

EMI

GridSite 1.x

USER GUIDE

Document identifier: **emi-gridsite-user-doc-1.0.0-1**

Document version: **1.0.0-1**

Date: **April 18, 2011**

Document status: **FINAL**

Document link: **<http://egee.cesnet.cz/cvsweb/SEC/GridSite.pdf>**

Abstract: In its simplest application, GridSite uses X.509 certificates loaded into unmodified versions of web browsers like Internet Explorer, Netscape or Mozilla to authenticate users, and then grants read and write authorization on this basis. HTML and text files can be edited within a browser window, or pages and binary files can be uploaded from local disk. Certificate based authentication of users is now far more practical with the start of large scale issuing of X.509 certificates within Grid projects.

GridSite architecture

The current 1.x series is a substantial rewrite of GridSite, consisting of mod_gridsite, gridsite-admin.cgi, a toolkit libgridsite and the htcp command-line HTTP(S) client. Sources of the development versions are available from the CVS Gridsite area, and are published under the Modified BSD License. Sources are available in the GridSite download area.

mod_gridsite is a loadable module for the Apache web server which provides access control and page formatting for GridSite HTTP(S) Fileservers, Websites and Web Services hosts. mod_gridsite also intercepts some processing in the standard mod_ssl module to support GSI Proxies and VOMS attribute certificates, as well as the normal X.509 client certificates. The verification of these credentials is handled by functions within mod_gridsite and the main GridSite shared library, without the need to patch or rebuild mod_ssl.

CONTENTS

USER GUIDE	3
READING FROM HTTP AND HTTPS SERVERS	3
AUTHENTICATING	3
AUTHORIZATION	4
MANAGING DIRECTORIES AND FILES	4
HTML FORMATTING IN GRIDSITE	4
COMMAND LINE USE	5

USER GUIDE

This Guide is intended for people using [GridSite](#) websites with conventional web browsers, especially people with write access to areas of the site. There is a separate [Administration Guide](#) with additional information for people managing access control and group membership. This Guide assumes you are familiar with basic Web and HTML concepts. Towards the end we discuss how to access servers with command line tools like curl and httpc.

READING FROM HTTP AND HTTPS SERVERS

GridSite servers are usually accessible both via HTTP and via HTTPS. You can always tell which version you are using by looking at whether the URL in your browser's location window starts with "http://" or "https://". HTTPS means that the connection to the server is encrypted, that you can verify you're talking to the real server and not an imposter, and gives you the option to authenticate to the site and perhaps gain write access.

Simple browsing of the website via HTTP or HTTPS is reasonably self-explanatory. If configured, additional links may appear in the footer of each webpage with links to this help, and to switch between HTTP and HTTPS versions of the page. Pages may also have a link to the page History, showing the dates of changes to that page and names of its authors.

When looking at HTTPS pages, you may find your browser reports it cannot verify the server's certificate since it does not recognise the Certification Authority (CA) it uses. You should attempt to load the CA's root certificate into your browser to stop these warnings. (This means your browser will be able to identify any servers using fake certificates which you shouldn't trust.) How you obtain the CA Root Certificate from a trust-worthy source depends on the CA. For example, the UK e-Science CA lets you download it [from their website](#) (<http://ca.grid-support.ac.uk/>) and others can be obtained from the [EU Grid PMA](#).

AUTHENTICATING

To go beyond reading pages you need to obtain a user certificate and load it into your web browser. How you do this again depends on the Certification Authority you have access to (for most Grid projects, CAs are organised on a national basis.) To use the UK e-Science CA example again, their website has links to the procedure for applying for a certificate from within a web browser.

A user certificate usually has a version of your name and affiliation as its Distinguished Name (DN) - for example, "/C=UK/O=eScience/OU=Manchester/L=HEP/CN=Andrew McNab"

Once you've obtained a user certificate in your name from your CA, you need to make sure it is loaded into the browser you normally use to browse the web. How you do this is different for different browsers and to some extent for different CAs (but if you applied for the CA through your browser, you may already have it there.)

Browsers want the certificate and private key in the PKCS#12 format, which is normally a single file with the extension ".p12". Many programs which are based on OpenSSL, such as Globus and curl, prefer the PEM (".pem") format for certificates, with separate certificate and key files ("usercert.pem" and "userkey.pem", for example.) If you only have the files in .pem format and have access to openssl, you can use its command line tools to convert PEM to PKCS#12:

```
openssl pkcs12 -in usercert.pem -inkey userkey.pem -export -out certkey.p12
```

Be very careful not to accidentally overwrite .pem or .p12 files when doing this kind of thing! In particular, if you lose your private key, you cannot retrieve it from your CA.

Once your user certificate is loaded, you should be able to see your certificate name appear when you look at an HTTPS GridSite page which has the page footers enabled - for example, the "Switch to HTTP" link present. If GridSite understands your user certificate, it displays a "You are ..." line in the footer. (However, the Apache webserver must also be set up with your CAs root certificate for this to work. The GridPP HTTPS home page is set up to recognise a good range of European and North American Grid CAs.)

AUTHORIZATION

Once users can prove their identity to the web server, it then becomes possible to give them appropriate rights depending on that identity. GridSite allows site administrators to specify these rights for individuals and groups using [GACL](#) access control files. (The [Administration Guide](#) explains how to manage these files.) GACL defines who can read files, who can list directories, who can write or create files and who can modify the GACL policy files. To get increased access to an area of a site, you need to contact the administrator for that area and give the DN of your certificate (it's not necessary to send any certificate files.)

MANAGING DIRECTORIES AND FILES

If you have list permission for the directory containing a page, you should see an extra link "Manage Directory" in the page's set of footer links, which allows you to browse the directory even if the normal index.html is present. If page histories are available, this listing view also has links to them.

The real power of GridSite becomes available if you have write access to a directory. In that case, the "Manage Directory" page has additional links to Delete or Rename pages and other files, and to Edit HTML and plain text files. An Edit link also appears in the footer links of HTML pages.

If you use the Edit function, you are presented with an HTML form containing the current filename and the full HTML or plain text of the page for you to edit. This allows you to maintain the content of the site "in place" and to see the result of your changes immediately, in context.

If you modify the filename in the form before saving, GridSite will make a new file with that name, and the old file will still be present, unmodified. (However, you cannot use this feature for creating a file in a different directory.) As you make changes, the history of the changes and your certificate DN are recorded, and available in the history page for that file.

For people with write access, the "Manage Directory" page also has options to upload a file from the computer your browser is running on, and to create files and directories. If it's enabled, you can also view the contents of WinZIP / PKZIP / .zip files, and unpack their contents into the current directory. (This feature is very useful if you have several files to upload at one time.)

HTML FORMATTING IN GRIDSITE

As well as providing access control and file management, GridSite provides some simple formatting of HTML pages by adding standard headers and footers. (If this isn't sufficient, GridSite will happily coexist with HTML preprocessor languages like SSI, PHP and JSP.)

If HTML formatting is enabled for the current directory, GridSite looks for the files gridsitehead.txt and gridsitefoot.txt in that directory, or goes up through the parent directories until they are found.

The <body> and </body> tags from the HTML file are replaced with the contents of the gridsitehead.txt and gridsitefoot.txt files, which should normally be chunks of HTML including a replacement <body> or

</body> tag. If either tag is absent from the original page, then the header or footer is just added rather than being inserted in place of the tag. (One consequence of this absence is that HTML header tags like <title> can end up after a <body> tag, and can get ignored by browsers - so always include <body> ... </body> in your pages.)

This simple system is suprisingly flexible, and allows a variety of top and bottom, or sidebar navigation layouts of pages. Since the <body ...> tag is under full control of the author of the gridsitehead.txt file, backgrounds, colour schemes and style sheets can easily be specified.

For example:

Source	HTML
page.html	<title>PAGE TITLE</title>
page.html (replaced)	<body>
gridsitehead.txt	<body text=blue> Heading text <table border=1> <tr> <td>Standard sidebar</td> <td>
page.html	<p> Page content...
page.html (replaced)	</body>
gridsitefoot.txt	</td> </tr> </table> Footer text </body>

produces pages with a layout like:

Heading text	
Standard sidebar	Page content...
Footer text	

COMMAND LINE USE

GridSite adds support for the HTTP PUT and DELETE methods, and this makes it easy to create or delete files from within programs and commands without using a web browser and HTML forms. It is straightforward, although slightly awkward, to use a standard HTTPS-aware client like curl to upload files, but GridSite provides httpc as a more convenient client program, which is easier to use with GSI Proxies and X.509 user certificates, and has a syntax closer to the familiar scp command.

The following examples assume the GridSite server has GSI support and use a GSI proxy as the client certificate. For non-GSI use, just skip the grid-proxy-init stage, and replace the proxy filename with \$HOME/.globus/usercert.pem and \$HOME/.globus/userkey.pem (or wherever your PEM format certificate and key are stored.)

First generate a GSI proxy with `grid-proxy-init`. This will create a proxy file in `/tmp/x509up_uXXXXX` where `XXXXX` is your Unix UID (also given by `id -u`.) The GSI proxy contains a temporary private key and certificate signed by your long-term user certificate.

You should make sure you have a copy of the CA root certificates of the CA's used by the servers you wish to talk to. These are usually installed in `/etc/grid-security/certificates` as files like `01621954.0`, and RPMs and tar files for many common European and North American CAs are available from the [EU Grid PMA](#).

To upload a file with `curl`:

```
curl --cert /tmp/x509up_u`id -n` --key /tmp/x509up_u`id -n` \  
      --capath /etc/grid-security/certificates \  
      --upload-file /tmp/new.file.txt https://server/new.file.txt
```

The equivalent `htcp` command is:

```
htcp /tmp/new.file.txt https://server/new.file.txt
```

since `htcp` looks for the GSI proxy and CA certificates automatically. `htcp` can also be used to copy remote files to the local machine by reversing the arguments. For more details, see the `htcp(1)` man page.

`htcp` also has options for deleting files, and doing short or long listings, and these can also be accessed using the `htrm`, `htls` and `htll` commands (which are normally symbolic links to `htcp`.)

Directory indexes are based on parsing the index returned by the web server and by using the HTTP HEAD method to obtain the file size and modification times.

All of the `ht**` commands can accept multiple source file arguments, and this allows you to copy multiple files to or from the server. Shell wildcard expansion on the local machine is especially useful:

```
htcp /tmp/new.*.txt https://server/
```