Infrastructure - Task #7304

get() call failing with Identifier ending in escaped '%'

2015-08-20 15:26 - Andrei Buium

Status:	Closed	**Start date:**	2015-08-20
Priority:	Normal	**Due date:**	
Assignee:	Jing Tao	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	CCI-2.0.0		
Milestone:	None	**Story Points:**	
Product Version:	*		

Description

The client creates an identifier ending in '%' like this:
Test20152328658749_common-unicode-ascii-escaped-%
and calls get() on it (with escaped pid in the URL):
https://cn-dev-ucsb-1.test.dataone.org/cn/v2/object/Test20152328658749_common-unicode-ascii-escaped-%25

It expects a NotFound exception but checks for ServiceFailures.
The above yields a ServiceFailure:
ServiceFailure: 0000: URLDecoder: Incomplete trailing escape (%) pattern: common-unicode-ascii-escaped-%

History

#1 - 2015-08-20 15:32 - Jing Tao

In MN, it works:
https://mn-demo-6.test.dataone.org/knb/d1/mn/v2/object/Test20152328658749_common-unicode-ascii-escaped-%25

I got the NotFound exception:

No system metadata could be found for given PID: Test20152328658749_common-unicode-ascii-escaped-%

#2 - 2015-08-20 16:04 - Jing Tao

- Target version set to CCI-2.0.0

#3 - 2015-08-20 18:18 - Jing Tao

It turns out that the rest-proxy decoded the "%25" to the "%" first. Then Metacat decoded the "%" again which caused the error.

I wrote a short class:
public class DecodeMessage {

```
public static void main(String[] args) throws UnsupportedEncodingException
{
    String msg="Test20152328658749_common-unicode-ascii-escaped-%";
    String strTMsg = URLDecoder.decode(msg,"UTF-8");
    System.out.println(strTMsg);
}
```

}

And I got the error:

Exception in thread "main" java.lang.IllegalArgumentException: URLDecoder: Incomplete trailing escape (%) pattern

If the string is "Test20152328658749_common-unicode-ascii-escaped-%25", the code works.

#4 - 2015-08-21 18:02 - Jing Tao

- Status changed from New to Closed

- % Done changed from 0 to 100

- translation missing: en.field_remaining_hours set to 0.0

The method getRequestPID in the AbstractProxyController class decoded the PID. We added a decoding flag in that method and overloaded the method without the flag - which means please don't decode.

Metacat has no problem to handle encoded pid, but we worry that if those encoded pids can cause problems in other service.

Ben and I dug around and found this method is called 74 times in those classes:
v1/v2.CNAuthoirztionController
v1/v2.CNCoreController
v1/v2.CNReadController
v1/v2.CNReplicationController

In each class, the code looks like (just different service class):
proxyCNAuthorizationService.isAuthorized(servletContext,
request, response,
getRequestPID(request, GET_RESOURCE_AUTHORIZATION_PATH_V1), accept, V1);

The ProxyCNAuthorizationSerivce is an interface and the MetacatProxyCNAuthorizationSerivce is the only implementation, so we are sure it wouldn't cause any issue since Metacat can handle encoded PID.

Same thing happens to ProxyCNCoreSerivce and ProxyCNReplicationSerivce. The ProxyCNReadSerivce has two implementations - one is the Metacat, the other is ProxyCNReadServiceImpl which is a mock service. The ProxyCNReadServiceImpl is not needed to worry either.

Andrei tested the new code and it worked.