AstroGrid Security Facade

Guy Rixon

Presentation to GWS-WG of IVOA, Baltimore, October 2008

A poor initial impression...

- *User experience [of certificate-based security]:*
  - *why is security necessary?*
  - *Certificates? .globus directories? WTF?*
- *Developer experience:*
  - *Buzkashi*

-- from M. Graham, *Alternate Security Mechanisms,* Trieste Interop

“IVOA security is hard; let's go shopping for alternatives”

Why this bad experience?

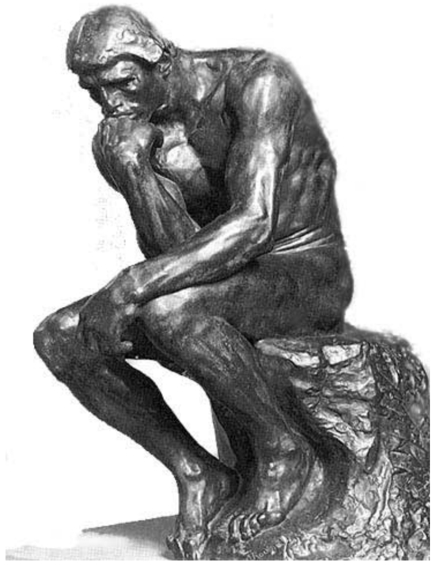
- Poor encapsulation of process
  - Users dealing with too many details
  - Never give a user live ammo, poisons or certificates
- Poor encapsulation of algorithms
  - Developers dealing with too many details
  - No one library does it all
  - Libraries don't play nicely together

Fixing the user experience

- Community services:
  - hide the certificate nasties inside a service with a nice UI
  - the model for which the IVOA protocols are intended
- ⇒ developers have to be able to write these service
  - ⇒fix the buzkashi problem first

Fixing the developer experience

- Single library of APIs for all security
- Covers all the necessary standards
- Actually designed for IVO work
- Uses 3rd-party jars widely
- Coherent, task-based docs
- No need to understand theory
- Hey, a man can dream...

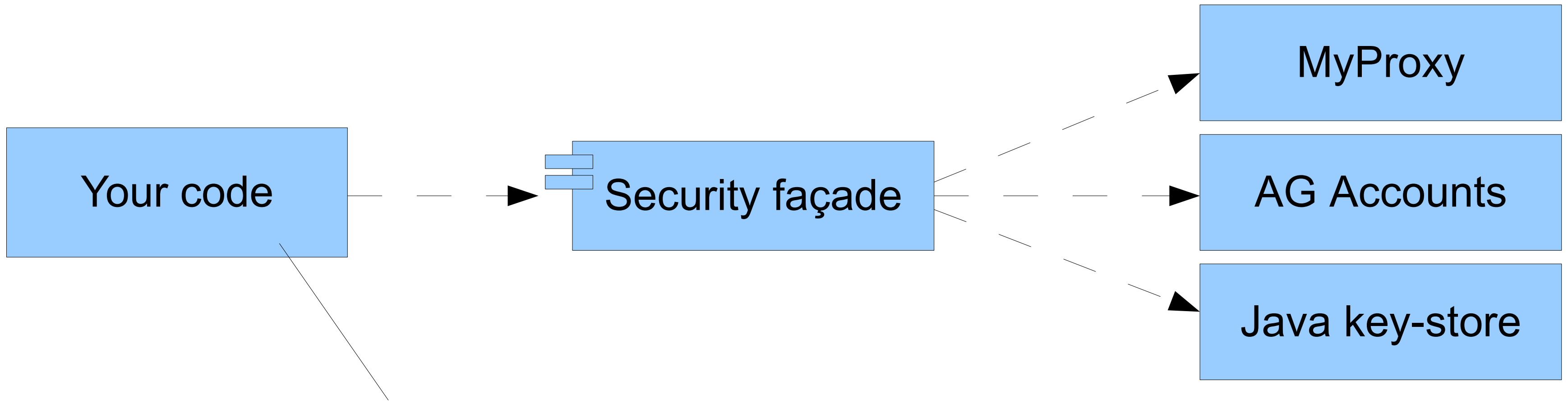


A possible solution

astrogrid-security-facade-2008.2.01.jar

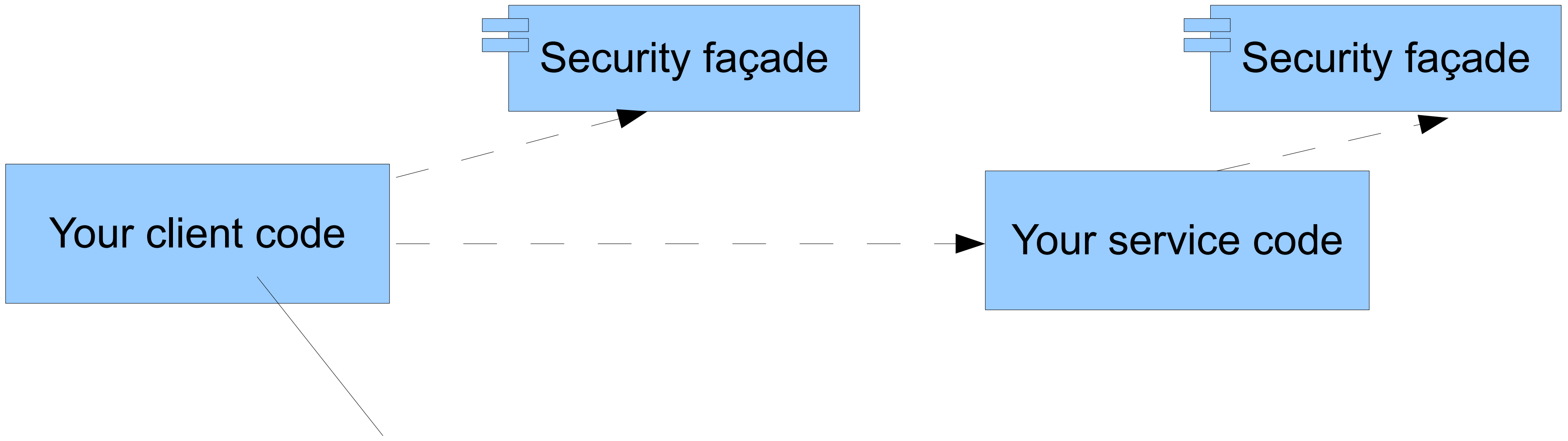
- Written for AstroGrid internal use
- Why not try it in other projects?
- Free (beer), open source, etc.
- Not dependent on the rest of AstroGrid
- Suggestions for enhancement accepted

Use case: signing on



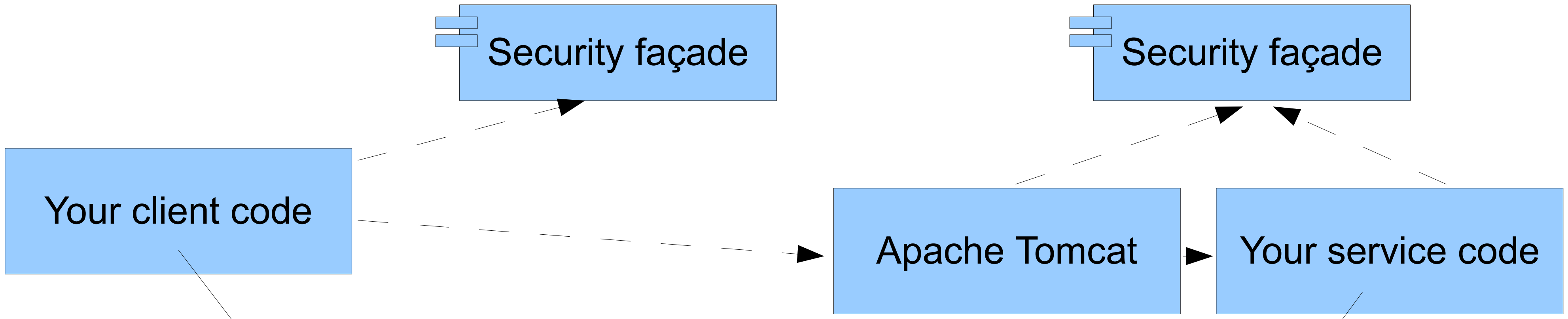
```
import org.astrogrid.security.SecurityGuard;
SecurityGuard sg = new SecurityGuard();
sg.signOn(userName, password, 36000, credentialSourceUri);
// sg now contains credentials for the current session
```

Use-case: digital signature



```
// sg is a security guard with credentials for the current session.
import org.astrogrid.security.AxisClientSecurityGuard;
AxisClientSecurityGuard sg2 = new AxisClientSecurityGuard(sg);
XyzPortType proxy = locator.getXyzPort(endpoint);
sg2.configureStub((org.apache.axis.client.Stub)proxy);
// Axis stub is now set up to sign digitally outgoing messages
```

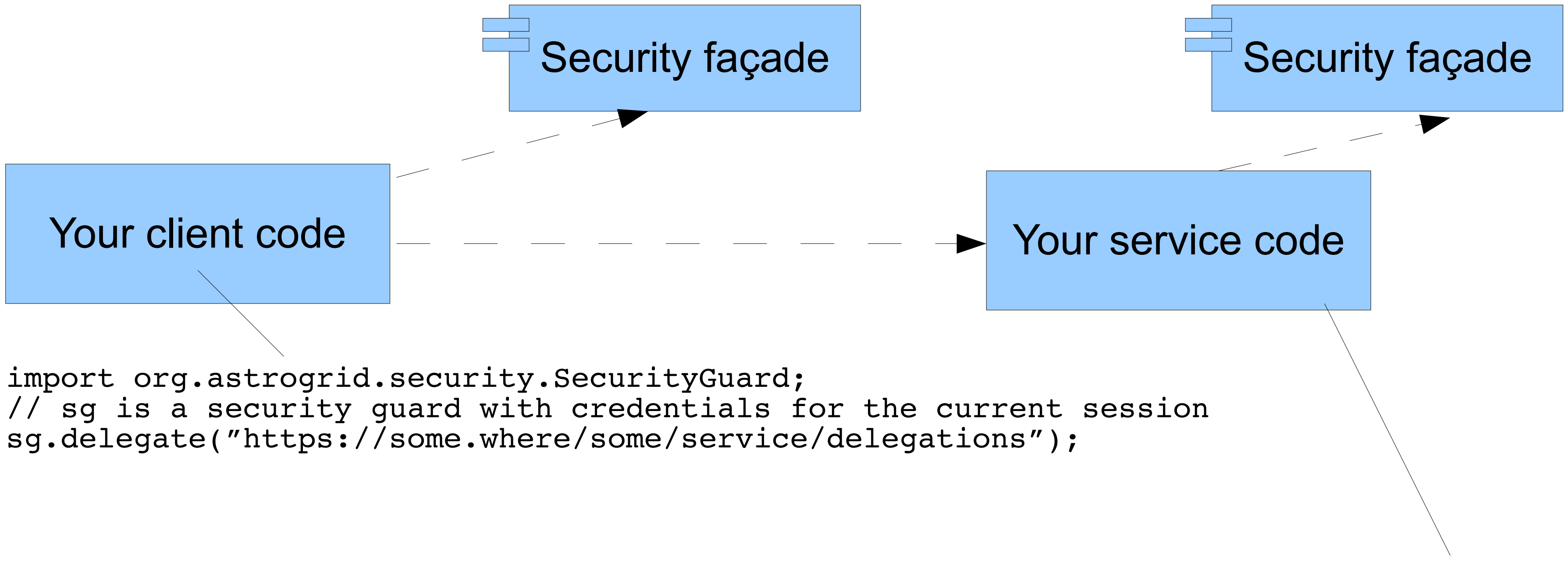
Use case: HTTPS authentication



```
import java.net.URL;
import java.net.HttpsURLConnection;
URL u = new URL("https://some.where/some/service");
HttpsURLConnection c = (HttpsURLConnection); u.openConnection();
// sg is a SecurityGuard with credentials for the current session
sg.configureHttps(c);
// Exchanges on connection c are now authenticated.
```

```
import org.astrogrid.security.HttpsServiceSecurityGuard;
HttpsServiceSecurityGuard sg = new HttpsServiceSecurityGuard();
sg.loadHttpsAuthentication(request); // request is the HttpServletRequest
```

Use case: delegation



```
import org.astrogrid.security.SecurityGuard;
// sg is a security guard with credentials for the current session
sg.delegate(”https://some.where/some/service/delegations”);
```

```
import org.astrogrid.security.HttpsServiceSecurityGuard;
HttpsServiceSecurityGuard sg = new HttpsServiceSecurityGuard();
sg.loadHttpsAuthentication(request); // request is the HttpServletRequest
sg.loadDelegation();
```

More information

- http://deployer.astrogrid.org/
- Guy Rixon <gtr@ast.cam.ac.uk>