IssueTracking HTTPCLIENT-1625

Designing HTTPCLIENT-1625

This page tracks the design goals of the complete redesign of the GSS-based authentication in HttpClient. Namely, implementation decisions, known issues, questions, testing, etc. All code will be developed in a separate branch.

Implementation Decisions

Implemenation decisions are comprised of several blocks like interface implementations, exception handling, logging, etc.

Interface Implementations

- `AuthSchemeProvider`: merely a factory for creating `AuthScheme` instances. Implemenation will be `GSSBasedSchemeProvider`. It will take in one argument, the OID string of the desired authentication mechanism or simply the `AuthScheme` name.
- `AuthSchemeBase` (implements `ContextAwareAuthScheme`): the implementation `GSSBasedScheme` will take in one argument, the OID string of the desired authentication mechanism or simply the `AuthScheme`. It will internally maintain a stateful `GSSContext` for the authentication against a target or a proxy. Since the implementation itself does not know when it will be nulled and garbage collected, it will maintain its state internally and release the `GSSContext` immediately upon successful completion or the first failure. This implemenation will **not** be threadsafe.
- `Credentials`: this will be `GSSBasedCredentials` and will take in a `GSSCredential`. Useful if not the default `GSSCredential` will be used. It is also necessary to create a `GSSPrincipal` class which will wrap the `GSSName` from the credential.
- `UserTokenHandler`: TBD

Exception Handling

TBD

Logging

TBD

Open Issues

1. Response token is not handed over to `GSSBasedScheme`, thus authentication can never be completed. It is highly likely that the `HttpAuthenticator` needs to be changed. There must be a notion of `isClientFirst` just as in SASL (RFC 4422, section 5, 2a).

Important Notes

- A `CredentialsProvider` with an fake item must be set otherwise authentication is not triggered.
- As it turns out, the entire authentication handling is server-first oriented. There is no way to integrate client-first unless the internal code is rewritten. See discussion.

Testing

Testing is comprised of two sections: unit tests and integration tests.

Unit Tests

It has to be determined how one can reasonably mock GSS objects to test the new implementation.

Integration Tests

Integeration tests will be performed in a corporate environment with the following setup:

- Client OS: Windows 7, RHEL 6, FreeBSD 9.x, HP-UX 11.31
- Java runtime: 1.6 and 1.7 from Oracle, OpenJDK and HP
- Target servers: Microsoft Forefront TMG (HTTP proxy) (SSPI), Microsoft IIS 7.5/8.0 (SSPI), Apache Web Server 2.2.x with mod_spnego (MIT Kerberos) and Apache Tomcat 6 with Tomcat Authnz SPNEGO AD (JGSS).

> **Note**
>
> Not all combinations can be tested.

Concrete requests are still open.

Questions

1. Why does `MalformedChallengeException` not extend `AuthenticationException` though it is documented for authentication purposes?
   OK: `MalformedChallengeException` signals syntax violation of some sort presenting the client from understanding the challenge whereas `AuthenticationException` signals inability or unwillingness to respond to the challenge. To me these are different type of issues, but I am open to changing it in 5.0.
2. The name of `ChallengeState` is quite confusing. Where is the state? This is merely a `ChallengeHostType`.
   OK: We can deprecate it and replace with `AuthCounterpartType` or some such in 4.5.
3. Can a `ContextAwareAuthScheme` instance be reused?
   OK: It can be re-used and will automatically be re-used within the same context (requests executed with the same instance of `HttpContext`).
4. Can an `HttpContext` be used concurrently?
   MO: in theory, yes. See Javadoc of `HttpContext`.

Todos

- Update documentation of `DefaultUserTokenHandler`