Table of Contents

HTTP Deployment TF - Information for sites

Introduction

The HTTP Deployment Task Force is overseeing and encouraging the deployment of HTTP/WebDAV as a new protocol for file management and access on the grid. Once the deployment is complete, an authenticated user should be able to successfully upload, retrieve, and manage files by simply using an HTTP client. In order to help administrators monitor the status of HTTP support on their server, a probe script for Nagios has been developed and made available.

More information on the task force is available at https://twiki.cern.ch/twiki/bin/view/LCG/HTTPDeployment.

Do I need to fix my site?

The HTTP Deployment Task Force is responding to a strategic direction within WLCG (further details on the TF home page) so sites are encouraged to deploy the interface and fix problems. One of the motivations of the strategy is to aid sites by making their storage systems more easily accessible by other communities which they may be serving.

While neither Atlas nor LHCb have formulated an official policy regarding HTTP access, the Task Force representatives describe this as "encouraged but not critical". Please ask your experiment contacts for more details if in doubt.

The probe

The TF has created a SAM/Nagios probe to evaluate functional compliance of storage systems' HTTP interfaces.

Results

Atlas results

LHCb: results

Operation

The administrator of the nagios instance specifies the host and the directory to which the probe script is pointed to, eg `https://dpmhead-trunk.cern.ch/dpm/cern.ch/home/dteam/`. The first two tests are read-only and do not require write access:

- Scan for all TLS ciphers the endpoint supports, and ensure there's at least one which is not RC4 or NULL.
- HEAD request to the given directory.
- GET request to the given directory.

The rest of the tests revolve around uploading and manipulating a test file. The proxy certificate with which the script is run needs to have read and write access to the target directory. The script performs the following requests:

- PUT request to upload a test file to the server.

- GET request to download the file again and verify that the contents are identical to the file previously uploaded.
- OPTIONS request to retrieve which methods the server allows.
- MOVE request to change the filename of the file which was uploaded.
- HEAD on the new filename to verify that the MOVE operation was successful.
- HEAD on the old filename, expecting to receive a "404 Not Found".
- PROPFIND on the new filename.
- DELETE on the new filename, so as not to pollute the server with test files.
- DELETE on the old filename, expecting to receive a "404 Not Found".

If the initial PUT request fails, it creates the problematic situation in which the other tests cannot proceed, since they depend on having a test file. This is still a valid situation, however, in case we only want to test the read capabilities of the server.

The probe then makes a last-ditch effort to read a file named `${VO name in capitals}_HTTPTFtest.txt` to test all read-only operations on it (everything except 'MOVE' and 'DELETE'). This file needs to be uploaded before-hand by the administrators. If it does not exist, the rest of the tests are marked as UNKNOWN.

Automatic cleanup of accumulated testfiles

The script will try to detect if there are accumulated testfiles from previous failed runs on the target directory and delete them. Only files matching the testfile pattern will be affected: `test_webdav_access_{numbers}`. Care has been taken not to ever delete anything which does not match this pattern:

- a regular expression parses the listing and only matches files that contain the above pattern.
- a second check right before sending any `DELETE` request makes sure the filename contains the string `test_webdav_access` and is not a directory.

Time limits

There are two different time limits, and each one applies to each request separately, and not to the entire execution of the script.

- A soft timeout supplied through '--warning'. If a correct response is received by the server after this threshold, the test will trigger a WARNING.
- A hard timeout supplied through '--timeout'. After this threshold, the script gives up on waiting for a response and the test is marked as CRITICAL.

A test that completes successfully within the warning threshold is marked as OK. The default value is 10 seconds for a soft and 30 seconds for a hard timeout.

Installation

todo, packages

Running manually

The script can also be run from the command line, without the need for a nagios instance. This can be useful when troubleshooting. As a first step, checkout the code:

```
git clone https://gitlab.cern.ch/lcgdm/nagios-plugins-webdav.git
cd nagios-plugins-webdav/src
```

Here's an example invocation - make sure to specify `-vv` to get verbose output.

```
./check_webdav -E /tmp/x509up_uxyz --uri https://dpmhead-trunk.cern.ch/dpm/cern.ch/home/dteam/ -v
```

More usage information is available by running `./check_webdav --help`. Please note, the script is known **not** to work on **Ubuntu**, whose default version of pycurl uses gnutls, not openssl. You should be able to run without any problems on a SL6 machine, eg lxplus.

Source code

The source code can be found here. The script has been developed and is being maintained by Georgios Bitzes, feel free to send him an email with suggestions and comments or contact the TF via GGUS.

Troubleshooting a failing endpoint

Please click on the failing test to check the long output and get more details on the root of the problem. You should be able to see any exception messages from curl (if any), information on the proxy used, as well as all requests and responses exchanged between the probe and the server.

Connection errors

There are two likely causes of this error - either the endpoint is not set up at all (meaning there's no process listening for HTTPS requests on the specified port), or a connection could not be established because of a TLS error.

Curl exception: (7, "couldn't connect to host")

No connection could be established at all - there's likely nothing listening for connections on the specified port.

(Service Check Timed Out)

Similar to the previous case, no connection could be established. The host is likely DROPing incoming packets to closed ports instead of REJECTing them, so the probe is left waiting indefinitely for a reply to its first request until nagios kills it.

Curl exception: (51, "SSL: certificate subject name '$subjectName' does not match target host name '$hostName'")

This most likely results from a server certificate misconfiguration. Every TLS certificate used to authenticate a server contains what is called a Common Name, which has to match the hostname a client is trying to connect to. If you're trying to connect to example1.cern.ch but the server provides a certificate with a Common Name of example2.cern.ch, the client has to reject the certificate. Using an Alternate Name is also possible, of course.

A solution would be to change the certificate the server authenticates itself with, perhaps by issuing a new one with `$hostName` for Common Name or Alternate Name.

Curl exception: (35, 'SSL connect error')

- Check that the proxy being used by nagios is valid and has not expired. If `timeleft` is `00:00:00`, this is not an issue with your server but with nagios.

Curl exception: (60, 'Peer certificate cannot be authenticated with known CA certificates')

The certificate presented by the server is not trusted by the probe script. This either means the certificate is not signed by an appropriate authority for use on the grid (maybe you are using a self-signed certificate?), or the script has not been configured correctly to trust the grid certification authorities. (the ones typically found in

`/etc/grid-security/certificates`)

HTTP status errors

If a test has reached this point, it means that at least it was able to establish a connection to the server, and TLS authentication with the proxy was successful.

404 file/path not found

- Verify that the directory the probe script is pointed to exists - check the request headers to see which file or directory is being tried.

Storage systems, specific advice

The storage providers participate in the TF and have given the following links to access their configuration or HTTP related documentation:

dCache

dCache responds with 401 when deleting non-existing file

The correct response is 404. The problem was fixed before dCache v2.15.0 was released; therefore, any site deploying 2.15-dCache will provide the correct return code. The fix has been backported to all other supported branches of dCache (2.14, 2.13, 2.12, 2.11 and 2.10) and will be available with the next dCache bug-fix releases: v2.14.15, v2.13.26, v2.12.37, v2.11.48 and v2.10.57.

HTTP PUTs do not reference a space token

The solution is to configure the appropriate WriteToken on the directory where the PUT is attempted, as explained in Chapter 21 ("Using Space Reservations without SRM") of The Book: - https://www.dcache.org/manuals/book.shtml

DPM

Docs - DPM

General SSL configuration advice

The recommended DPM setup involves running https on the head node, where authentication and authorisation are performed, and running only http on the disk servers. This gives a significant performance boost. The head node redirects with an auth token which the disk server interprets, so auth proceeds as before even in this case. The default puppet configuration gives this recommended setup, but older yaim configs do not. The advice is to ensure that `NSSecureRedirect Off` is configured for `lcgdm_ns_module` in zlcgdm-dav.conf on the head node and that port 80 is accessible on the disk servers. The other advantage of this configuration is that you can avoid debugging ssl problems on your disk servers.

SSL Config on the head node (and on the disk servers if you must)

Failure to follow the following advice may lead to instabilities in the service. A puppet configuration will manage all this for you.

Ensure that you are managing gridsite's session cache, which can exhaust the file system's inodes. Make sure you have the following in `zlcgdm-dav.conf`.

```
GridSiteGridHTTP off
```

```
GridSiteAutoPasscode off
```

httpd has to be reloaded in order to read new CRLs, which means they can "expire" in long running processes. The solution is to periodically perform a `/usr/sbin/apachectl graceful`, ideally after fetch-crl is run, or otherwise for example every 6 hrs. Later version of `lcgdm-dav-server` install this by default.

Older DPM versions were distributed with an SSL config which did not enable all strong ciphers, leading to problems with modern clients (eg curl/Ubuntu, chrome) because there is no cipher overlap. **Remove** the following offending lines from zlcgdm-dav.conf

```
    # This improves HTTPS performance when the client disables encryption
    SSLCipherSuite      NULL-MD5:NULL:RC4-MD5:RC4:+LOW:+MEDIUM:+HIGH:+EXP
    SSLHonorCipherOrder on
```

403 Forbidden

- For a PUT failure on a DPM system: the `Write` flag is not enabled (on the head node, disk nodes) in `NSFlags` in zlcgdm-dav.conf (for both modules, `lcgdm_ns_module` and `lcgdm_disk_module`)

307 Temporary Redirect

- For a PUT failure on a DPM system: the `RemoteCopy` flag is not enabled (on the head node) in `NSFlags` in zlcgdm-dav.conf (for both modules, `lcgdm_ns_module` and `lcgdm_disk_module`)

EOS

Docs - EOS

StoRM

Docs - StoRM

xrootd

No information received

Task Force Contact

The HTTP Deployment Task Force can be contacted via GGUS or by email.

This topic: LCG > HTTPTFSAMProbe
Topic revision: r33 - 2017-05-19 - OliverKeeble