values (hostname, port, resolution, …). Clicking ‘Save’, ‘Save and Make Certificates’ or ‘Cancel’ will return to the previous page.

When running SAGE2 for the first time, click ‘Save and Make Certificates’. The SAGE2 server needs certificates in order to allow client connections.

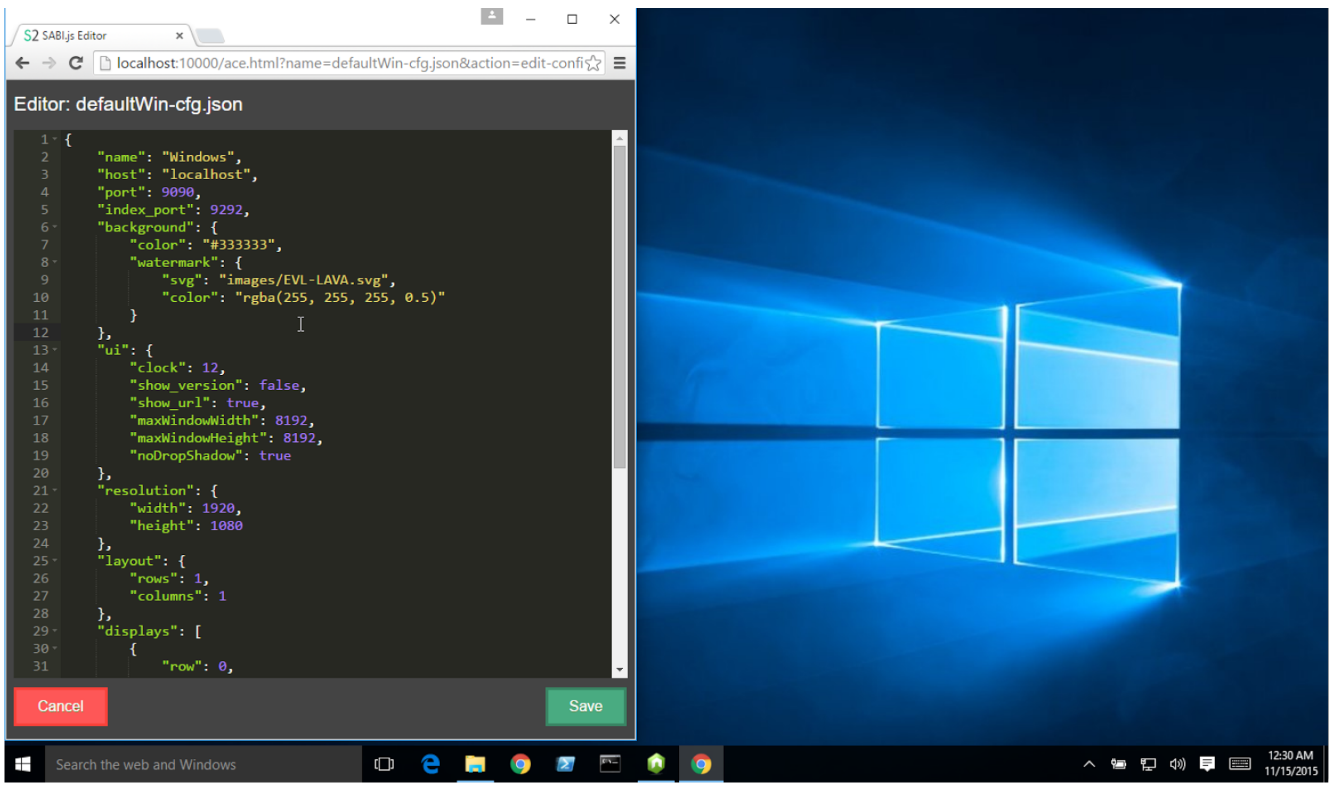
If you have signed certificates, edits to the configuration should probably use the ‘Save’ option instead because the ‘Save and Make Certificates’ will overwrite any existing certificate that matches the Host or any of the Alternate Hosts entries.

Configuration Page

- **Hostname / IP to display**: SAGE2 shows this value in the top left corner of the display. This will be visible for all using the SAGE2 wall. If you do not have a static IP, you may need to frequently update this value.
- **Alternate Hostnames / IPs**: Add all hostnames and IP's from which users should be able to access the SAGE2 wall in addition to the displayed value. Even if the machine is associated with other IP addresses, only the values listed will be allowed for users to connect in on.
- **Port number**: By default this is 9292. But will require everyone who connects to your wall to add :9292 to the address in order to access. To use standard http port, use 80. This will remove the need to specify the port number. Secure Port Number: By default 9090. By using 443, users connecting to the SAGE2 wall will not need to specify the secure port.
- **Resolution of display**: This value must be in pixels. Do not include suffix or units marker. Layout: Number of panels in the display configuration. * Note: If you intend to use the web controller, the entered resolution value should be the entire width and height of all panels and for the panel layout it should be 1x1. The reason is that the web controller will launch one firefox window large enough to span the entire display.
- **Remote Sites**: Enter in sites that you wish to see the status and share files with. Label: The name to display on the display. Hostname / IP address: Valid hostname / ip which the remote site can be accessed. Secure Port Number: The port number that the remote site uses.
- **Dependency location on the computer**: Where the installations for ImageMagick and FFMpeg are installed. If you are using the SAGE2 binary zip, then this should always be bin\
- **Background**: A background color can be entered as a CSS hex value. If an image is desired to be shown, the path from the public directory of SAGE2 can be used. The path must be within the public directory otherwise the file cannot be served. The Option to use an SVG is also available, this too must use a public directory path reference otherwise it cannot be served.
- **Meeting ID**: When users connect to the SAGE2 wall they will be prompted for the meeting ID before being able to see content. This applies to the UI, display, and audio manager. Web Controller Password: Necessary if you want to use the web controller to remotely start, stop and reset the meeting ID. Configuration Page Password: Necessary to access the configuration page again.

For more configuration variables, see:

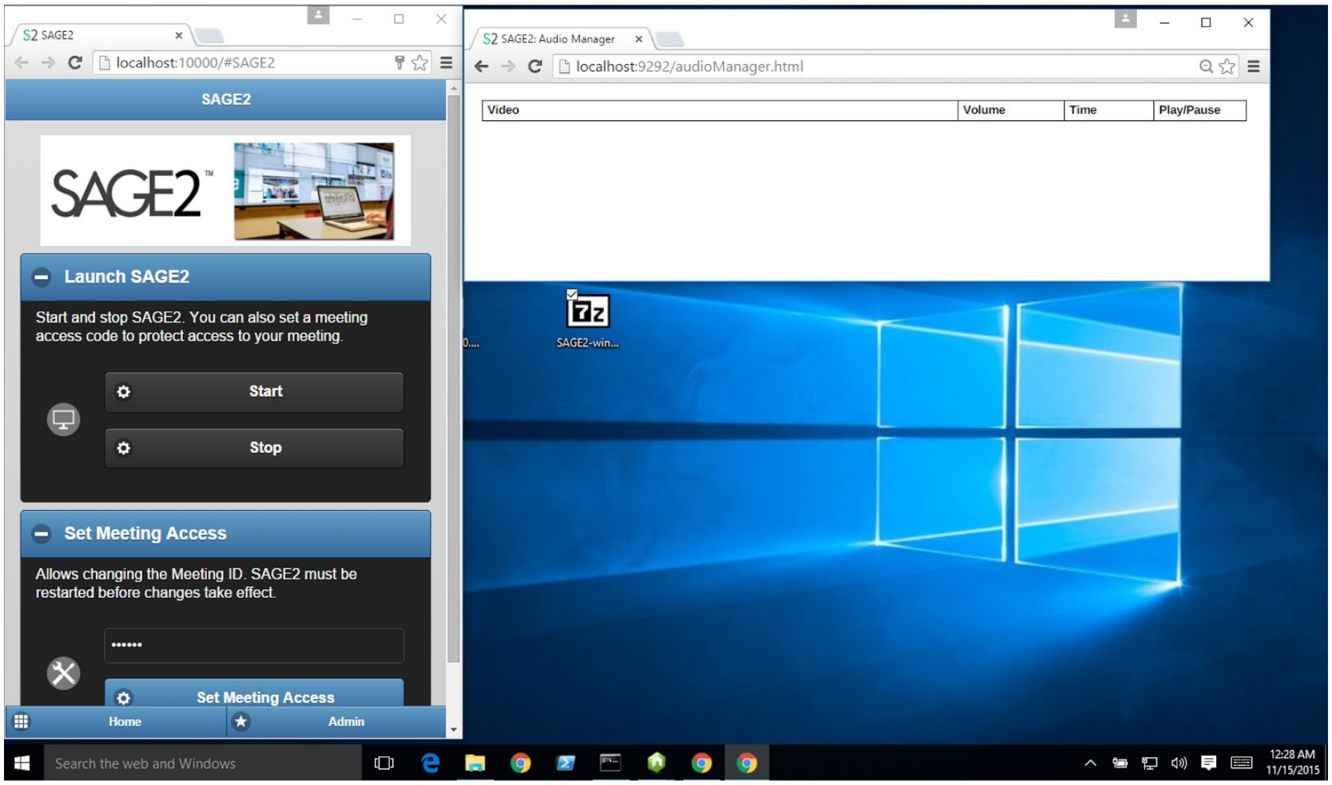
https://bitbucket.org/sage2/sage2/wiki/Configuration


S2 SABI.js Editor
localhost:10000/ace.html?name=defaultWin-cfg.json&action=edit-confi
Cancel
Save
Search the web and Windows
12:30 AM
11/15/2015

Editor: defaultWin-cfg.json

```
{
    "name": "Windows",
    "host": "localhost",
    "port": 9090,
    "index_port": 9292,
    "background": {
        "color": "#333333",
        "watermark": {
            "svg": "images/EVL-LAVA.svg",
            "color": "rgba(255, 255, 255, 0.5)"
        }
    },
    "ui": {
        "clock": 12,
        "show_version": false,
        "show_url": true,
        "maxWindowWidth": 8192,
        "maxWindowHeight": 8192,
        "noDropShadow": true
    },
    "resolution": {
        "width": 1920,
        "height": 1080
    },
    "layout": {
        "rows": 1,
        "columns": 1
    },
    "displays": [
        {
            "row": 0,
```

More complex edits can be done in the ‘expert’ mode. This a fully functional JSON editor with syntax highlighting and syntax validation (see symbols in the left margin). Click ‘Cancel’ or ‘Save’ to cancel or validate your entries and return to the launch page.

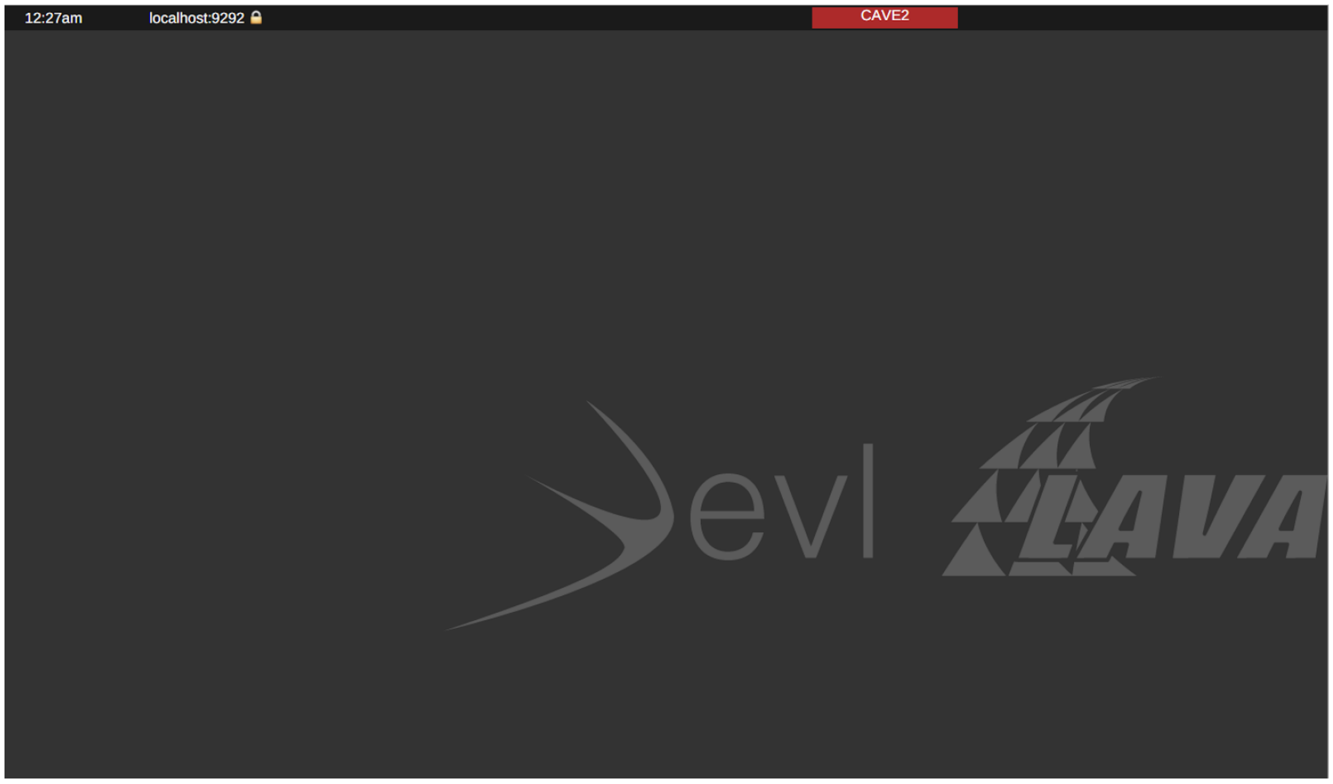


Start and stop SAGE2. You can also set a meeting access code to protect access to your meeting.

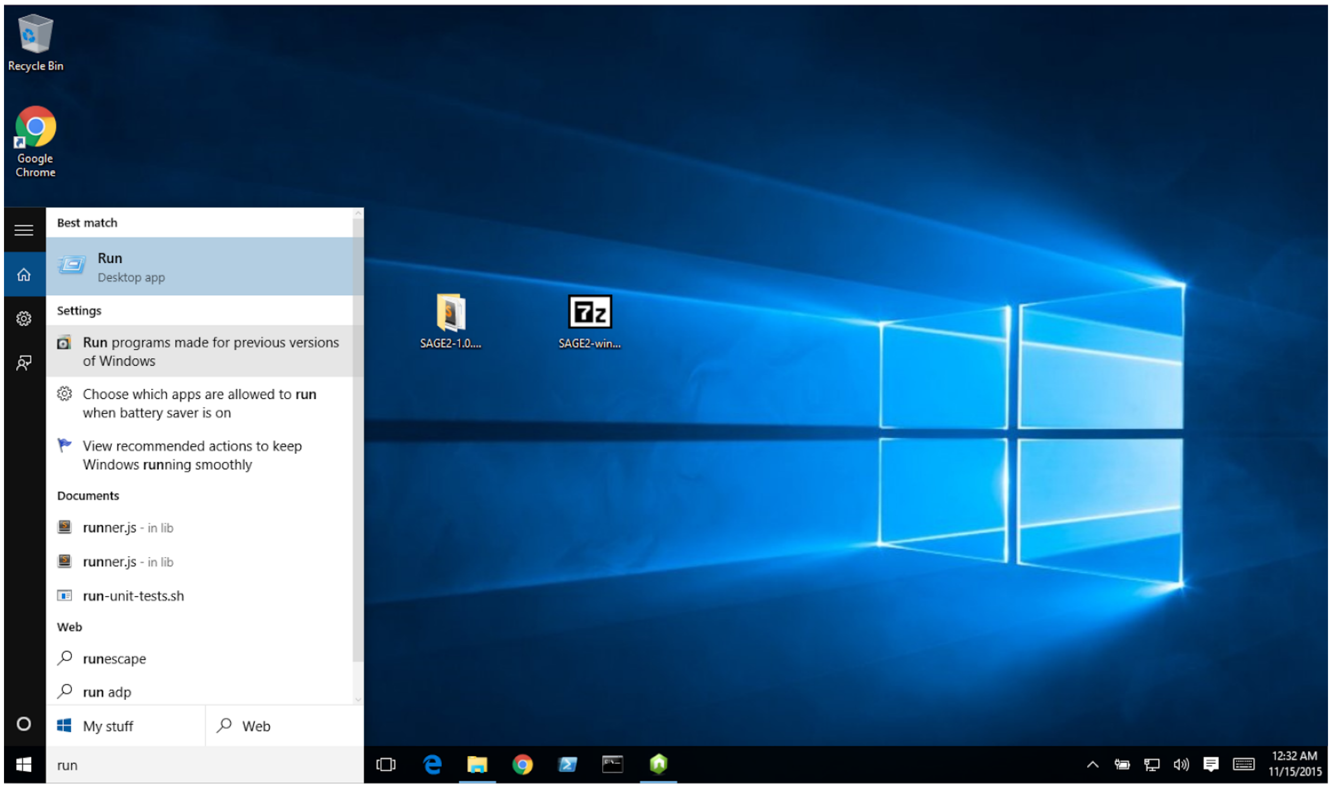
Allows changing the Meeting ID. SAGE2 must be restarted before changes take effect.

Video	Volume	Time	Play/Pause

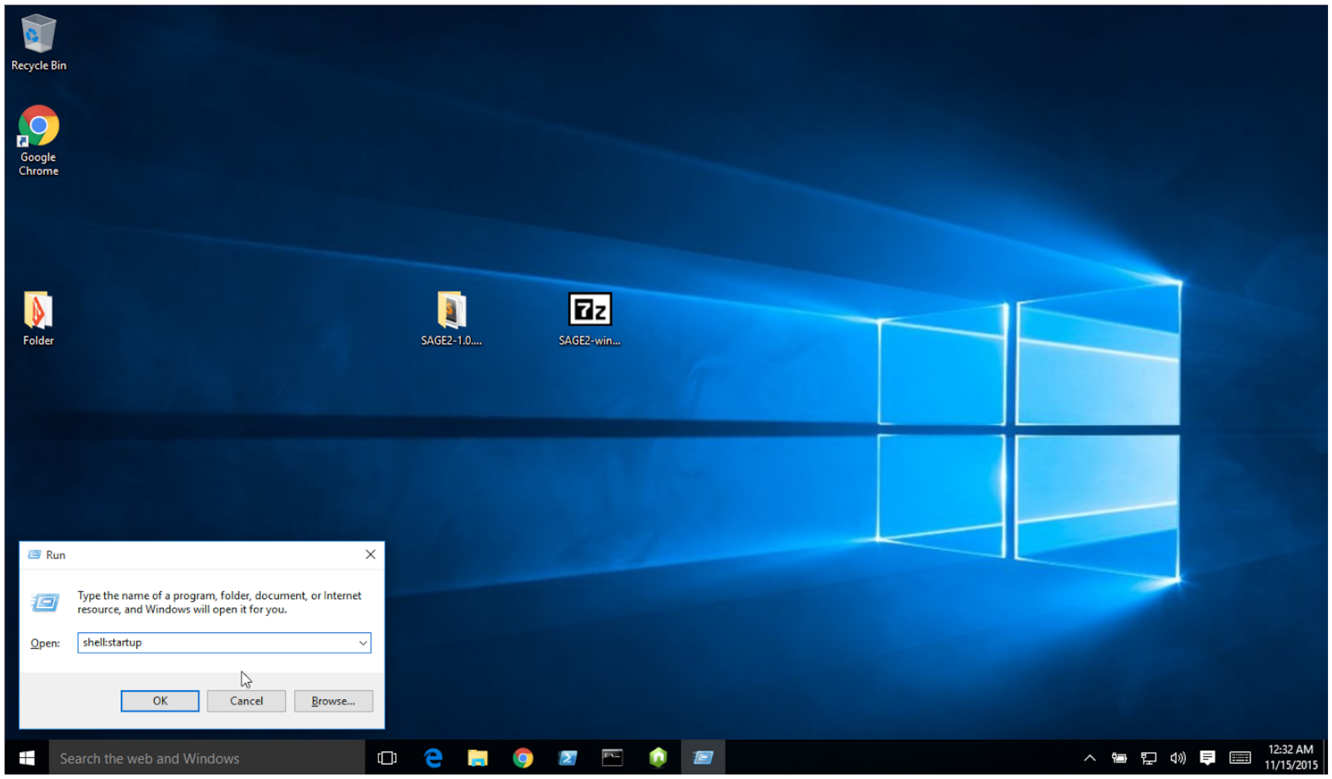
Once the configuration is finalized, one can start SAGE2 by just pressing the *'Start'* button. Behind the scene, a SAGE2 server is launched (inside an automatically minimized window) and two clients are started inside two Electron windows: an audio manager and a display client (starts fullscreen).



Here’s the default client, occupying the full screen. The URL of the SAGE2 is shown on the top left menu bar. If a lock symbol is present, it means the SAGE2 session is protected by a password (i.e. meeting ID).



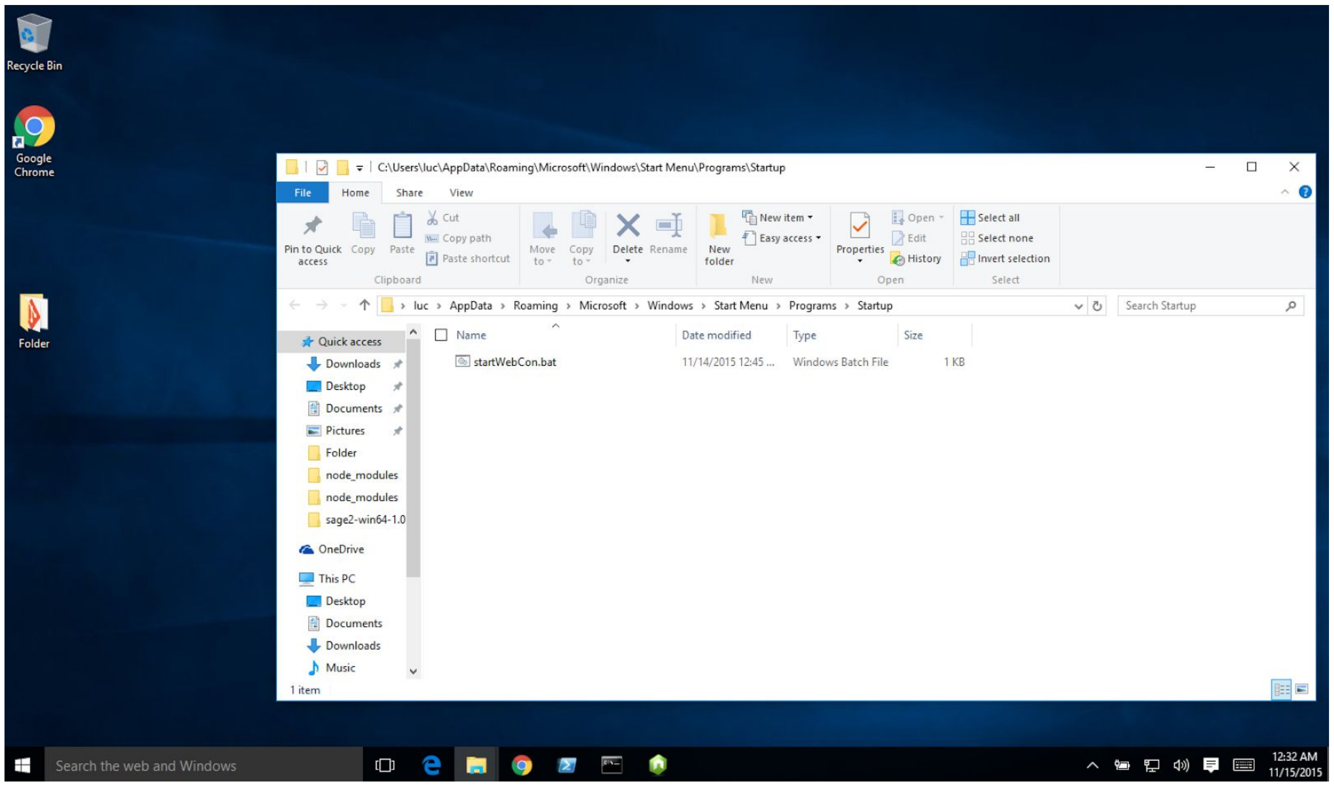
Once the launcher is started the first time, it creates a script to start automatically after each login of the current user (after reboot or logout). You can check the script by opening the a ‘Run’ command.



Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: shell:startup

To access the startup item folder for your current version of Windows, type ‘shell:startup’ in the ‘Open’ textbox of the ‘Run’ tool.



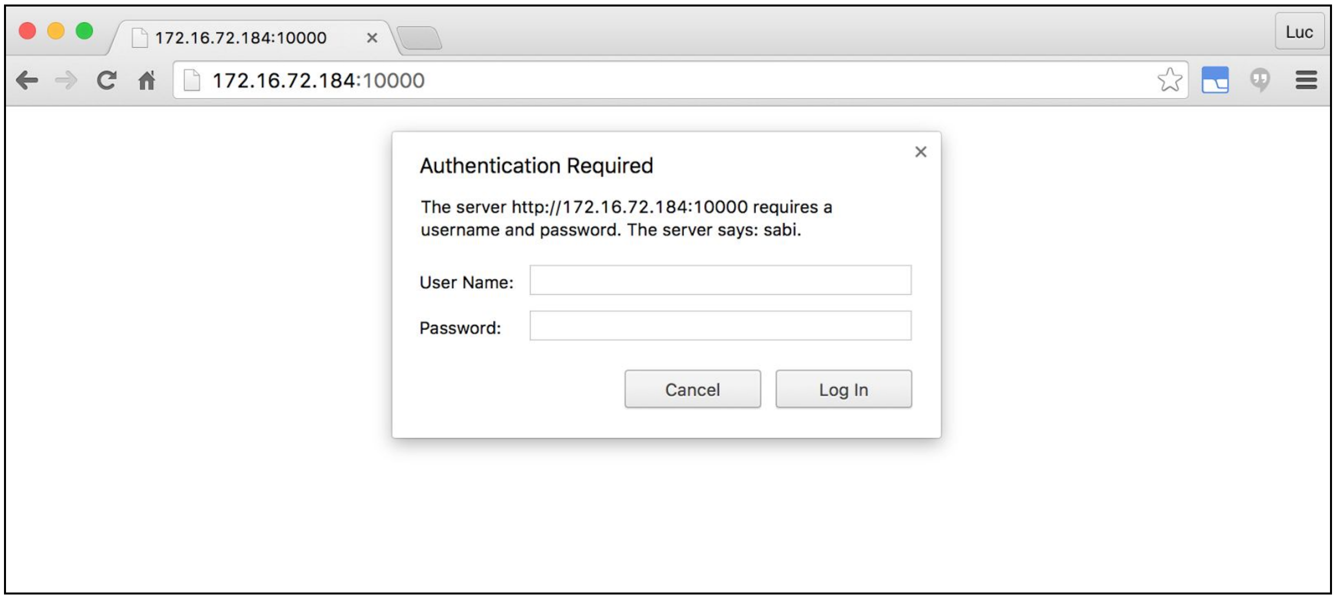
C:\Users\luc\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup

Name	Date modified	Type	Size
startWebCon.bat	11/14/2015 12:45 ...	Windows Batch File	1 KB

1 item

12:32 AM
11/15/2015

The file explorer will open a window showing your startup items. You can check or edit the script: *startWebCon.bat*. This will make sure that the launcher is always started after a reboot, crash or update.



Authentication Required

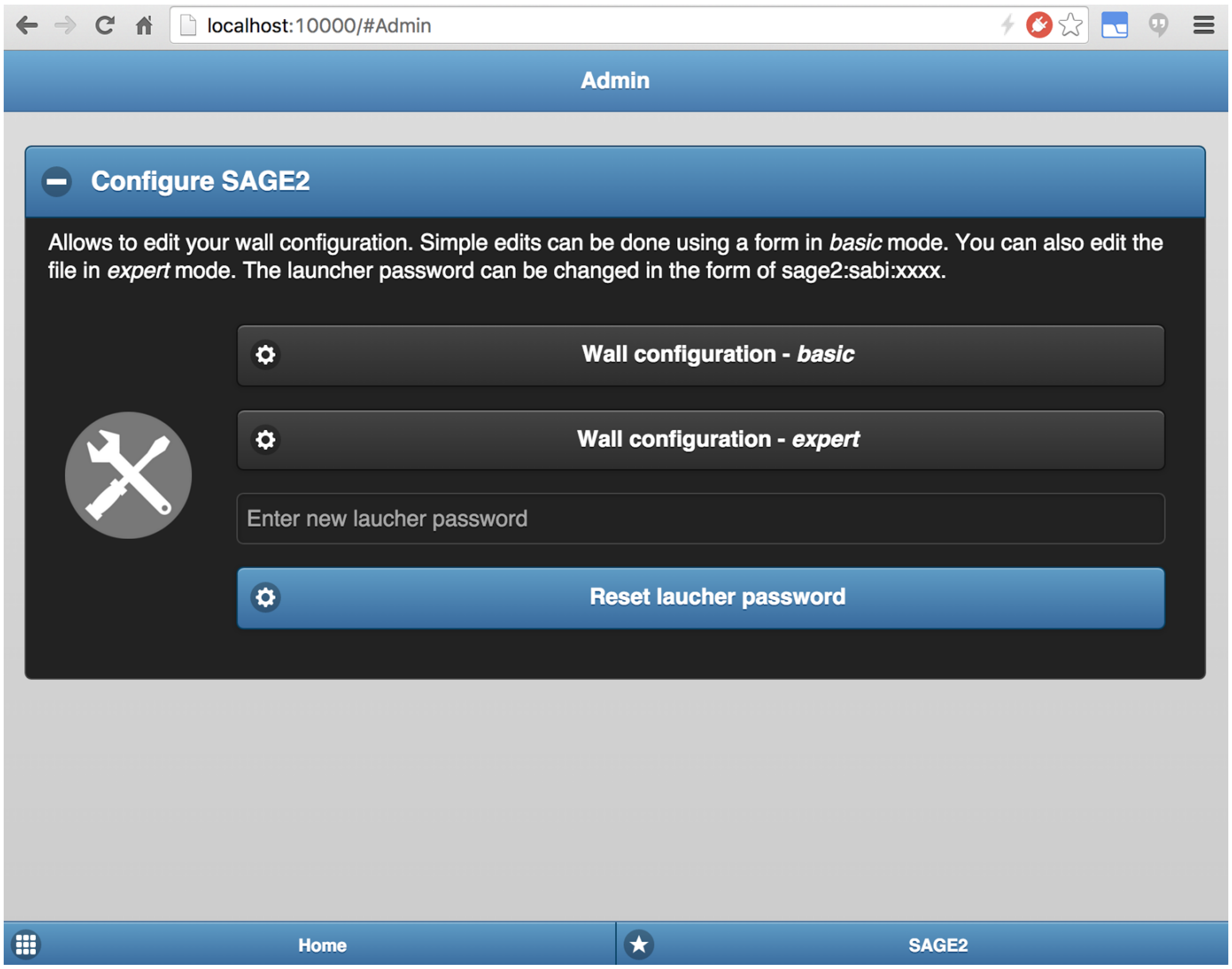
The server http://172.16.72.184:10000 requires a username and password. The server says: sabi.

User Name:

Password:

You can access remotely the launcher at the port 10000 on the SAGE2 machine. The default name is *sage2* and the default password is also *sage2*. You can change the launcher password in the *admin* page. The password must be specified in the form: *sage2:sabi:newpassword.*

This is where knowing the IP address of your SAGE2 machine is important. While working on the SAGE2 machine itself, the launcher is accessible at http://localhost:10000. However, from your laptop, your need to use full IP address or hostname of the machine: http://myserver.com:10000.


localhost:10000/#Admin

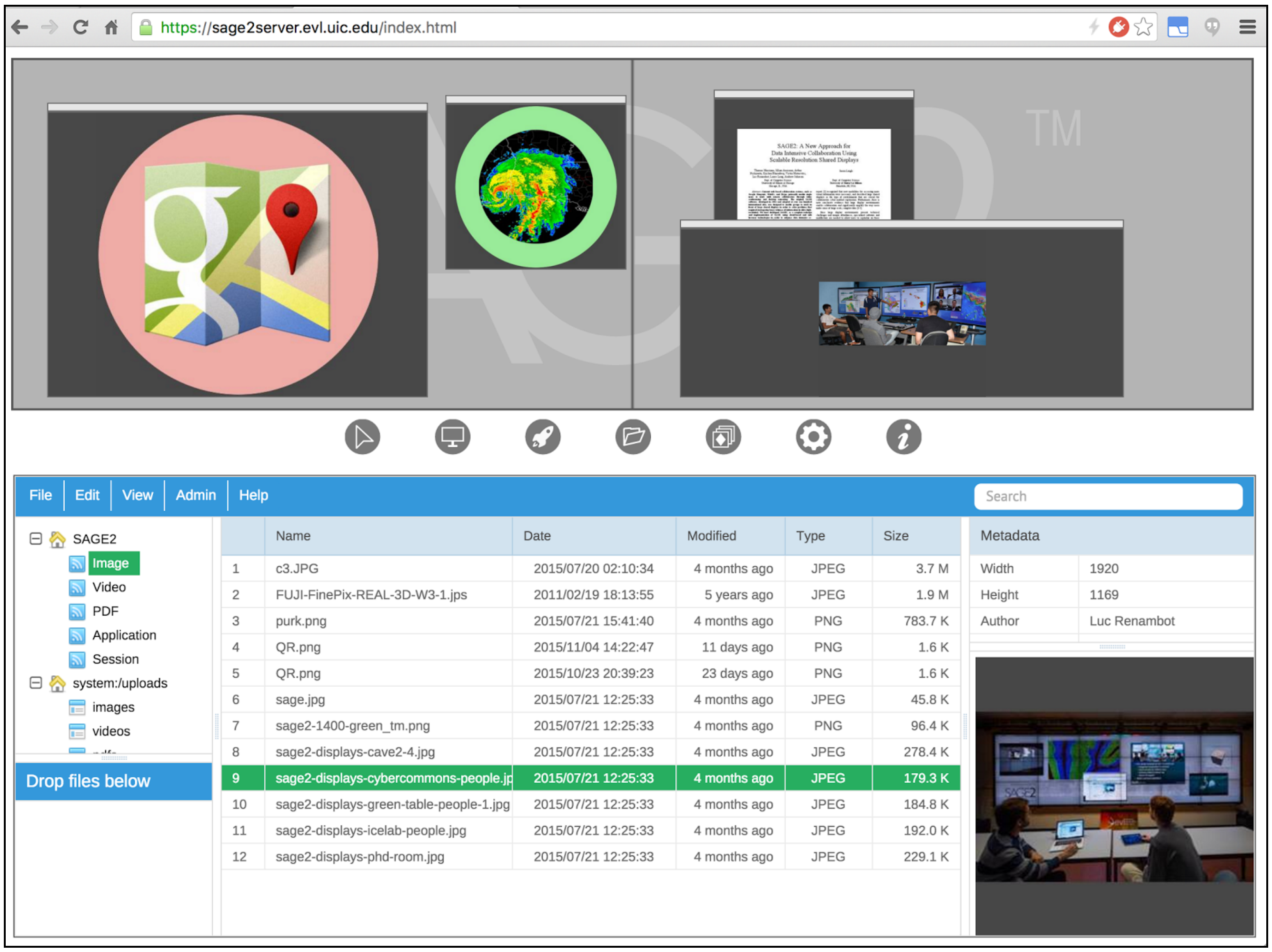
Admin

Configure SAGE2

Allows to edit your wall configuration. Simple edits can be done using a form in *basic* mode. You can also edit the file in *expert* mode. The launcher password can be changed in the form of sage2:sabi:xxxx.

Wall configuration - *basic*
Wall configuration - *expert*
Enter new laucher password
Reset laucher password
Home
SAGE2

Admin page of the launcher for SAGE2. Enter your new password (*sage2:sabi:newpassword*) and click ‘Reset launcher password).



	Name	Date	Modified	Type	Size
1	c3.JPG	2015/07/20 02:10:34	4 months ago	JPEG	3.7 M
2	FUJI-FinePix-REAL-3D-W3-1.jps	2011/02/19 18:13:55	5 years ago	JPEG	1.9 M
3	purk.png	2015/07/21 15:41:40	4 months ago	PNG	783.7 K
4	QR.png	2015/11/04 14:22:47	11 days ago	PNG	1.6 K
5	QR.png	2015/10/23 20:39:23	23 days ago	PNG	1.6 K
6	sage.jpg	2015/07/21 12:25:33	4 months ago	JPEG	45.8 K
7	sage2-1400-green_tm.png	2015/07/21 12:25:33	4 months ago	PNG	96.4 K
8	sage2-displays-cave2-4.jpg	2015/07/21 12:25:33	4 months ago	JPEG	278.4 K
9	sage2-displays-cybercommons-people.jp	2015/07/21 12:25:33	4 months ago	JPEG	179.3 K
10	sage2-displays-green-table-people-1.jpg	2015/07/21 12:25:33	4 months ago	JPEG	184.8 K
11	sage2-displays-icelab-people.jpg	2015/07/21 12:25:33	4 months ago	JPEG	192.0 K
12	sage2-displays-phd-room.jpg	2015/07/21 12:25:33	4 months ago	JPEG	229.1 K

Metadata	
Width	1920
Height	1169
Author	Luc Renambot

Once SAGE2 is running, you can access an interface client from your laptop using the IP address or hostname of your SAGE2 server. If your use an HTTP address, you will be redirected automatically to the secure address through HTTPS.
The snapshot above shows the file manager to manage and organize your assets. Press the button with a folder icon in the toolbar to open the file manager, press again to close.

Advanced configuration

For more complex Windows installations, you might need extra tools:

- Firefox - https://www.mozilla.org/en-US/firefox/new/
  - more flexible than Chrome to setup across multiple monitors
- AutoHotkey - http://ahkscript.org/
  - Windows macro and automation tool
- Notepad++ - https://notepad-plus-plus.org/
  - or any text editor

The launcher is configured by a JSON file: **sabi.js/config/sage2.json.** It defines mainly two operations: starting and stopping SAGE2. For a more complex display system, you must provide two **.bat** files which will perform these operations. We provide a script for a single-node single-display configuration. More complex setup can be supported using *AuthoKey* and *Firefox*.

Now, SAGE2 uses Electron as a display client (https://electron.atom.io) by default. You can still use Chrome if you prefer.

For instance:

Starting SAGE2: see **sabi.js/scripts/sage2_on_electron.bat**

- starts SAGE2 with the server script
- launches Electron with a audio manager
  - *http://hostname:port/audioManager.html*
- launches Chrome with a display client 0
  - *http://hostname:port/display.html?clientID=0*

Stopping SAGE2: see **sabi.js/scripts/sage2_off.bat**

- kills the SAGE2 server (using taskkil)
- closes the audio manager
- closes the display client 0

Installation from sources

Installation instructions to setup SAGE2 for MacOS, Windows and Linux directly from sources.

- Note: When using multiple computers to run displays, the install instructions need only be performed on one which will host the server. All others need only connect through a browser and access their respective ID based on tile position.

Configuration

- Create a configuration file for your display environment
- Save file in /config folder
- Select your configuration file
    - Option 1: name your configuration file '-cfg.json'
        - (eg. host = thor.evl.uic.edu, config file is 'thor-cfg.json')
    - Option 2: create a file 'config.txt' in
        - Specify the path to your configuration file in 'config.txt'

Run

- Open Terminal / Cmd
    - `cd <SAGE2_directory>`
    - `node server.js` (options: `-l` log, `-i` interactive prompt, `-f <file>` specify a configuration file)
- Open Web Browser
    - Google Chrome
        - First time use - install SAGE2 Chrome Extension
    - Firefox
        - First time use - go to `about:config` and set `media.getusermedia.screensharing.enabled` to true and add domain(s) for SAGE2 server(s) to `media.getusermedia.screensharing.allowed_domains`
    - Electron
        - Open a terminal window and navigate to the SAGE2_directory.
        - `cd <SAGE2_directory>`
        - `./node_modules/.bin/electron electron.js -s http://hostname:port -d 0`
            - The above command will open one browser pointed at display 0, which is `display.html?clientID=0`
        - There are additional flags available:
            - -h, --help   output usage information
            - -V, --version   output the version number

        - -d, --display <n>   Display client ID number (int)
        - -s, --server <s>   Server URL (string)
        - -f, --fullscreen   Fullscreen (boolean)
        - -n, --no_decoration   Remove window decoration (boolean)
        - -x, --xorigin <n>   Window position x (int)
        - -y, --yorigin <n>   Window position y (int)
        - --width <n>   Window width (int)
        - --height <n>   Window height (int)
        - --password <s>   Server password (string)
        - --hash <s>   Server password hash (string)
        - --cache   Clear the cache
        - --console   Open the devtools console
    - SAGE2 pages
        - Display Client: https://<host>:<port>/display.html?clientID=<ID>
        - Audio Client: https://<host>:<port>/audioManager.html
        - SAGE UI: https://<host>:<port>
        - SAGE Pointer: https://<host>:<port>/sagePointer.html (Allow pop-ups)

Supported File Types

- Images
    - JPEG, PNG, TIFF, BMP, PSD
- Videos
    - MP4, M4V, WEBM, MOV (containing MP4)
- PDFs

Install for Windows 64-bit (7, 8.1, 10)

Download

- Download Google Chrome and install (follow installer instructions): click 'Download Chrome for another platform' and select 'Windows 8 / 7 64-bit'
- Download 7-Zip and install (follow installer instructions): .msi for 64-bit x64
- Download Node.js and install (follow installer instructions): version v8.x LTS or V9.x
- Download ImageMagick and install (follow installer instructions)
- Download Ghostscript and install (follow installer instructions): Windows (64 bit), GPL Release
- Download Git for Windows and install (follow installer instructions)
- Download FFmpeg (64-bit Shared and 64-bit Dev)

- Download ExifTool (Windows Executable)

Install FFMPEG

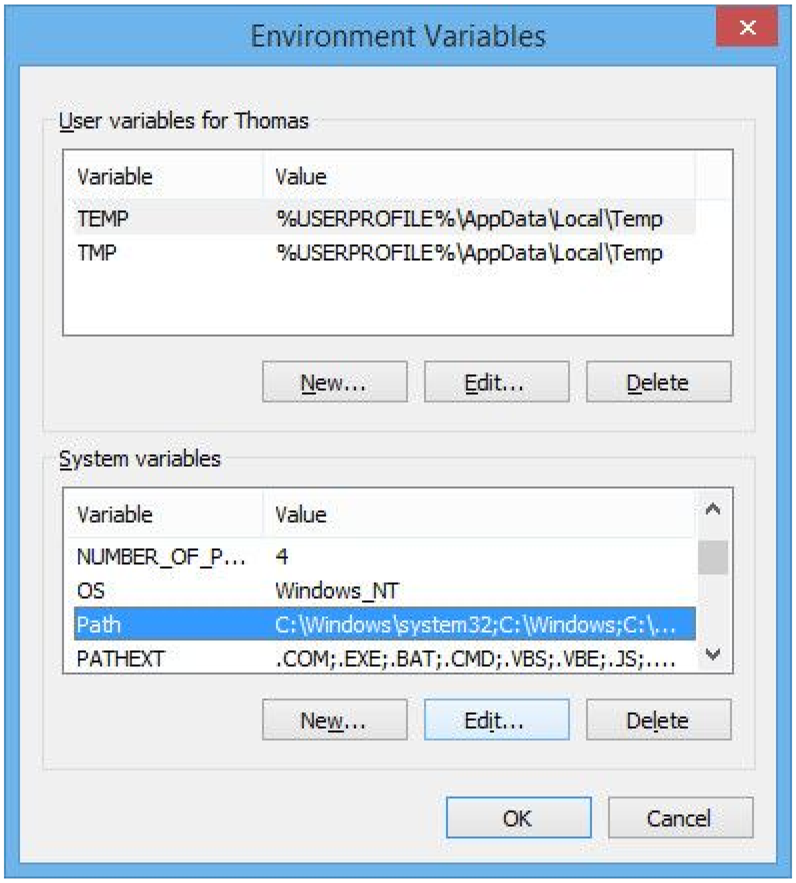
- Use 7-Zip to extract FFmpeg downloads (right-click > 7-Zip > Extract Here)
- Create folder 'C:\Dev' if it does not already exist
- Move extracted FFmpeg Shared folder to 'C:\Dev\' and name it 'ffmpeg-win64-shared'
- Move extracted FFmpeg Dev folder to 'C:\Dev\' and name it 'ffmpeg-win64-dev'

Install EXIFTOOL

- Use 7-Zip to extract ExifTool download (Right-click > 7-Zip > Extract Here)
- Create folder 'C:\local\bin' if it does not already exist
- Rename binary to 'exiftool.exe' and move to 'C:\local\bin'

Set Environment

- For Windows 7
    - Right-click 'Computer'
    - Click 'Properties'
    - Click 'Advanced system settings'
- For Windows 8.1
    - Click Windows Logo and type 'env'
    - Click 'Edit the system environment variables'
- Click 'Environment Variables'
- Add 'C:\Program Files (x86)\Git\bin;C:\Program Files\7-Zip;C:\local\bin;C:\Dev\ffmpeg-win64-shared\bin' to your system PATH variable
- You may have to log out and log back in for the environment variables to apply


Environment Variables

User variables for Thomas

Variable	Value
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
NUMBER_OF_P...	4
OS	Windows_NT
Path	C:\Windows\system32;C:\Windows;C:\...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;....

Clone SAGE2

- **Launch command prompt (cmd.exe)**
    - `cd <directory_to_install_SAGE2>`
    - `git clone https://bitbucket.org/sage2/sage2.git`

Generate HTTPS keys

- Edit '<SAGE2_directory>\keys\GO-windows.bat'
    - Add lines with list of hostnames for your server
    - Save file
- Double click 'GO-windows.bat'
    - On Windows 8.1 you will need right-click and select 'Run as Administrator'

Install Node.js Modules

- **Launch command prompt (cmd.exe)**
  - `cd <SAGE2_directory>`
  - `npm run in` (install pre-compiled binary modules) or `npm install` (compile and install binary modules - requires Visual Studio Express 2013 for Windows Desktop)
    - If binary packages are not available (such as after a new version of node) and need to be compiled, you will also need to install Python 2.7.10 (not 3.x)
    - Make sure python.exe is added to Path (either manually or enabled in during Python install under Customize Python).
- **Troubleshooting**
  - 'Error: ENOENT, stat 'C:\Users{User Name}\AppData\Roaming\npm'
    - You will need to manually create that folder
  - 'C:\Dev\ffmpeg-win64-dev\include\libavutil\common.h(34): fatal error C1083: Cannot open include file: 'inttypes.h': No such file or directory'
    - Run `npm install node-demux`
      - If you have multiple Visual Studio installs, you may also need to add `--msvc_version=2013`
      - Requires at least Visual Studio 2013 to install

Install for Mac OS X (10.7 - 10.11)

Download

- Download Google Chrome and install (follow installer instructions)
- Download Node.js and install (follow installer instructions)
- Download homebrew and install (Terminal command creates full install)
  - If you get a popup about installing the 'command line developer tools', select 'Install'
    - Press 'Return' in Terminal once command line tools install finishes
  - Run `brew doctor` once install finishes

Install Dependencies

- **Open Terminal**
  - `brew install ffmpeg --with-libvpx --with-libvorbis --with-ffplay`
  - `brew install imagemagick --with-ghostscript --with-webp --with-fontconfig`

        - `brew install exiftool`

Clone SAGE2

- Open Terminal
    - `cd <directory_to_install_SAGE2>`
    - `git clone https://bitbucket.org/sage2/sage2.git`

Generate HTTPS Keys

- Open the file 'GO-mac' inside the '<SAGE2_directory>/keys/' folder
    - Add additional host names for your server in the variable `servers` (optional)
    - Save file
- Open Terminal
    - `cd <SAGE2_directory>/keys`
    - `./GO-mac`
        - enter your password when asked (the keys are added into the system)

Install Node.js Modules

- Open Terminal
    - `cd <SAGE2_directory>`
    - `npm run in` (install pre-compiled binary modules) or `npm install` (compile and install binary modules)

ERRORS

When running `npm run in` and error is given stating: "fatal error: 'libavcodec/avcodec.h' file not found #include <libavcodec/avcodec.h>". Xcode isn't searching /usr/local/ for include files or libraries. Open Terminal and execute the following command: `xcode-select --install`

When starting SAGE2 an error is given regarding a node module that is missing a dependency. If the error is regarding a libavXXX file, check the /usr/local/lib for libav links. Should the folder not have the specified files, this means the brew installation of ffmpeg did not fully complete. Try reinstall ffmpeg with brew. This may require using `brew link` and/or `brew override` commands.

If the following error is produced:

Assertion failed: (!uvio_active(&stream->io_watcher, UVPOLLOUT) ||
!ngx_queue_empty(&stream->write_completed_queue) ||

```
!ngx_queue_empty(&stream->write_queue) || stream->shutdown_req != NULL ||
stream->connect_req != NULL), function uv_read_stop, file ../deps/uv/src/unix/stream.c, line 1329.

Abort trap: 6
```

The current solution **is to update node to the most recent (8.x LTS or 9.x currently).**

Install using fresh Linux installation

If you want to install SAGE2 on a “clean” virtual machine or a new physical machine, we wrote scripts to install all needed dependencies and the SAGE2 source code. At this moment, we support Ubuntu, openSuse, and CentOS, all in 64-bit mode.

Install your favorite Linux distribution (basic desktop configuration, for instance XFCE or Gnome). Then download one of the following scripts and run it as a regular user, with ‘sudo’ ability. The script will ask your password when needed to install packages.

Feel free to run the script one line at a time, if you have specific requirements.

Ubuntu (16.0.4)

    https://bitbucket.org/sage2/sage2/downloads/SAGE2ubuntu-1604.sh

Ubuntu (14.0.4 and up)

    https://bitbucket.org/sage2/sage2/downloads/SAGE2ubuntu.sh

CentOS (7 and up)

    https://bitbucket.org/sage2/sage2/downloads/SAGE2centos7.sh

openSuse (13.x and up)

    https://bitbucket.org/sage2/sage2/downloads/SAGE2opensuse.sh

For Docker, see:

    https://bitbucket.org/sage2/sage2/wiki/Install%20(Docker)

Install Instructions for Docker (Windows, Mac OSX and Linux)

Install Docker

If you have Docker installed jump to "Install SAGE2". Follow the installation instructions for your operating system here: https://docs.docker.com/installation/

Test Docker

Now verify that the installation has worked by downloading the ubuntu image and launching a container:

```
docker run -i -t ubuntu /bin/bash
exit
```

Install SAGE2

Get the latest SAGE2 image from dockerhub

```
docker pull sage2/master
```

Install the data containers

```
docker create -v /sage2/config --name sage2Config sage2/master
docker create -v /sage2/keys --name sage2Keys sage2/master
docker create -v /root/Documents/SAGE2_Media --name sage2Uploads sage2/master
```

Generate ssl keys, using your domain name as an argument to the below command. For example, if you are installing SAGE2 on sage2server.evl.uic.edu, then you need to pass it as _.evl.uic.edu Example:

```
docker run --rm -it --volumes-from sage2Keys sage2/master /sage2/keys/GO-docker
_.evl.uic.edu
```

Edit the configuration file

```
docker run --rm -it --volumes-from sage2Config sage2/master /bin/bash
cd /sage2/config
vi docker-cfg.json
exit
# Minimum configuration is setting the host (line 6).
```

Running SAGE2:

First time you run SAGE2, you need to create a container that specifies how to run SAGE2.
With the command below you create the container and run SAGE2:

```
docker run -d --volumes-from=sage2Config --volumes-from=sage2Keys
--volumes-from=sage2Uploads -p 9090:9090 -p 9292:9292 --name sage2 sage2/master
```

To see if it is running

```
docker ps
```

To see the logs

```
docker logs sage2
```

To stop sage2

```
docker stop sage2
```

To start sage2 (Always use this command when starting SAGE2, unless this is the first time running!)

```
docker start sage2
```

Setting the time

By default the timezone is set to Chicago time. To set the right timezone for your installation, you need to run the follow command while the container is running (For example setting it to Arizona):

```
docker exec -e "CONTAINER_TIMEZONE=US/Arizona" sage2
/sage2/bin/docker_set_timezone.sh
```

Full list of timezone codes: https://en.wikipedia.org/wiki/List_of_tz_database_time_zones

To find your area code, look under column TZ*.

NOTE: The above command is not persistent. Timezone will reset once you stop it. We suggest creating a startup script that first executes the run command (starting sage2 server) and then executes the exec command to set the right timezone.

Updating SAGE2

```
docker pull sage2/master
```

Backing up your data: You only need to backup your sage2Uploads data container.

```
docker export --output="sage2Uploads.tar" sage2Uploads
```

More information about backing up data containers in docker here:

https://docs.docker.com/engine/reference/commandline/export/

Restoring your data:

```
docker import sage2Uploads.tar
```

More information about restoring data containers in docker here:

https://docs.docker.com/engine/reference/commandline/import/

Security

This section describes security features in the SAGE2 environment, addressing frontend and backend aspects. At its core, SAGE2 is a web server enabling collaborative work on large displays. Like any web server, it is susceptible to attacks and, like any web user, a SAGE2 user is vulnerable to security issues. However, SAGE2 controls both the frontend (user experience) and the backend (the web server itself), which somewhat limits the types of attacks.

Introduction

As for any web server within your infrastructure, you must apply the same best practices – network access, user login, firewall, etc. – which are outside the scope of the SAGE2 software.

SAGE2 developers apply best practices to ensure a secure environment:

- SAGE2 is built upon Node.js (https://nodejs.org/), the industry-leading technology for web services (Node.js's package ecosystem is the largest ecosystem of open source libraries in the world). It is used by many companies (Netflix, IBM, Walmart, PayPal, …).
- SAGE2 is always built with the latest and most secure version of Node.js.
- SAGE2 requires administrators to install secure and up-to-date SSL certificates to ensure secure and encrypted communication between the server and the users. All communication between the server and users is encrypted (using HTTPS and Secure Websockets).
- SAGE2 relies on Google Chrome and Electron as display and user interface (UI) frameworks (https://electron.atom.io/). Electron is a framework that creates native applications with web technologies, and is used by companies such as Microsoft, Facebook, Slack, and Docker to deliver desktop applications
- User access to the SAGE2 server can be controlled by a password (simple MD5 hashing). This is just intended as a meeting access token, and does not replace secure authentication to the network.
- Various security features can be enabled to enforce better security (cross-site scripting, clickjacking protection, content type options, content security policy, etc.), which can limit functionality in some cases. Best practices from Apache or NGINX can be applied (see description below).

- External websites can be shown within SAGE2 and are run within ‘sandboxed’ browser instances (i.e., webviews).
- Several companies ran security audits (e.g., using HP Fortify) after installing SAGE2 and reported several CVEs (Common Vulnerabilities and Exposures, https://cve.mitre.org/), which were promptly fixed. Various university campuses with SAGE2 installations (UIC, UCSD) run continuous network audits as well.
- For maximum isolation, a SAGE2 server can be run within an application container (SAGE2 supports Docker). Firewall and port mapping must be configured to enable communication among the server, the displays and the users.

Server Configuration

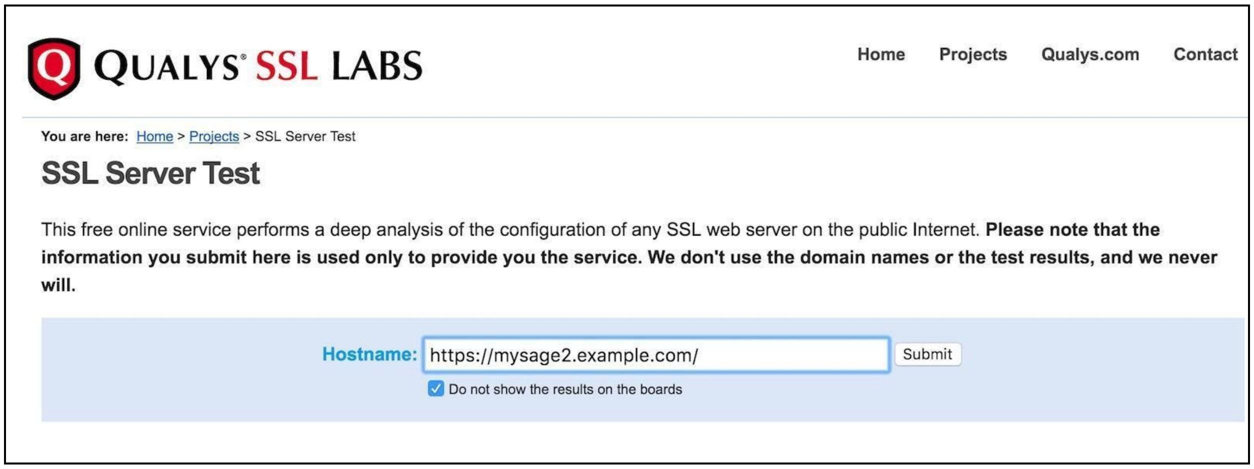
As the SAGE2 server is an HTTP/HTTPS server, it is susceptible to common web attacks. The first step to ensure security is to provide valid SSL certificates as described in this section.

The HTTP and HTTPS ports are the only ports that need to be opened for SAGE2 use: port 80 and 443 for production scenarios. The default ports for user development are 9292 and 9090.

To verify your setup, use “ssllabs.com” to ensure that your certificates are valid, up to date, and using proper cyphers. Put your HTTPS address at the following URL:

https://www.ssllabs.com/ssltest/

Within a few minutes, you should get a report of the validity of your setup. The SAGE2 server should get an A or A+ rating.



You are here: Home > Projects > SSL Server Test

SSL Server Test

This free online service performs a deep analysis of the configuration of any SSL web server on the public Internet. **Please note that the information you submit here is used only to provide you the service. We don't use the domain names or the test results, and we never will.**

Hostname: https://mysage2.example.com/

✓ Do not show the results on the boards

The HTTP header responses from the SAGE2 server can be configured with the following attributes:

Content type of the HTTP response, default value

- "Content-Type" ⇒ "text/html; charset=utf-8";

X-Frame-Options: the header can indicate whether a browser is allowed to render a page within an <iframe> element or not. This is helpful to prevent clickjacking attacks by ensuring your content is not embedded within other sites. For example, values can be "SAMEORIGIN" or "DENY". See more here <https://developer.mozilla.org/en-US/docs/HTTP/X-Frame-Options>.

- “X-Frame-Options" ⇒ "SAMEORIGIN";

X-XSS-Protection: This header enables the Cross-Site Scripting (XSS) filter built into most recent web browsers. It's usually enabled by default, so the role of this header is to re-enable the filter for this particular website if it was disabled by the user.

- "X-XSS-Protection" ⇒ "1; mode=block";

X-Content-Type-Options: The only defined value, "nosniff", prevents Internet Explorer and Google Chrome from MIME-sniffing a response away from the declared content-type. This also applies to Google Chrome when downloading extensions. This reduces exposure to drive-by download attacks and sites serving user uploaded content that, by clever naming, could be treated by MSIE as executable or dynamic HTML files.

- "X-Content-Type-Options" ⇒ "nosniff";

More security features are also available, by setting extra parameters in your SAGE2 configuration file:

```
        security: {
                // The SSL method to use
                secureProtocol: "TLSv1_2_method",
                    // Enable HSTS: HTTP Strict Transport Security - once HTTPS, always
HTTPS
                enableHSTS: true,
                // Enable CSP: Content-Security-Policy
                enableCSP: true
                // Enable HPKP: HTTP Public Key Pinning Extension
                enableHPKP: true,
        },
```

`secureProtocol: String, The SSL method, e.g. SSLv3_method, is used to force a SSL version. The possible values depend on your installation of OpenSSL and version of NodeJS.`

Configuration: security.secureProtocol: string (undefined by default).

- Valid values: TLSv1_2_method, SSLv23_method, ... Default: SSLv23_method (as in node v8.3)

HTTP Strict Transport Security: HSTS is an opt-in security enhancement. Once a supported browser receives this header, the browser will prevent any communications from being sent over HTTP to the specified domain and will instead send all communications over HTTPS.
Configuration: security.enableHSTS: true/false (false by default)

- "Strict-Transport-Security" ⇒ "max-age=31536000";

Content-Security-Policy: Instead of blindly trusting everything that a server delivers, Content-Security-Policy defines the HTTP header that allows you to create a whitelist of sources of trusted content, and instructs the browser to only execute or render resources from those sources. Even if an attacker can find a hole through which to inject script, the script won't match the whitelist, and therefore won't be executed. *default-src 'none'* ➔ default policy that blocks absolutely everything.
Configuration: security.enableCSP: true/false (false by default)

- "Content-Security-Policy" ⇒
  "default-src 'none';
  plugin-types image/svg+xml;
  object-src 'self';
  child-src 'self' blob:;
  connect-src *;
  font-src 'self' fonts.gstatic.com;
  form-action 'self';
  img-src * data:;
  media-src 'self';
  style-src 'self' 'unsafe-inline' fonts.googleapis.com;
  script-src * 'unsafe-eval';"

HTTP PUBLIC KEY PINNING: HPKP Key pinning is a trust-on-first-use (TOFU) mechanism. The first time a browser connects to a host it lacks the information necessary to perform "pin validation", so it is not able to detect and thwart a MITM attack (man in the middle). This feature only allows detection of these kinds of attacks after the first connection.
Configuration: security.enableHPKP: true/false (false by default)

- "Public-Key-Pins" ⇒ 'pin-sha256="Pin1"; pin-sha256="Pin2";max-age=2592000; includeSubDomains'

- The values of the ‘pins’ are generated with the scripts provided in the ‘keys’ folder of SAGE2: GenHPKP.bat for Windows, and GenHPKP.sh for Unix (MacOSX and Linux).
    - The scripts take two parameters: your main CA certificate and the CA certificate for 127.0.0.1 (the second pin is used as a backup).
    - Example: *./GenHPKP.sh myserver-ca.crt 127.0.0.1-ca.crt*
    - This generates two files (pin1.sha256, pin2.sha256) containing hashes/pins for your site. The pins will be added to the HTTP responses, the browser will then compare the values to the ones previously received. If different, the HTTP request will be denied
- If you change or update your site certificates, the pins should be regenerated.

Web Certificates

SAGE2 can generate so-called “self-signed certificates” to support running the secure HTTPS protocol (instead of the insecure common HTTP protocol). This requires SSL certificates to prove the identity of the server to its clients.

Self-signed Certificates

To generate self-signed certificates:

- Windows
    - Edit '<SAGE2_directory>\keys\GO-windows.bat'
        - Add lines with a list of hostnames for your server
        - Save file
    - Double click 'GO-windows.bat'
- MacOSX
    - Open the file 'GO-mac' inside the '<SAGE2_directory>/keys/' folder
        - Add additional host names for your server in the variable servers (optional)
        - Save file
    - Open Terminal
        - cd <SAGE2_directory>/keys
        - ./GO-mac
            - enter your password when asked (the keys are added into the system)
- Linux

  - Open the file 'GO-linux' inside the '<SAGE2_directory>/keys/' folder
    - Add additional host names for your server in the variable servers (optional)
      - for instance, add the short name and the fully qualified domain name of your server
    - Save file
  - In a Terminal
    - cd <SAGE2_directory>/keys
    - ./GO-linux

However, self-signed certificates are only valid for testing and development since they are not recognized by any trust authorities. Modern web browsers reject self-signed certificates and ask the user to approve the security exception. Some mobile browsers ignore completely such certificates.

Signed Certificates

Anyone installing SAGE2 should acquire valid SSL certificates from a reputable source. Most campuses in the USA can get certificates for free (or a modest sum) through their local campus IT department. For instance, see https://www.incommon.org/

> *InCommon, operated by Internet2, provides a secure and privacy-preserving trust fabric for research and higher education, and their partners, in the United States. InCommon's identity management federation serves 9 million end-users. InCommon also operates a related assurance program, and offers certificate and multifactor authentication services.*

Alternatively, web hosting companies usually sell certificates to their clients. A number of security companies also sell host certificates, requiring you to prove that you “own” the server in question.

A requirement for any certificate is that your host has a fully qualified domain name (FQDN): this is the complete domain name for a specific computer, or host, on the Internet. The FQDN consists of two parts: the hostname and the domain name. For example, an FQDN for a hypothetical SAGE2 server might be sage.somecollege.edu.

Host certificates are only valid for a fixed period of time (1 or 3 years, usually) and for a unique machine. Wildcard certificates supports any machine within a network subdomain (for instance,

.my_dept.my_univ.edu) and function the same way as host certificates. Not all certificate authorities generate wildcard certificates and they are generally more expensive.

The process goes as follows:

1. you generate a request and a private key for any given server,
2. you send the request to a certificate authority,
3. you receive a signed certificate
    a. you might also receive a CA (certificate authority) certificate providing information on the chain of authority entities involved in your certificate.
4. your signed certificate and your private key are setup in SAGE2 to provide a valid secure SSL connection with the clients
    a. The CA certificate chain can be provided to speed up the validation process.

There are four files involved:

- A CSR or Certificate Signing request is a block of encrypted text that is generated on the server for which the certificate will be used. It contains information that will be included in your certificate, such as your organization name, common name (domain name), locality, and country. It also contains the public key that will be included in your certificate. The file uses the ".csr" extension and it looks like this:
    - `-----BEGIN CERTIFICATE REQUEST-----`
    - `MIIC4jC...`
    - `…`
    - `-----END CERTIFICATE REQUEST-----`
- A private key is created at the same time that you create the CSR. It usually named with the extension ".key"
    - `-----BEGIN RSA PRIVATE KEY-----`
    - `MIIEp…`
    - `...`
    - `....hw==`
    - `-----END RSA PRIVATE KEY-----`
- The server certificate received from your certificate authority, named with the extension ".cer", is a "X509 Certificate only, Base64 encoded" file:
    - `-----BEGIN CERTIFICATE-----`
    - `MIIF….`
    - `….`
    - `-----END CERTIFICATE-----`

- You might get a CA certificate, or multiple certificates (in case of a hierarchy of certificate authorities), which is a single “X509, Base64 encoded” file.
    - `-----BEGIN CERTIFICATE-----`
    - `MIIF….`
    - `….`
    - `-----END CERTIFICATE-----`
    - `-----BEGIN CERTIFICATE-----`
    - `MIIF….`
    - `….`
    - `-----END CERTIFICATE-----`
    - `-----BEGIN CERTIFICATE-----`
    - `MIIF….`
    - `….`
    - `-----END CERTIFICATE-----`

SAGE2 requires the private key and the signed certificate. Optionally, it will load the CA certificates, if available. Certificates for SAGE2 should be named very precisely in order to be loaded automatically by the server, and stored in the ‘keys’ folder:

- the server private key should be named: [*host*]-server.key
    - where [*host*] is the value associated with the key ‘host’ in your configuration file
    - example: if my server is called *‘traoumad.evl.uic.edu’* then my key file should be:
        - *traoumad.evl.uic.edu-server.key* in the ‘keys’ folder
- the server host certificate should be named: [*host*]-server.crt
    - where [*host*] is the value associated with the key ‘host’ in your configuration file
    - example: if my server is called *‘traoumad.evl.uic.edu’* then my key file should be:
        - *traoumad.evl.uic.edu-server.crt* in the ‘keys’ folder
- optionally, the CA certificates should be named: [*host*]-ca.crt
    - where [*host*] is the value associated with the key ‘host’ in your configuration file
    - example: if my server is called *‘traoumad.evl.uic.edu’* then my key file should be:
        - *traoumad.evl.uic.edu-ca.crt* in the ‘keys’ folder
- for domain certificates:
    - the domain private key should be named with a ‘_’ and then the domain name and the extension “.key”
        - example: for the “evl.uic.edu” domain, the file is named “_.evl.uic.edu.key”
    - the domain certificate should be named with a ‘_’ and then the domain name and the extension “.crt”

        - example: for the “evl.uic.edu” domain, the file is named “_.evl.uic.edu.crt”
    - the CA certificate should be named with a ‘_’ and then the domain name and the extension “-ca.crt”
        - ex: for the “evl.uic.edu” domain, the file is named “_.evl.uic.edu-ca.crt”

You also need the same series of files for each and every hostname that your SAGE2 server uses and is accessed through. This is important for multi-home computers (often the case in clusters, or machines having connections to multiple networks). Those certificates are usually used internally and can be self-signed certificates. Common names are added in your “alternate_hosts” section of your SAGE2 configuration, such as “127.0.0.1” and “localhost”.

For example, the following configuration will need a valid signed certificate for the name “iridium.evl.uic.edu” and self-signed certificates for the following names: “localhost”, “127.0.0.1”, “131.193.183.199”, “iridium.evl.optiputer.net”. These are all valid names to access this unique server from multiple network locations:

```
        name: "Cybercommons",
        host: "iridium.evl.uic.edu",
        port: 443,
        index_port: 80,
...
        alternate_hosts: [
                "localhost",
                "127.0.0.1",
                "131.193.183.199",
                "iridium.evl.optiputer.net"
        ],
...
```

The extra self-signed certificates can be generated by the commands described in the beginning of the section (scripts in the ‘keys’ folder).

Let’s Encrypt

A third way is emerging to generate valid certificates and increase the use of valid certificates on the Internet. Here is an excerpt from “https://letsencrypt.org/about/”:

Let's Encrypt is a free, automated, and open certificate authority (CA), run for the public's benefit. Let's Encrypt is a service provided by the Internet Security Research Group (ISRG).

The key principles behind *Let's Encrypt* are:

- Free: Anyone who owns a domain name can use *Let's Encrypt* to obtain a trusted certificate at zero cost.
- Automatic: Software running on a web server can interact with *Let's Encrypt* to painlessly obtain a certificate, securely configure it for use, and automatically take care of renewal.
- Secure: *Let's Encrypt* serves as a platform for advancing TLS security best practices, both on the CA side and by helping site operators properly secure their servers.
- Transparent: All certificates issued or revoked will be publicly recorded and available for anyone to inspect.
- Open: The automatic issuance and renewal protocol will be published as an open standard that others can adopt.
- Cooperative: Much like the underlying Internet protocols themselves, *Let's Encrypt* is a joint effort to benefit the community, beyond the control of any one organization

The process is still fairly manual and mostly described for the Unix operating system but it works and can generate valid signed certificates for use in SAGE2.

The process is fairly automated for existing mainstream HTTP servers (Apache, Nginx, …), but requires some automation for SAGE2. This is work-in-progress and we hope to support it in the future. The process involves having the HTTP server running and providing a generated file to the 'letsencypt.org' server to prove the server identity. The CSR request is then generated and signed automatically to provide a valid certificate. This would automatically provide a free certificate for anyone with a server and a FQDN hostname.

Audits

UCSD Campus network

From ___@eng.ucsd.edu:

1 - “Campus did a Qualsys scan of my SAGE2 install, and it looks like it is susceptible to directory traversal. Also, I have a public interface that is configured, but it is not in the SAGE2 configuration, yet it is still responding via IP address.”

2 - The IP address is not listed as an alternate host, and is there solely to allow updating of SAGE2 via npm since I could not get npm to honor the proxy config. Not a big deal, as long as the public facing interface is only there for software updates/patching.

I'll see if I can get a specific CVE, but here is the detailed output from the scan minus the /etc/passwd file contents. Note, this interface should not be responding on 9090 at all, and I use password protection on the site as well:

```
GET /..//..//..//..//..//..//..//..//..//etc/passwd HTTP/1.1
Host: [REDACTED]
Connection: Keep-Alive


HTTP/1.1 200 OK
Content-Type:
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Access-Control-Allow-Headers: Range
Access-Control-Expose-Headers: Accept-Ranges, Content-Encoding, Content-Length, Content-Range
Cache-Control: no-cache, no-store, must-revalidate
Pragma: no-cache
Expires: 0
Content-Length: 2147
Date: Fri, 02 Jun 2017 05:31:45 GMT
Connection: keep-alive
```

Air France – KLM / Group Technology Office

From: ___@klm.com

Find attached a security scan as performed by HP Fortify. As you may want to have a look first, I did not include your developers. You are welcome to forward them to your team member as you see fit.

Noted as most important findings:

- *Webserver within the Docker container: directory traversal should be configured off, e.g. to prevent access to the /etc/password file.*
- *JavaScript should prevent Cross Side Scripting (XSS) – the report shows code samples / recommendations.*
- *Weakness introduced by Self Signed Certificates: there are SSL code suites that are not safe anymore.*

There are more details and recommendations in the report. We are happy to discuss them with you and your development team. As the report already contains many details, please let us know if / when you like me to arrange such a call.

[….] However we can only proceed if the security vulnerabilities have been addressed, a standard procedure in admitting new applications.

NASA's Goddard Space Flight Center

___@nasa.gov
NASA Center for Climate Simulation
Goddard Space Flight Center

“Our security team has done a preliminary review of SAGE2 and I thought you’d be interested in the findings (see attached). If a conversation would be useful, let me know. We remain very interested in trying to get our proxy approach to work.”