How changing '<' to '>=' introduced a weird and hard to track bug

Date:	2022-03-22
modified:	2022-03-22
tags:	Uyuni, reposync, urlgrabber, bug, proxy
description:	Even small changes can have a massic and unexpected impact
category:	Code
slug:	how_changing_less-than_to_greater-equal_introduced_a_weird_and_hard_to_track_bug
Author:	Dominik Wombacher
lang:	en
transid:	how_changing_less-than_to_greater-equal_introduced_a_weird_and_hard_to_track_bug
Status:	published

After upgrading to Uyuni 2022.2 I wasn't able to sync openSUSE Leap 15.3 and Oracle Linux 8 repositories anymore, the configured HTTP Proxy was ignored. Multiple people reported similar issues: #4932 (Archive: [1], [2]), #4850 (Archive: [1], [2]), #4826 (Archive: [1], [2]).

During multiple troubleshooting sessions I learned a lot about the Python Codebase and the reposync tool.

It took my quite a while to track down the Issue to the `urlgrabber` package.

What happened? Method `find_proxy` in `urlgrabber.grabber.URLGrabberOptions` will identify the proxy server that should be used based on the provided URL Scheme.

Let's assume `server.satellite.http_proxy` in `/etc/rhn/rhn.conf` is set to `http://10.11.12.13:80`.

```
>>> from urlgrabber.grabber import URLGrabberOptions
>>> opts = URLGrabberOptions(proxy=None, proxies={'http': 'http://10.11.12.13:80', 'https': 'http://10.11.12.13:80', 'ftp': 'http://10.11.12.13:80'})
>>> opts.find_proxy(b'http://test.example.com', b'http')
```

That's the simplified version of the relevant Code, a Instance of `URLGrabberOptions` contains a list of proxies, one per URL Scheme, the method `find_proxy` is used to choose the right proxy from `opts.proxies` based on two parameter, `url` and `scheme`, both are passed as type `bytes`.

The expected result, `opts.proxy` contains the value `http://10.11.12.13:80` from type `str`, actual result, `None`.

The package `urlgrabber` is quite old and not that actively maintained. Python 3 support was added in Version 4, the last release is from October 2019.

It looks like that the above issues, that proxy is None, comes from an inconsistent bytes / string conversion. If you pass the scheme as string instead bytes, you get the expected result:

```
>>> from urlgrabber.grabber import URLGrabberOptions
>>> opts = URLGrabberOptions(proxy=None, proxies={'http': 'http://10.11.12.13:80', 'https': 'http://10.11.12.13:80', 'ftp': 'http://10.11.12.13:80'})
>>> opts.find_proxy(b'http://test.example.com', 'http')
>>> opts.proxy
'http://10.11.12.13:80'
```

To get the Sync of openSUSE Leap 15.3 and Oracle Linux 8 Repositories working again through a http proxy, a small change was already sufficient:

```
diff --git a/backend/satellite_tools/download.py b/backend/satellite_tools/download.py
index 3d064e5c6ce..3b7a02d5176 100644
--- a/backend/satellite_tools/download.py
```

```
+++ b/backend/satellite_tools/download.py
@@ -114,7 +114,7 @@ def __init__(self, url, filename, opts, curl_cache, parent):
         self.parent = parent
         (url, parts) = opts.urlparser.parse(url, opts)
         (scheme, host, path, parm, query, frag) = parts
-        opts.find_proxy(url, scheme)
+        opts.find_proxy(url, scheme.decode("utf-8"))
         super().__init__(url, filename, opts)

     def _do_open(self):
```

Pull Request #4953 (Archive: [1], [2]) was accepted and merged, unfortunately a few user still reported issues, especially related to CentOS 7 this time.

After a quick check, I had the impression that syncing EL8 based Distributions like Oracle Linux 8 share the same Code as EL7 based Distributions like CentOS 7. More or less right, but due to things like *mirrorlist*, there are some additional steps and further calls of `urlgrabber` split across the reposync code.

I had to find another way to fix it without touching multiple files and methods. So I gave some hacky Monkey Patching a try. First I reverted the previous workaround:

```
diff --git a/python/spacewalk/satellite_tools/download.py b/python/spacewalk/satellite_tools/download.py
index 3b7a02d5176..3d064e5c6ce 100644
--- a/python/spacewalk/satellite_tools/download.py
+++ b/python/spacewalk/satellite_tools/download.py
@@ -114,7 +114,7 @@ def __init__(self, url, filename, opts, curl_cache, parent):
         self.parent = parent
         (url, parts) = opts.urlparser.parse(url, opts)
         (scheme, host, path, parm, query, frag) = parts
-        opts.find_proxy(url, scheme.decode("utf-8"))
+        opts.find_proxy(url, scheme)
         super().__init__(url, filename, opts)

     def _do_open(self):
```

Then I created a new `find_proxy` method, which just triggers the original one but performs the bytes to string conversion. The magic happens in the two lines after the method, `urlgrabber_find_proxy` becomes the method from `urlgrabber` and my own version replaces the original one. That way it doesn't matter where in `yum_src.py` anything related to `urlgrabber` will be triggered, scheme will always be converted to a string a the proxy set as configured and expected.

```
diff --git a/python/spacewalk/satellite_tools/repo_plugins/yum_src.py b/python/spacewalk/satellite_tools/repo_plugins/yum_src.py
index 85013cfb36a..39f36be61e5 100644
--- a/python/spacewalk/satellite_tools/repo_plugins/yum_src.py
+++ b/python/spacewalk/satellite_tools/repo_plugins/yum_src.py
@@ -80,6 +80,17 @@
 APACHE_USER = 'wwwrun'
 APACHE_GROUP = 'www'

+
+# Monkey Patch 'urlgrabber.grabber' method 'find_proxy' to enforce type string for variable 'scheme'
+# Workaround due to wrong byte/string handling in 'urlgrabber' package
+# Required by reposync to connect through http_proxy as configured
+def find_proxy(self, url, scheme):
+    urlgrabber_find_proxy(self, url, scheme.decode('utf-8'))
+
+urlgrabber_find_proxy = urlgrabber.grabber.URLGrabberOptions.find_proxy
+urlgrabber.grabber.URLGrabberOptions.find_proxy = find_proxy
+
+
 class ZyppoSync:
     """
     This class prepares a environment for running Zypper inside a dedicated reposync root
```

There is a Issue (Archive: [1], [2]) in the urlgrabber repository, until that's fixed, it looks like that a workaround in the Uyuni / reposync Codebase will be required.

I created a new Pull Request (Archive: [1], [2]) in the Uyuni Project based on the above described fix, let's see if anyone comes up with a more elegant solution or if the guys are happy with that one and agree to merge it.

Based on my tests, by syncing *openSUSE Leap 15.3*, *Oracle Linux 8*, *CentOS 7* and *Ubuntu 20.04* repositories, it should finally resolve all, so far known, Issues related to reposync and HTTP Proxy.

And what had all this to do with a change of '<' to '>='?

In PR #4604 (Archive: [1], [2]) the version was bumped, changing `python3-urlgrabber < 4` to `python3-urlgrabber >= 4` caused all that trouble and lot of issues where syncing repositories behind a http proxy was just not possible anymore.