

Generic Report & Document Repository Intranet v1 Installation Guide

| Version | Author | Date | Software Covered |
|---------|---------------|----------------|------------------|
| 1.0 | Simon Edwards | September 2012 | 0.0.1+ |
| 1.1 | Simon Edwards | October 2012 | 0.0.2+ |
| 1.2 | Simon Edwards | December 2012 | 0.1.0+ |

Table of Contents

| | |
|---|-----------|
| INTRODUCTION | 4 |
| Software Dependencies | 4 |
| External Dependencies | 6 |
| Recommended Installation Approach | 6 |
| BACKEND INSTALL | 8 |
| HP-UX Post Installation Steps..... | 9 |
| AIX Post Installation Steps | 9 |
| Linux Post Installation Steps | 9 |
| Solaris Post Installation Steps | 9 |
| Windows Installation Steps | 10 |
| Back-end Authentication Service Configuration | 11 |
| FRONTEND INSTALL | 14 |
| Front-end Configuration Files..... | 15 |
| Roles File Configuration | 15 |
| Globals File Configuration | 16 |
| The “Ports” Section | 17 |
| The “Variables” Section | 17 |
| The “Email_addresses” Section..... | 18 |
| The “Directories” Section..... | 19 |
| The “tmp_content” Section | 20 |
| Macro Handling Configuration File | 21 |
| LDAP Group Integration..... | 22 |
| LDAP Daemon Requirements | 22 |
| LDAP Daemon Response formats | 23 |
| Temporary Directories | 24 |
| Scheduled Tasks | 25 |
| Style Sheet Creation / Guidelines | 26 |
| Preferences Defaults / Settings | 27 |
| Custom Authentication Interface..... | 31 |
| APPENDIX A: EXAMPLE APACHE 2 VIRTUAL SERVER CONFIG | 34 |

APPENDIX B: WINDOWS GRI BACKEND SERVICE..... 35

Introduction

It is impossible to document everything; and so if you do come across any issues or queries please contact the author using the following email address:

simon.edwards@advantsys.co.uk

The GRI software is available as two parts;

- Backend - this must be present on any server that has either reports to present on the intranet site, or perform UNIX user authentication (or both).
- Frontend - The actual intranet site - installed on the host presenting the site.

Each package is installed into a different directory to reduce confusion (especially considering in many cases the frontend and backend packages are likely to be installed on the same machine).

The Frontend (where the web server will be running), is installed into:

```
/opt/gri
```

The backend (which can be installed on the same server as the frontend, or other servers where reports are present), is installed into:

```
/opt/gri_backend
```

☆ All steps in this guide should be carried out as "root" on the appropriate servers (if UNIX) or by a user with administrator privileges on Windows platforms.

Software Dependencies

The GRI environment is mainly written in Perl and although care has been taken to use standard libraries and modules; the following might require installation on the servers where either the frontend and/or backend are installed:

- Digest::SHA1
- XML::Parser
- XML::Simple
- IO::Handle
- MIME::Base64
- Cwd
- CGI
- DBD
- DBI::SQLite
- Authen::Simple::PAM (for "Linux" environments)

Communication Configuration Requirements

GRI is able to work with either insecure client/server communication, or secure communication. Obviously the insecure configuration is more straightforward and has less network overhead - but is only really suitable for testing environments or those with secure networks. To provide secure communication the following additional Perl module will be necessary on all clients and servers:

Crypt::CBC
Crypt::Twofish
Crypt::RSA or Crypt::OpenSSL::RSA

It should be noted that both "Crypt::RSA" and "Crypt::OpenSSL::RSA" are supported but due to lack of interoperability one or the other should be picked for all machines.

External Dependencies

The software can work with compressed and gzip compressed files, and thus the following programs must exist if this facility is to be made use of:

- zcat
- gunzip

The above programs are expected to exist in any of the following directories which are searched for them:

- /usr/bin
- /bin
- /usr/local/bin
- /opt/oss/bin
- /usr/contrib/bin

If secure client/server communication is to be used and the module chosen for RSA encryption is "Crypt::OpenSSL::RSA" the relevant OpenSSL libraries will need to be installed too.

Recommended Installation Approach

The software products can be installed in any order but the easiest approach tends to be as shown below.

1. Determine front-end web server to use
The server which runs the web services needs to be defined. This server must run a compatible version of Perl with the modules highlighted previously. An appendix is included in this document which describes the typical configuration for Apache Web services, though other web services should work as well.
2. Install a backend package on Web Server
The package should then be installed. The package is available as a tar-ball of several different package types, including the "TP2" package type which should allow package installations in most environments (after "TP2" itself is installed of course).
3. Install front-end package on Web Server
Next the front-end package should be installed. Both the front-end and back-end can be installed without clashing on the same host since a single host can store reports and present them via the intranet front-end.
4. Optional: Create and Install Initial Public/Private Keys
This step is necessary if you wish to use the secure communication for all client/server communication within the Generic Reporting Intranet product.
5. Configure/Start Authentication Daemon
The authentication daemon is responsible for checking whether a particular user's access/password is valid. It typically is configured to run on the web-server but can run on any other server which hosts the UNIX accounts.

Even if single-sign-on facilities are required it make sense to configure this now since it still needs to be run to enable user management. LDAP user facilities will be available in later releases.

6. Optional: Install/Configure backend package on other servers
At any time additional copies of the back-end software can be installed on other servers where reports are present and need to be made visible via the GRI intranet software.

Backend Install

The backend package is installed into the following location:

```
/opt/gri_backend
```

The package space requirements are small; less than 1Mb. The software is available in native package formats or as a TP2 package. To install as a TP2 package it must be installed in the "root" namespace as follows:

```
# tp2 install --namespace root --verbose --pkg gri_backend
```

Obviously you need to be the "root" user to perform the above installation. Once installed post configuration is necessary to ensure the necessary daemons are started.

On all servers a configuration file determines which daemons get started. The configuration file defines the following variables:

| Variable | Purpose |
|------------------------------------|---|
| GRI_AUTHD | Set to 0 not to start the authentication daemon, 1 to start it (typically 0). |
| GRI_AUTHD_CONFIG | Indicates the name of a configuration file to use. This is optional and is the name of an XML file defining the users the "authd" process should return when listing users. |
| GRI_BACKEND | Set to 0 not to start the backend daemon, 1 to start it (typically 1). |
| GRI_BACKEND_SECURE | Set this to "1" if you wish to make use of RSA encryption for communication between the backend server and the web services layer. |
| GRI_SECURE_WRITER | Set to 0 not to start the secure writer daemon, 1 to start it (typically 0). |
| GRI_SECURE_WRITER_DIRECTORY | Set to the name of the root owned directory where the secure writer writes files to. |

This configuration file is available as "/opt/gri_backend/rc.config.d/gri_backend.example". It may need to be copied and modified depending on the OS type on which it has been installed.

If we assume the web server is also going to act as the authentication server, then the contents of this file should be modified to be:

```
GRI_AUTHD=1
GRI_BACKEND=1
GRI_BACKEND_SECURE=0
GRI_SECURE_WRITER=0
GRI_SECURE_WRITER_DIRECTORY="/data/reports"
```

HP-UX Post Installation Steps

The following file on this OS determines which daemons are started when the server boots:

```
/etc/rc.config.d/gri_backend
```

Set the variables as described previously. To start the necessary daemons without waiting for a reboot then run the following commands:

```
# /sbin/rc3.d/S900gri_authd start
# /sbin/rc3.d/S900gri_backend start
# /sbin/rc3.d/S900gri_dwritter start
```

AIX Post Installation Steps

Here the configuration file to alter to define which daemons are started is:

```
/etc/gri_backend
```

To start the daemons without waiting for a reboot run the following:

```
# /opt/gri_backend/bin/start-gri_backend.aix start
```

Linux Post Installation Steps

Unfortunately defining a single set of instructions for Linux is not possible since various distributions use different methods of set-up.

If you have a file "/etc/rc.local" then the following can be added to it:

```
/opt/gri_backend/init.d/gri_backend
/opt/gri_backend/init.d/gri_authd
```

Make sure that the above script is executable by running:

```
# chmod +x /opt/gri_backend/init.d/gri_backend
```

Also set-up the configuration file:

```
# cp /opt/gri_backend/rc.config.d/gri_backend.example /etc/gri_backend.conf
```

At this point the daemons without requiring a reboot issue the commands:

```
# /opt/gri_backend/init.d/gri_backend
# /opt/gri_backend/init.d/gri_authd
```

Solaris Post Installation Steps

To be added

Windows Installation Steps

To be added.

Secure Backend Service Configuration

If use of RSA encryption is required then the configuration file used by the process will need to be modified. The setting that needs changing is "GRI_BACKEND_SECURE":

```
GRI_BACKEND_SECURE=1
```

It will be necessary on each server the backend is installed on to ensure that this setting is replicated. GRI does not supported a mixed secure/insecure environment - if secure backend processing is required (and it is recommended) then it must be set to secure on all servers that run the backend process.

It is also necessary to ensure that a directory is created to contain the public key of the web servers that are to communicate with the backend process. On the backend server run the following to create the directory if needed:

```
# mkdir /opt/gri/backend-pubkeys
```

It is now necessary to create a key-pair for this backend server by running the following command:

```
# cd /opt/gri_backend/bin
# ./rsa_makekeypair
# ./rsa_makekeypair --email backend@$(hostname) \
  --comment "backend on $(hostname)" --ident backend/$(hostname) \
  --file /opt/gri_backend/backend-pubkeys/backend-$(hostname) --bits 1024
```

Once this has been created the following files will exist on the backend server, (using "lubuntu1" as the demo hostname):

```
/opt/gri/backend-pubkeys/backend-lubuntu1.private
/opt/gri/backend-pubkeys/backend-lubuntu1.public
```

The "private" file will be copied to the web server as part of the frontend installation process, but that can be left for now.

If the backend service is already running it should be stopped (this step is dependent on OS system or distribution), using Linux for example:

```
# /etc/init.d/gri_backend stop
# /etc/init.d/gri_backend start
```

★ Please ensure that once the frontend is installed (web server host) that the section indicating the steps necessary for secure backend communication are followed.

Back-end Authentication Service Configuration

Insecure Configuration

This section covers the "insecure" configuration - which is the default. In this instance the communication between the client (which is the web server) and the server (the host running the authentication daemon) occurs in plain text. Such a configuration has much less overheads than the secure configuration and is well suited if the web server and authentication server are the same host - particularly if communication to the web-server is properly firewalled from outside connections.

The "authentication" daemon is responsible not just for checking and authenticating user accounts, but also for providing lists of users for the interface to function. Since not all user accounts on the machine should be visible a configuration file must be used to determine which accounts from the host should be considered relevant.

This configuration file is called the "gri_authd_config.xml" and has the following path:

```
/opt/gri_backend/gri_authd_config.xml
```

When the software is installed this file will not be present, though the example version can be used as a basis for the initial configuration:

```
# cd /opt/gri_backend
# cp gri_authd_config.xml.example gri_authd_config.xml
```

The default settings if a configuration file is not available are equivalent to a configuration file containing:

```
<?xml version="1.0" standalone="yes"?>
  <authd protocol='insecure' pubkeys="/opt/gri_backend/authd-
pubkeys">11223</authd>
  <method>
    <minuid>1000</minuid>
    <gecos_field>1</gecos_field>
    <gecos_field_sep>,</gecos_field_sep>
    <verify>minwords: 2</verify>
    <format>fornames,surname</format>
  </method>
```

The above indicates that any UID above 1000 should be used to list users that may have access. It also indicates that the first Gecos field (field "0") contains the username; that fields in the Gecos data are comma-separated and are in the format "joe bloggs".

The "method" entries can be repeated multiple times and all are tried against all usernames to ascertain whether to consider a user being valid or not.

Hence another "method" element might be:

```
<method>
  <minuid>1000</minuid>
  <maxuid>999999</maxuid>
  <gecos_field>1</gecos_field>
  <gecos_field_sep>\\</gecos_field_sep>
  <verify>minwords: 2</verify>
  <uname_format>^z.*</uname_format>
  <format>surname,forenames</format>
</method>
```

★ The "uname_format" is useful - above it will only list users that have accounts beginning with the letter "z".

Secure Communication Installation

The "authd" section has one small change only to the actual configuration file - in the "authd" element:

```
<authd protocol='secure' pubkeys="/opt/gri_backend/authd-pubkeys">11223</authd>
```

The "protocol" setting of "secure" ensures that the daemon process will only expect secure requests to be sent to it and reject any others.

Of course the Web server also needs to know to start secure communication - it acts as the "client" in such communications. To do this you will need to alter the following file once the front-end has been installed (see next main section):

```
/opt/gri/globals_config.xml
```

This will need to be copied from an example one for a new installation, for example:

```
$ cd /opt/gri
$ cp globals_config.xml.example globals_config.xml
$ chmod 666 globals_config.xml
```

The "ports" section - usually situated near the top of this file will need a line for the "authd" and for secure communication it should look like the following:

```
<authd protocol='secure' host="localhost" description="Localhost">11223</authd>
```

Before communication can take place a public and private key pair need to be generated and the files placed in the correct locations. Firstly on the authd server do the following:

```
$ cd /opt/gri/bin
$ ./rsa_makekeypair --email gri@ogri --comment "None" \
  --ident gri/$(hostname) --file ~/auth_rsa --bits 1024 --overwrite
```

In the above example a key length of 1024 is used; ideally it needs to be at least 2048 bits in length.

The "--ident" settings should be "gri/" followed by the hostname of the server running the authentication service.

Depending on the platform this may take some time to run, but when complete will have generated two files:

```
~/auth_rsa.public  
~/auth_rsa.private
```

On the authentication host a directory for the must be created to contain all public keys of the "clients" (i.e. just the web server usually), and then the first public key copied into it:

```
$ mkdir /opt/gri/authd-pubkeys  
$ chmod 700 /opt/gri/authd-pubkeys  
$ chown 0:0 /opt/gri/authd-pubkeys
```

★ This directory can be modified via use of the command line option "--pubkeys" - which should have the alternative directory as an argument. This directory must exist if specified.

```
$ cp ~/auth_rsa.public /opt/gri/authd-pubkeys
```

Although this completes the configuration for the public key, on the web server - which might be this host - please ensure the steps shown in the "Secure Authentication Configuration" section on page 20 are followed too.

Automatic Daemon Start-up

Once the configuration file is in place you should ensure that the daemon is started when the host in question is restarted. There are various ways for this to happen depending on the UNIX variant or the particular distribution of Linux or BSD.

However whichever the environment typically the command that should be executed tends to be:

```
/opt/gri_backend/bin/gri_authd --config /opt/gri_backend/gri_authd_config.xml
```

Frontend Install

The frontend package is only necessary on the server that presents the website. It installs software into the following directory:

```
/opt/gri
```

- ☆ Since this software interacts with the web server, the ownerships and permissions of the files must be handled carefully. Hence in following commands the user "www_owner" should be replaced by the user ID used for the web services - this might be "nobody", "www", "www-data", "httpd" or anything else; please check your configuration!

Although larger than the backend package, it is still small; taking less than 10Mb. Over time this will grow because although the actual reports are not kept on the web server, the following data is:

- *User Preferences* - users are able to change limited settings from the front-end, each gets saved as a separate configuration file (1-2Kb per user).
- *User Alerting* - If a user has defined alerts a small file (one per user) keeps information on which files they want alerting on and another the status of file alerts. Again these are typically 1-2Kb per user.
- *Report Definitions* - For every report type defined a small XML file is generated. Again this is likely to be less and 1Kb in size.
- *Portal Definitions* - As with report definitions each portal requires a small XML file to define it. Typically portal sizes range from 1Kb to 10Kb at a maximum.
- *Report / Image caching* - If a report to display is HTML then typically it is cached locally on the web server to allow links and frames to work correctly. All images referenced will be copied to the web server to the html modified in the cache to reference them.

Taking the above factors into account the storage requirements for the front-end, ignoring caching will probably a maximum of 20Mb of space. The space allocated to caching depends on the number of users, the type and size of reports, and whether they contain images or not.

- ☆ The location of the caching can be modified to sit in a different directory or file system compared to the rest of the product which should be strongly considered for larger deployments.

The package is available in several native packaging formats and also as a TP2 package, which can be installed using the following command:

TP2 Package Installation

```
# tp2 install --namespace root --verbose --pkg gri_frontend
```

Installation instructions for OS-native packages are not specified due to the number of different support options; TP2 packages are strongly encouraged to be used due to the cross-platform nature.

Front-end Configuration Files

When the front-end package is installed the following example configuration files are put into place:

- `gri_macro_handler.xml.example` - settings which determine some of the HTML macro handling used by the tool.
- `globals_config.xml.example` - Settings which define the web server overall settings, including any security features.
- `gri_authd_config.xml.example` - User authentication configuration set-up.
- `gri_roles_config.xml.example` - Available roles configuration.

Notice that these are example files - they will not overwrite the true configuration files (which is useful when you upgrade the software!).

Roles File Configuration

The GRI software offers several ways of allowing the site and all content to be managed effectively. It does so by providing the concept of roles - each role has certain access rights. The standard roles configuration is not part of the basic install, but an example (which is most likely suitable for more environments) is available:

```
/opt/gri/gri_roles_config.xml.example
```

The example configuration (shown below) is usually suitable:

```
<?xml version="1.0" standalone="yes"?>
<gri_roles_config>
  <role name="user_admin" title="User Administration"
    description="Allows addition, modification and deletion of users."/>
  <role name="report_admin" title="Report Administration"
    description="Allows addition and deletion of reports"/>
  <role name="portal_admin" title="Portal Administration"
    description="Allows portals to be added, removed and modified."/>
  <role name="docrepos_admin" title="Document Repository Administration"
    description="Allows complete document repository administration."/>
  <role name="access_admin" title="Portal Access Management Administration"
    description="Allows viewing of portal requests and status of all requests."/>
</gri_roles_config>
```

Hence to use this as the template use the following commands to copy it:

```
# cd /opt/gri
# cp gri_roles_config.xml.example gri_roles_config.xml
# chown www_owner gri_roles_config.xml
# chmod 600 gri_roles_config.xml
```

Each user of the software is able to belong to zero or more roles, providing a fine-grained access control for site administration. It is recommended that one user be given all roles; but that user should not regularly use that account; only use it during site administration.

The "Globals" File Configuration

The file "globals_config.xml" is the most important configuration file and typically resides in the following location on the web server:

```
/opt/gri/globals_config.xml
```

This configuration file contains information about the site in general rather than how the HTML is processed (see the next section for that). There are a number of settings in this file, many of which define the overall look and workings of the site.

The configuration file also determinates the communication protocols and hosts/ports when the web server needs to authenticate access and also communicate with other remote servers.

```
<?xml version="1.0" standalone="yes"?>
<gri_global_config>
  <ports>
    <authd protocol="insecure" host="localhost"
      description="Localhost">11223</authd>
    <backend>11222</backend>
  </ports>
  <variables>
    <variable name="AUTH_HOST" value="GRI test machine"/>
    <variable name="SUPPORT_EMAIL" value="gri@localhost"/>
    <variable name="TITLE" value="Test GRI"/>
    <variable name="PROTOCOL" value="http"/>
    <variable name="URLPREFIX" value="mycomputer/gri">
    <variable name="COMPANY" value="My company"/>
    <variable name="GRI_DESC" value="Your intranet site"/>
    <variable name="HOST" value="localhost"/>
    <variable name="MAX_UPLOAD_SIZE" value="20"/>
    <variable name="MIME_FILE" value="/opt/gri/types.txt"/>
    <variable name="STYLE_SHEET" value="/stylesheet-example.css"/>
    <variable name="LEFT_HEADER_LOGO" value="your_logo.jpg"/>
    <variable name="MIDDLE_HEADER_LOGO" value="your_middle.gif"/>
    <variable name="RIGHT_HEADER_LOGO" value="your_logo2.gif"/>
  </variables>
  <email_addresses>
    <access_requests>gri@localhost</access_requests>
  </email_addresses>
  <directories>
    <auditdir>/opt/gri/logs</auditdir>
    <portals>/opt/gri/portals</portals>
    <templates>/opt/gri/templates</templates>
    <macros>/opt/gri/macros</macros>
    <reports>/opt/gri/reports</reports>
    <alerts>/opt/gri/alerts</reports>
    <users>/opt/gri/users</users>
    <database>/opt/gri/database</database>
    <docrepos>/opt/gri/database</docrepos>
  </directories>
  <tmp_content localdir="/opt/gri/reports/tmp"
    urlref="gri/reports/tmp"/>
</gri_global_config>
```

To take the example and use it (and alter it as appropriate) the commands such as the following should be used:

```
# cd /opt/gri
# cp globals_config.xml.example globals_config.xml
# chown www_owner globals_config.xml
# chmod 600 globals_config.xml
```

The “Ports” Section

The defines which host is running the authentication daemon for the site and what port it listens on. Hence the “authd” element has the attributes “host” and “description” which should be populated. The value is the actual port to connect to on that host.

The “backend” indicates the port on which all the backend-daemons listen on. This means that all backend daemons must currently use the same port.

The “protocol” element is important; it can be set to “insecure” (the default) to ensure the communication is clear-text, or “secure” to make use of RSA encryption.

The “Variables” Section

The elements here are useful to define several aspects of the site work; including graphics; links and some text on key pages. Each variable can be changed as necessary:

| Variable | Purpose |
|------------------------|---|
| AUTH_HOST | This text is used on the login screen to indicate to the user where the authentication information is taken from. |
| SUPPORT_EMAIL | The email address to use on the login page to allow users to send queries to for the site administrators. |
| TITLE | The title of the site on the pages that are generated. |
| PROTOCOL | Defines whether the software should generate links on “http” or “https” (which are the two available supported settings). |
| URLPREFIX | This is typically the fully qualified name of the web server - for example “wbserv1.mycompany.com”. |
| COMPANY | A short one-line description of the company. |
| GRI_DESC | A short one-line description of the site; for example “Reporting Intranet for Wigets.com”. |
| HOST | The hostname where the web services run; typically set to the same value as the “URLPREFIX” - that is fully qualified name of the host. |
| MAX_UPLOAD_SIZE | The size in MB of the largest file that can be uploaded. The limit is typically “20”. |
| MAX_EMAIL_SIZE | The size in MB of files that can be used to emailed as attachments to users. |
| MIME_FILE | The file defines the various file extensions recognised by GRI. |
| STYLE_SHEET | The name of the style sheets. This should be the same as the setting for “GRI_STYLE_SHEET” in the macros configuration file |

| Variable | Purpose |
|----------------------------|---|
| | and is relative to the document root for the web-server. |
| LEFT_HEADER_LOGO | The small logo to use in the top-left of the header box. |
| MIDDLE_HEADER_LOGO | The log/thin logo to use in the middle of the header box. |
| RIGHT_HEADER_LOGO | The small logo to use on the right of the header box. |
| LEFT_HEADER_LINK | [Optional] Puts a link on the image on the left of the header. |
| MIDDLE_HEADER_LINK | [Optional] Puts a link on the image on the middle of the header. |
| RIGHT_HEADER_LINK | [Optional] Puts a link on the image on the right of the header. |
| SETUP_INIT_ROLES | If set to "1" will provide a link when the software is first run to allow an one-click set-up of the initial site administrator. |
| SETUP_INIT_PORTALS | If set to "1" and no portals exist will provide a message indicating that the site administrator is yet to set up any portals. If the current user has roles to allow portals to be configured it will provide a link to the portals/reports administration page. |
| SETUP_INIT_DOCREPOS | Similar to the above setting, but for document repositories. If you do not intend to have document repositories then set this to "0". When set to one will indicate repositories are not yet configured, or provide a link to set one up if the current user is defined with a suitable role. |

The logos referenced above are expected to be small; the size of the header area is defined by the size of the logos.

The "Email_addresses" Section

At present there is only a single element defined here - the "access_requests" email address. This is the email address which is used to send a message to the administrators allowing the access request to a portal to be granted.

This can be a single email address or a comma-separated list of email addresses.

The “Directories” Section

The directories section defines many directories that are used by the software and are pretty much self-evident.

| Directory | Purpose |
|------------------|---|
| auditdir | The directory where any audit logs are written to. |
| portals | The directory containing the portal configuration files. |
| templates | The directory containing the template web pages. |
| macros | The directory containing the commands to perform macro-replacement in the web page templates. |
| reports | The directory containing the report definition XML files. |
| alerts | Contains users alerts and the status of each. |
| users | The directory which contain the default preferences and user preferences. |
| database | The directory containing the files that make up the document repository. |
| docrepos | Typically set to the same value as “database” - where the actual uploaded files are copied to. |
| requests | A location where user requests for portal access are dealt with. This is currently not used, but is under active development. |

The “tmp_content” Section

This is a single element with two attributes;

- **localdir** - A directory used for temporary files; should be writable by the “www_owner” (typically set to “/opt/gri/reports/tmp”).
- **urlref** - The URL reference to get to the temporary directory defined above (typically set to “/opt/gri/reports/tmp”)

Secure Authentication Configuration

Use of the “secure” method of communication for the authentication daemon is recommended. This ensures only the web server is able to communicate with the authentication server and that all sensitive information is strongly encrypted over networks.

This section will assume that previously (as shown in “Secure Communication Installation” on page 12) the following two files exist for defining the RSA key-pair:

```
~/auth_rsa.public  
~/auth_rsa.private
```

The web server has the private key of the pair, and this should be placed in a directory as follows:

```
# mkdir /opt/gri/authd-privkey  
# chown webuid:webgid /opt/gri/authd-privkey  
# chmod 700 /opt/gri/authd-privkey  
# cp ~/authd_rsa.private /opt/gri/authd-privkey/authd.private
```

If the authentication server has already been set-up it is now possible to run a test. Assuming you are running as “root” or the web server user, then try the following :

```
# cd /opt/gri/tests  
# ./authd_client.pl  
Enter username: myuser  
Enter password: fgkfg45  
Result: OK
```

If a “Result:” line is returned this means that the web server to authentication daemon service is working; using the secure protocol if configured to do so.

Macro Handling Configuration File

This is a more detailed XML file and defines many attributes of how the site works. Usually copying the file is a good starting point, for example:

```
# cd /opt/gri
# cp gri_macro_handler.xml.example gri_macro_handler.xml
# chown www_owner gri_macro_handler.xml
# chmod 600 gri_macro_handler.xml.example
```

The example configuration file will look similar to the following:

```
<?xml version="1.0" standalone="yes"?>
<gri_macro>
  <env>
    <var name="GRI_LIBS" value="/opt/gri"/>
    <var name="GRI_CONFIG" value="/opt/gri/globals_config.xml"/>
    <var name="GRI_MACRO_CODEDIR" value="/opt/gri/macros"/>
    <var name="GRI_MACRO_TMPDIR" value="/opt/gri/tmp"/>
    <var name="GRI_MACRO_CACHE" value="/opt/gri/cache"/>
    <var name="GRI_USER_TO_EMAIL" value="/opt/gri/bin/null_mapper"/>
    <var name="GRI_STYLE_SHEET" value="/stylesheet-white.css"/>
  </env>
  <site>
    <mpl supported="no"/>
    <cache maxage="1"/>
    <!--
      <authentication program="/opt/gri/get_sso_id" logout="no"/>
    !-->
  </site>
</gri_macro>
```

The following "env" variables must be defined for the site to function:

| Var Name | Purpose |
|--------------------------|--|
| GRI_LIBS | This is the top level directory in which a "libs" folder can be found containing several libraries. |
| GRI_CONFIG | The "global" settings for the site - typically left alone and refers to the file defined in the next section. |
| GRI_MACRO_CODEDIR | Defines the directory where the macro command files reside - should not need to be changed. |
| GRI_MACRO_TMPDIR | Defines a temporary directory where files can be written to by the web user - typically does not need to be changed. |
| GRI_MACRO_CACHE | Not actually used at present; should be left unchanged. |
| GRI_USER_TO_EMAIL | This defines an external program to call to take a UNIX account and return the user's email address. The only one available by default is "null_mapper" which simply returns the UNIX user name - which is not suitable for most environments. |
| GRI_STYLE_SHEET | The name of the default style sheet to use for a user - this is the file as available in the web-servers document root - for example "/gri-default.css" |

LDAP Group Integration

In the "/opt/gri/globals_config.xml" file the following section defines whether the portal should provide LDAP group integration facilities:

```
<group_handling>
  <supported>1</supported>
  <gr_query_port>11199</gr_query_port>
  <gr_query_hosts>localhost</gr_query_hosts>
</group_handling>
```

To turn off group integration set the "supported" element shown above to "0". If you do wish to use this facility the following elements in "group_handling" must be set:

| Element | Purpose |
|-----------------------|---|
| supported | Set to "1" to turn on LDAP group support, and "0" to turn it off. |
| gr_query_port | A port number the daemon for querying of group information is listening on. |
| gr_query_hosts | A comma-separated list of hostnames to use for queries. Each host will be tried in turn until one responds on the port in question. |

LDAP Daemon Requirements

The daemon that is implemented must support the following clear-text queries from the reporting intranet web front-end server:

| Query | Purpose |
|----------------------|--|
| USERS | Returns a list of user ID's known on the server. |
| GROUPS | Returns a list of group ID's known on the server/LDAP database. |
| GROUP_COUNT | Returns a count of the number of groups known on the LDAP server. |
| USER nn[,...] | Returns a list of details for one or more user ID's specified. |
| GROUP nn | Returns the list of users in the specified group - and also a description of the group if available. |

If the queries are expensive (in terms of time or CPU effort) then it is expected for the daemon in question to cache the details for an appropriate period depending on the organisation in question.

LDAP Daemon Response formats

The expected responses from the queries are all in JSON - **though the first 8 bytes indicate the number of bytes that need to be read** - to allow easier parsing by the web processes. Some examples for each type are shown in the table.

| Query | Response |
|------------------------------------|---|
| USERS | 00173899{"error": "", "users": ["akz6mw", "ny84q9", "xxx123", , "nhob7me"], "rc": 0} |
| USER zippy | 00001324[{"user": {"department": "DATABASE STORAGE", "memberof": ["group1", "DG group 1", "DG global comms", "DG emea comms", "DG auto fix 2012", "DG perl groups"], "displayname": "Edwards, Simon", "uid": "XXX123", "email": "simon.edwards@mycorp.com"}, "error": "", "uid": "xxx123", "rc": 0}] |
| GROUP_COUNT | 00000038{"count": 54991, "error": "", "rc": 0} |
| GROUPS | 01584017{"error": "", "groups": ["*SS5Q1CIFS_D", "*SS5Q1PIFS_D", "*SS5Q1PITC_D", "*SS5Q1PMOR_D", "*SS5Q1PTSI_D", "*SS5Q1UPDT_D", "---ChngDistr-UX" , "xfr!dev D"], "rc": 0} |
| GROUP "DG EMEA Consultants" | 00000242{"error": "", "users": ["abym9b", "q1k1kcg", "x2w3n3f", "ddknnjf", "rekf69t", "nygxanl"], "rc": 0} |
| GROUP_SEARCH "consultants" | 00000992{"error": "", "groups": ["EMEA.MX.CONULTANTS", "RRE-EMEA_ECHO Consultants", "DG UNIX Consultants", "DG WINDOWS Consultants", "DG CONTRACT Consultants"], "rc": 0} |

Some notes about the above outputs:

- ☆ The responses have been edited to reduce space - for example the response from the "GROUPS" query (as indicated by the first 8 bytes) is actually 1,500,00 characters long!
- ☆ Apart from the "USER" request the JSON will be in the form of a single object/hash/dictionary. For the "USER" request a list of objects is returned - even if only one user is queried.
- ☆ All objects returned include an "rc" item - this is the return code and is "0" if successful or "1" if a failure occurs. For the "USER" request each user object has a separate "rc" item.
- ☆ The group search is case insensitive and can be a standard extended UNIX regular expression - for example the following is a perfectly acceptable request:

```
GROUP_SEARCH "(IBM|project)\s+Consultants"
```

The implementation of a suitable daemon is beyond the scope of this document.

Temporary Directories

The software requires a significant number of sub-directories under "/opt/gri" that must exist. These are first described and then the necessary commands to create them are given.

| Directory | Purpose |
|-------------------------------|---|
| /opt/gri/audit_reports | The directory where the audit reports generated by GRI itself are kept. |
| /opt/gri/logs | The directory where the audit log is kept. If this directory exists each and every page access for every user is recorded. |
| /opt/gri/database | This is the directory where the report repository files are kept. The names of the files kept in this directory are always numeric - or prefixed by a numeric - apart from the directory file itself - "gri.db". |
| /opt/gri/reports/tmp | A temporary directory used to store grabbed images and frames for reports to show on screen. HTML reports shown might include references to other pictures and frames. GRI grabs these and makes local renamed copies in this directory and manages the HTML to compensate. |
| /opt/gri/tmp | A directory for very short-lived temporary files ; usually this will appear empty. |
| /opt/gri/alerts | A directory which contains details of the alerts for all users and the status of these alerts. |

To create and set with the correct permissions do the following as "root":

```
# cd /opt/gri
# mkdir -p audit_reports logs database reports/tmp tmp alerts
# chmod 2750 audit_reports logs database reports/tmp tmp alerts
# chown www_owner:other audit_reports logs database reports/tmp tmp alerts
```

Scheduled Tasks

One of the requirements of the software is to ensure that certain tasks are performed on a regular basis. All tasks should be run as "root". These tasks include:

- **Sending out alerts** - when users subscribe to alerting the software has to regularly check the reports and if any are updated generate an email for the user.
- **Audit Reporting** - Nightly reports generated for the usage audit trail. This is optional but highly recommended.
- **Latest Reports List** - a task which is used to show the latest reports each user has access to the GRI homepage.

The sending out alerts a "crontab" entry similar to the following should be created:

```
# --- START: Send out alerts for users subscribed to updated reports ---
# Occurrences: Every 10 mins
# Added: 13.01.2010
#
0,10,20,30,40,50 * * * * /usr/bin/ksh -c "cd /opt/gri; bin/send_gri_alerts"
#
# --- END: Send out alerts for users subscribed to updated reports ---
```

☆ The shell specified above as "/usr/bin/ksh" made need to be changed for your environment; another common value might be "/bin/bash".

For updating the information on the latest available documents applicable for each user to be optionally shown on the GRI home page, the following entry is necessary:

```
# --- START: Update list of updated reports for GRI ---
# Occurrences: 10mins/40mins past the hour, between 06:00 - 20:00
# Added: 17.08.2009
#
10,40 6-20 * * * cd /opt/gri; bin/gen_latest_reports >/dev/null 2>&1
#
# --- END: Update list of updated reports for GRI ---
```

For auditing three jobs must be run on a nightly basis. These are configured via entries similar to the following:

```
# --- START: Generate audit report for GRI access ---
# Occurrences: Daily, 23:58-23:59
# Added: 13.01.2010
#
59 23 * * * /opt/gri/bin/gri_audit_report1.wrapper
58 23 * * * /opt/gri/bin/gri_audit_report2.wrapper
58 23 * * * /opt/gri/bin/gri_audit_report3.wrapper
#
# --- END: Generate audit report for GRI access ---
```

Style Sheet Creation / Guidelines

Whichever style sheet is used it is quite easy to generate new ones. The HTML generated by the software is very straightforward and uses as few styles as possible. At around 175 lines of text it is quite easy to customize for an environment.

The most common change is to simply provide a different colour scheme. Considering the "stylesheet-white.css" file, there are only 9 colours that need to be changed:

| Colour | Purpose |
|--------|---|
| 000000 | Used for the majority of the text items. Hence unless there is a dark background this should be a dark colour. |
| 999999 | This is used for disabled text or "greyed" out areas. Typically should be close to the background colour. |
| D4001A | Used for a font called bigred - actually typically very similar to the "Header 2" colour - which is typically "FF0000". |
| E0E0E0 | The border colour for input fields. |
| E5E5E5 | The background colour for box and table header rows. |
| E9E9E9 | This is the border colour for all tables. |
| F5F5F5 | Background colour for "odd" rows in tables containing alternating colour backgrounds. |
| FF0000 | Header 2 colour (also used for its underline). Typically a different colour than the standard text colour. |
| FFFFFF | Used for the background colour; so typically a white or off-white or a light colour. |

Preferences Defaults / Settings

The GRI environment now supports per-user preferences; though at present use of this facility is quite limited. This functionality is based on having a default set of preferences which are used if a user's preferences have not been set.

The default set of preferences has an example which can be used, and is copied as follows:

```
# cd /opt/gri/users
# cp __DEFAULT__-prefs.xml.example __DEFAULT__-prefs.xml
# chown www_owner:www_group __DEFAULT__-prefs.xml
```

The contents define what appears on the preferences page, and the defaults currently based on the above example are:

```
<?xml version="1.0" standalone="yes"?>
<userprefs>
  <pref name="recent_files_pane" desc="Show information on recent files visible to user on front page."
    values="true|false">dHJlZQ==</pref>
  <pref name="recent_files_pane_count" min="1" max="20"
    desc="Number of files to list in recent files pane.">NQ==</pref>
  <pref name="recent_files_oldest" min="1" max="240"
    desc="Maximum age in hours of oldest file to show in recent files pane.">NzI=</pref>
  <pref name="gri_stylesheet" values="bac|blue|grey" desc="Choose intranet colour scheme.">YmFj</pref>
  <pref name="gri_alert_fontsize" values="8pt|10pt|12pt"
    desc="Define text size used in alerting email.">MTJwdA==</pref>
  <pref name="gri_alert_fontstyle" values="Comic Sans MS|Arial|Courier|Times"
    desc="Define text font used in alerting email.">Q29lcml1cg==</pref>
  <pref name="hover_report_info" values="Yes|No" desc="Show additional information when hovering over report
titles.">Tm8=</pref>
  <pref name="hover_portal_info" values="Yes|No" desc="Show additional information when hovering over portal
titles.">Tm8=</pref>
  <pref name="html_email" values="Yes|No" desc="Receive HTML email.">WWVz</pref>
  <pref name="date_fmt" values="%d/%m/%Y|%m/%d/%Y|%d/%d/%y|%m/%d/%y" desc="Change date format
used.">JWQvJW0vJVk=</pref>
  <pref name="datetime_fmt" values="%d/%m/%Y %H:%M:%S|%d/%m/%Y %H:%M|%m/%d/%Y %H:%M:%S|%m/%d/%Y %H:%M:%S"
desc="Change date/time format used.">JWQvJW0vJVkgJUg6JU0=</pref>
  <pref name="long_datetime_fmt" values="%c|%d/%m/%Y %H:%M:%S|%d/%m/%Y %H:%M|%m/%d/%Y %H:%M:%S|%m/%d/%Y %H:%M:%S"
desc="Change date/time format used.">JWM=</pref>
</userprefs>
```

The syntax for the preferences should be self-evident; if a preference is to provide a list of choices then the "values" element should be a "|" separated list of values to present/validate. If it should be a number then the "min" and "max" attributes should be set to the inclusive range of numbers.

The information in this file is used to dynamically generate Javascript on the preferences page to validate all changes to these settings.

The first thing to note about the preferences is how the values are stored; they are not clear text but instead base64 encoded. This allows preferences in the future to be added which include characters that might cause encoding problems in a standard XML file with recourse to CDATA sections.

To change the above examples into normal text a simple Perl fragment can be used, for example to convert "YmFj":

```
$ echo YmFj | perl -MMIME::Base64 -e '$x=<STDIN>; print decode_base64($x),"\n";'
bac
```

Converting text values to base64 values is just as straightforward. For example to encode "Arial" to base64, the following code be used:

```
# echo Ariel |
perl -MMIME::Base64 -e '$x=<STDIN>; chomp $x; print encode_base64($x,""),"\n";'
QXJpZWw=
```

The preferences currently available are described now.

| Preference | Purpose |
|--------------------------------|---|
| recent_files_pane | Indicates whether the front-page of GRI should show a panel containing information on recently updated documents that the user has access to. |
| recent_files_pane_count | The count of the number of documents to use in the "recent files" pane if visible. Values suitable are between "1" and "20". |
| recent_files_oldest | By "recent", the utility looks back this number of hours for documents to show in the pane - should be between "1" and "240". The defaults is "72" (3 days). |
| gri_stylesheet | This defines the suffix to the stylesheet that can be used as a colour scheme for the user. For example if set to "blue" the following stylesheet is used from the document root: stylesheet- blue .css |
| gri_alert_fontsize | The default font size to use in the HTML emails generated to alert users on report updates they want notification of. Values available "8pt", "10pt" and "12pt". The default is "12pt". |
| gri_alert_fontstyle | The font to use in the HTML email generated for an alert. Currently the available options are defined as "Comic Sans MS", "Arial", "Courier" ad "Times". The default is "Courier". |
| hover_report_info | When the cursor is positioned over a report in the reports list in a portal it will indicate additional details regarding that report. |
| hover_portal_info | When the cursor is hovering over a portal in the main page it will show some additional information regarding that portal. |
| html_email | A value of "yes" or "no" is used to indicate whether alerting emails are sent as html or plain text. |
| date_fmt | The format of printing a "short date" - DD/MM/YYYY is the default format (shown via "strftime" or "date" command formatting). |
| datetime_fmt | The format when printing a date and time - again using the "strftime" or "date" command formatting. The default is |

| Preference | Purpose |
|--------------------------|---|
| | "DD/MM/YYYY HH:MM:SS". |
| long_datetime_fmt | A longer string to present the current date/time - the default is the locale setting for date formatting. |

Custom Authentication Interface

By default the GRI environment is configured to use its own cookie for session authentication. This then links into the authentication daemon running a particular server/port to authenticate the user details.

This works perfectly fine but means that you occasionally have to enter a username and password to gain access to the site. For some sites this can be improved by making use of other infrastructure that can provide the authentication that is necessary.

To perform a custom authentication the "authentication" element is needed in the "gri_macro_handler.xml" configuration file, for example:

```
<site>
  <mpl supported="no"/>
  <cache maxage="1"/>
  <authentication program="/opt/gri/get_sso_id" logout="no"/>
</site>
```

The authentication element has two attributes;

- **program** - The name of an executable to run which determines the user ID to use for access to the site.
- **logout** - set to "yes" or "no". This indicates whether a logout link should be provided. At present if custom authentication is provided this should always be set to "no".

The "program" specified should print the user ID on standard output to use. An example above uses an integrated CA "Siteminder" configuration and uses a standard environment variable provided to convert to the username format matching the UNIX ID's (to match report access definitions).

```
#!/usr/bin/perl

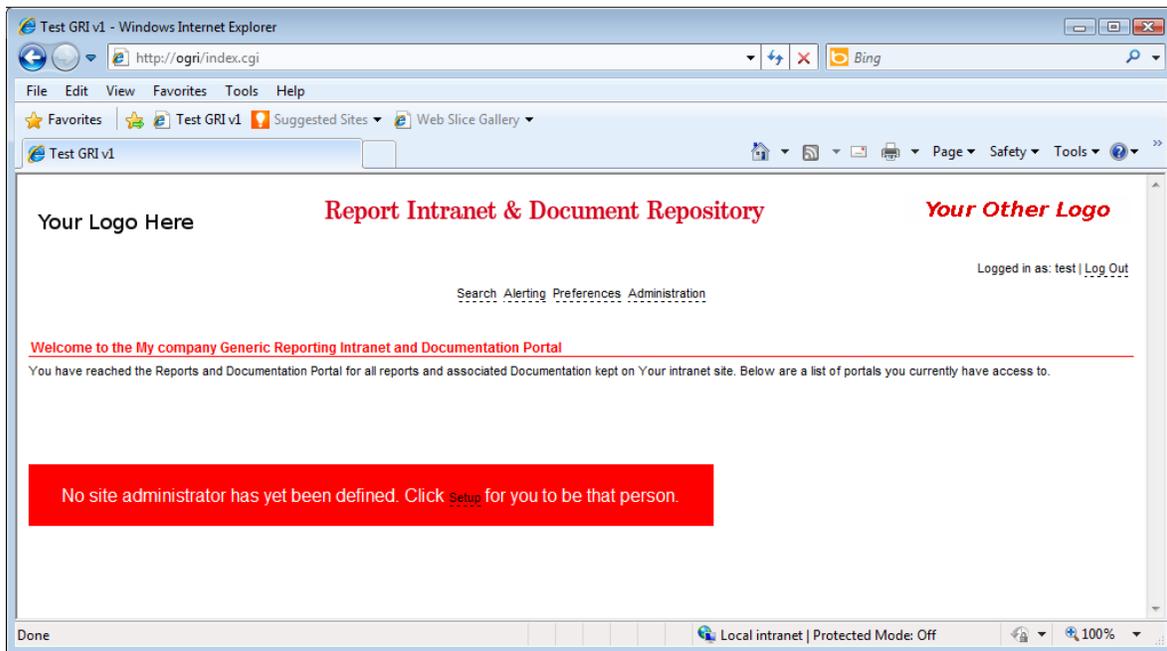
if(exists($::ENV{HTTP_SMUSER}) && length($::ENV{HTTP_SMUSER})) {
    $x=lc($::ENV{HTTP_SMUSER});
    $x =~ s!^\.*\\!!g;
    $x =~ s!^\.*\\!!g;
    print "$x\n";
}
exit(0);
```

Using GRI for the first time

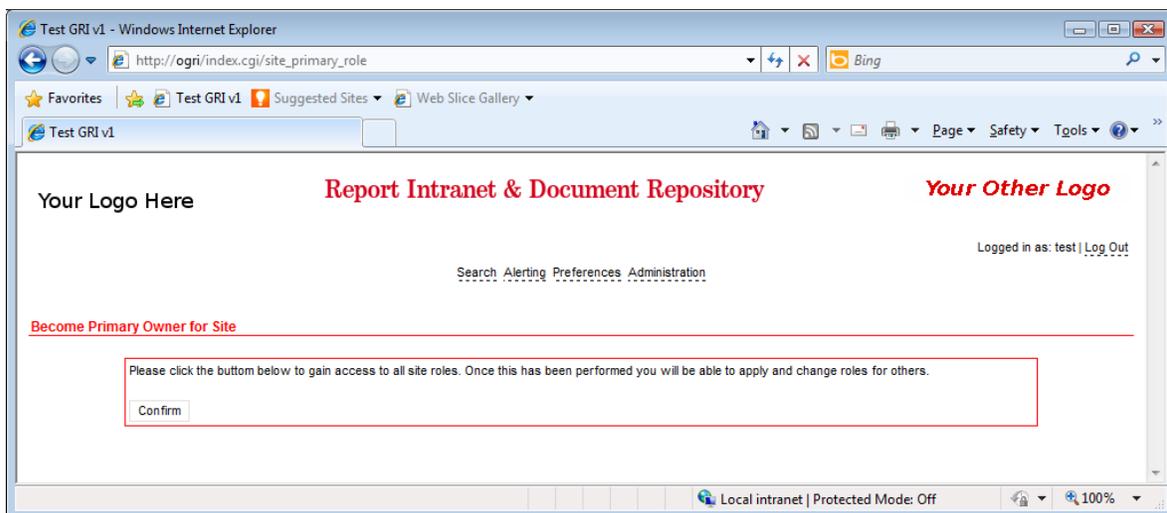
It is important to ensure that once the software is installed you access it first before allowing customers/users access. This can be used to ascertain whether the authentication is working as expected, and more importantly make use of some initial configuration links that might have been enabled to appear in the configuration file.

Defining the initial administrator

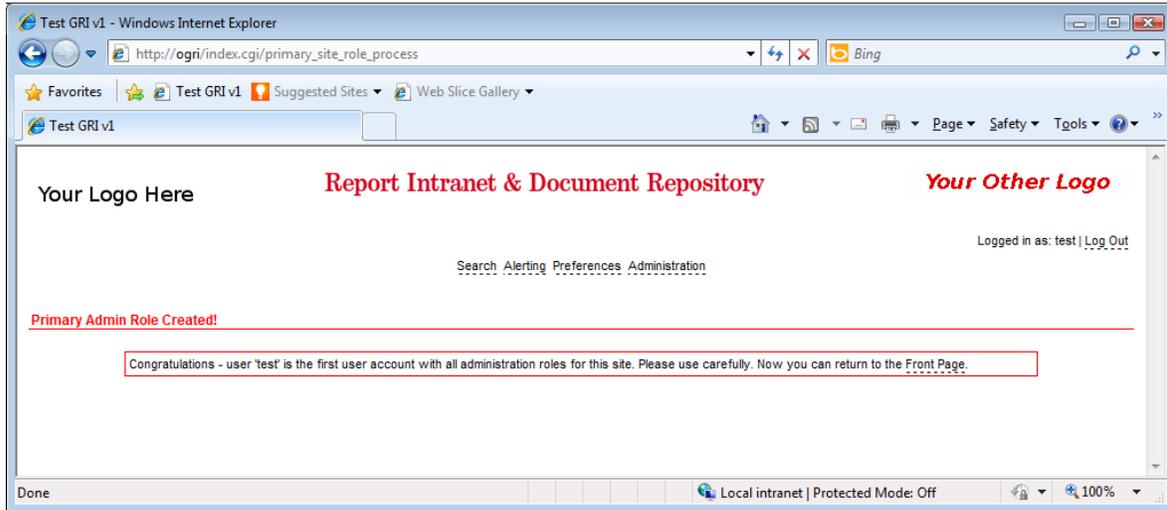
If the setting for "SETUP_INIT_ROLES" is set to 1, the first person that logs in will see a message similar to the following:



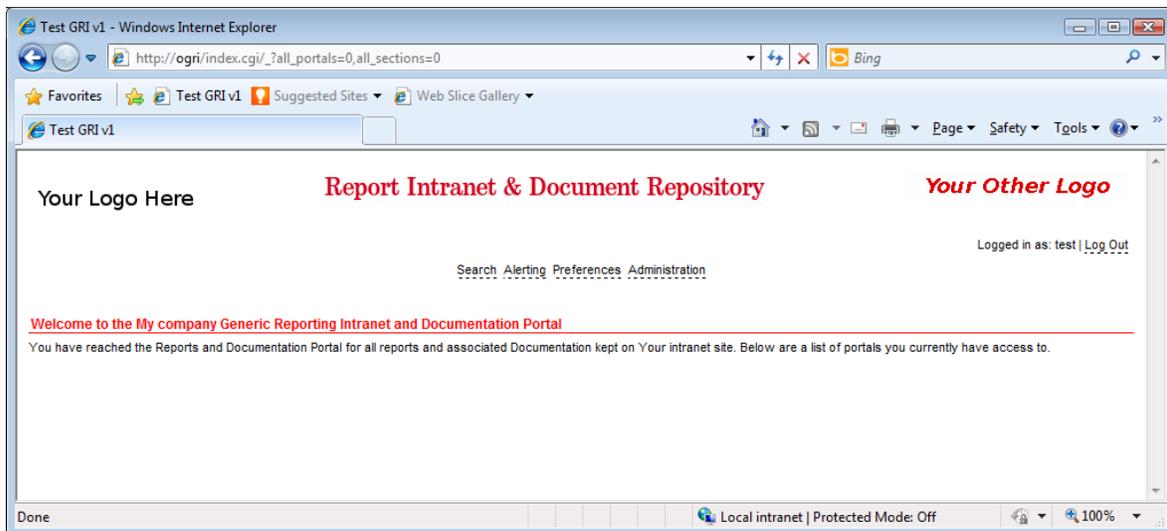
It is recommended you click on the [Setup](#) link to become the initial site administrator for security reasons. This can always be handed over to another user later quite easily. Before becoming the site administrator you will be asked to confirm the decision:



A message is then shown confirming you as administrator:



Once the administrator has been defined the initial page may appear quite blank:



However as the administrator you are now able to use the "Administration" links to define reports, portals, user groups and document repositories as necessary. For more information please see the "GRI Administrators Guide" documentation.

Appendix A: Example Apache 2 Virtual Server Configuration

The requirements for GRI are quite modest and so the virtual server definition for Apache is very straightforward.

In this case the easiest way is to extend the default site configuration file, often found at the following file path:

```
/etc/apache2/sites-enabled/000-default
```

```
<VirtualHost *>
  ServerName ogri
  DocumentRoot /opt/gri
  <Directory "/opt/gri">
    allow from all
    AddHandler cgi-script .cgi
    AddHandler cgi-script .pl
    Options +Indexes
  </Directory>
</VirtualHost>
```

You must then ensure your local clients have an IP address available for "ogri" pointing at the address of the web server host. Then they should be able to access the site:

<http://ogri>

Of course if HTTPS is configured then instead use:

<https://ogri>

Appendix B: Windows GRI Backend Service

Use of the "instsrv" from the Windows resource kit is recommended to install the server. For example in this instance the following command was used:

```
instsrv GriBackend "C:\srvapps\windows resource kits\tools\srwany.exe"
```

The above generates a Skeleton entry - the actual application and parameters must be added to the registry. In the registry editor navigate to:

```
HKEY_LOCAL_MACHINE
  .. SYSTEM
    .. CurrentControlSet
      .. Services
```

Navigate to "GriBackend" and add a "Parameters" Key, and under that:

```
Application "C:\Python31\python31.exe"
AppParameters "D:\scripts\gri\start_gri.py"
```

In this instance the Python script "start_gri.py" is used to initiate the actual Perl process in the background.