The Credentials Pattern

Patrick Morrison and Eduardo B. Fernandez

Dept. of Computer Science & Engineering, Florida Atlantic University,
777 Glades Road, Boca Raton, FL 33341-0991
morrison@fau.edu, ed@cse.fau.edu

Provide portable means of recording authentication and authorization information for use in distributed systems.

Context

This pattern addresses data for authentication and authorization in distributed systems.

Example

Entry in to a country is an example of the problem to be solved; Countries must be able to distinguish between their citizens, citizens of nations friendly and unfriendly to them, trading partners, guests, and unwanted persons. Different persons may have different rules for how long they may stay, and for what they may engage in while they are in the country. Computer systems share some of these traits; they must be able to distinguish between members of their user community, and non-members. These non-members may be eligible or ineligible to gain system access or participate in transactions.

Problem

In individual computer systems, the authentication and authorization of a principal can be handled by that system's operating system, middleware and/or application software; all facts of the principal's identity and authorization are created by and are available to the system. With distributed systems, this is no longer the case. A principal's identity, authentication and authorization on one system does not carry over to another system. If a principal is to gain appropriate access to another system, some means of conveying this information must be introduced.

More broadly, this is a problem of exchanging data between trust boundaries. Within a given trust boundary, a single authority is in control, and can authenticate and make access decisions on its own. If the system is to accept requests from outside its own authority/trust boundary, an appeal must be made to some external authority about whether access and rights should be granted to the requesting principal. At the heart of both the external request and the appeal to authority is the data necessary to make these decisions.

The solution to this problem must resolve the following forces:

- Persistence: data must be packaged and stored in a way that survives travel between systems.
- Portability: data must be stored in a way that allows it to be used in contexts other than the one

it was created in.

- Protection: The original authentication and authorization information must be preserved intact, including defenses against tampering and forgery.
- Authentication: The data available must be sufficient for identifying the principal to the satisfaction of the accepting system's requirements.
- Authorization: The data available must be sufficient for determining what actions the presenting principal is permitted to take within the accepting system.

Solution

Introduce *Credentials,* which record the facts of authentication and authorization when the Subject is authenticated to the system.

Structure

Figure 1 illustrates the component relationships involved in Credential creation.

The Principal is an online entity such as a person or a process. The Principal has a set of Credentials, representing its identity and its authorization rights. A credential is a composite reflectng facts about the rights available to the principal. Each credential attribute maintains whether it has been authenticated, and when that occurred. As a separate concept, the credential attribute flags whether it is presently enabled, allowing principal control over whether to exercise the right implied by the credential. Expiration date allows control over the duration of the rights implied by the attribute.

Credentials are issued by an authority, and are checked by an Authenticator. Specialization of a credential is achieved through setting credential attribute names and values.

Some specific specializations of credential attributes are worth mentioning. Identity, created by setting an attribute name to, say, 'username' and the value to the appropriate username instance, reflects that the subject has been authenticated and identified as user known to the Authenticator. Privilege, named after the intended privilege, implies some specific ability granted to the subject. Group and Role can be indicated in a similar fashion to Identity.

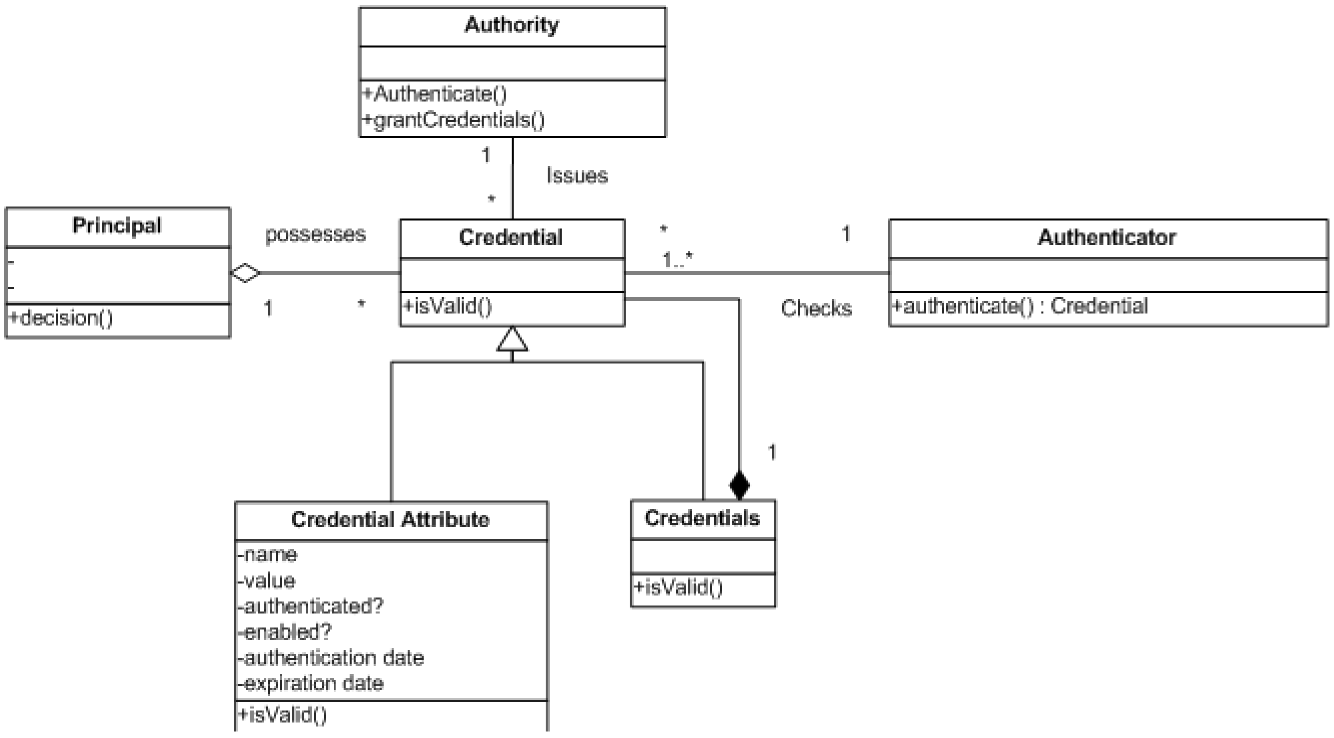

Figure 1: Credentials Class Diagram

Dynamics

There are two primary use cases; Issue Credentials, by which Credentials are granted to the Principal by an Authority, and Subject Authentication, where an Authenticator accepts Credentials provided to it by a Principal, and makes an access decision based on those Credentials.

Issue Credentials

The Principal presents itself and any required documentation of its identity to an Authority (e.g. Certificate Authority, State Department, etc...) . Based upon its rules and what it ascertains about the Principal, the Authority creates and returns a set of credentials. Anti-forgery devices for the medium in which the credentials are recorded must increase in strength relative to the kinds of information and operations under access control in the system involved.

A sequence diagram is given in Figure 2.   The returned data may include an identity credential, group and role membership credentials, and privilege credentials. As a special case, the Authority may generate a defined subset of 'public' credentials for Principals not previously known to the system. These credentials are made available to Authenticators which reference this Authority.

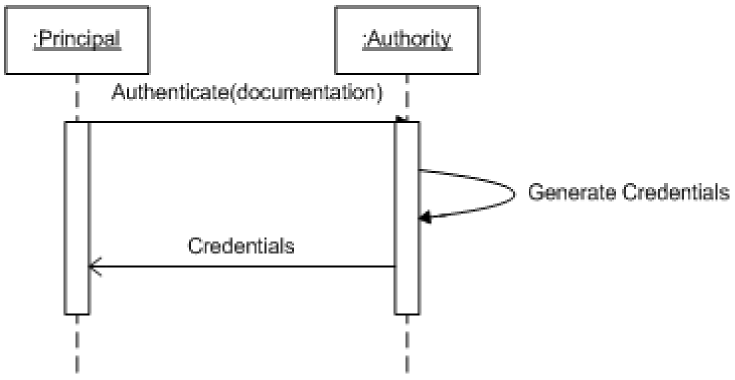

Figure 2: Grant Credentials sequence diagram

Principal Authentication

The Principal requests authentication at an Authenticator, suppling its name and authentication information, if it has this information. The Authenticator checks the Credentials and makes an access decision. There are different phases and strengths of check that may be appropriate for this step. For example, when entering my local warehouse club, I need only flash a card that looks like a membership card to the authenticator standing at the door. When it comes time to make a purchase, however, the membership card is checked for validity, expiration date and for whether it belongs to the person presenting it. In general, the authenticator is responsible for checking the authenticity of the credentials themselves (anti-forgery), whether they belong to their bearer, and whether they constitute valid access to the requested object(s). There is a good discussion of levels of inspection on page 246 of [And01b].

A sequence diagram is shown in Figure 3.

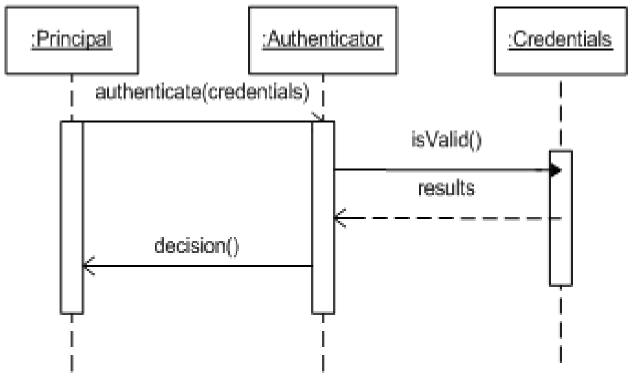

Figure 3: Subject Authentication sequence diagram

Consequences

This pattern has the following advantages:

- Fine-grained authentication and authorization information is recorded in a uniform, persistent and portable way.
- Credentials from a trusted authority can be considered proof of identity and of authorization.

This pattern has the following disadvantages:

- The authority issuing the credentials must be trusted
- The credentials must be made tamper-proof

Example Resolved

Create an authority, "Passport Agency". Give it the responsibility of verifying identity and granting a passport to subjects requesting credentials. The passport embodies the authority of the granting agency, and embodies the identity of the subject as verified by the agency. Design the physical characteristics of the passport so that it is hard to forge copies.

Place Authenticators at appropriate border crossings. As subjects seek entry to the country, check their passports for validity, and grant/deny entry based on the results.

Known Uses

This pattern was extracted from the ideas embodied in X.509 Certificates, CORBA Security Service's Credentials [And01], Windows security tokens [Bro05], XML assertions [Hug05], SAML, and passports.

Related Patterns

Metadata-based Access Control [Pri04] describes a similar environment. The *Credentials* pattern complements *Security Session* [Sch06] by giving an explicit definition of that pattern's 'Session Object', as extracted from several existing platforms. The *Authenticator* pattern [Bro99] describes the implementation of an authenticator.

References

[And01] R. Anderson, *CORBA Security Service Specification*, OMG 2001.
http://www.omg.org/docs/formal/02-03-11.pdf

[And01b] R. Anderson, *Security Engineering*, Wiley 2001.

[Bro99] F.L. Brown, J. DeVietri, E.B. Fernandez, "*The Authenticator Pattern*", Proceedings of Pattern Language of Programs (PloP'99), August 15-18, 1999.
http://citeseer.ist.psu.edu/brown99authenticator.html

[Bro05] K. Brown, *"The .NET Developer's Guide to Windows Security"*, Addison-Wesley, 2005.
http://pluralsight.com/wiki/default.aspx/Keith.GuideBook/HomePage.html

[Hug05] J. Hughes, E. Maler, *Security Assertion Markup Language (SAML) 2.0 Technical Overview*,
http://xml.coverpages.org/SAML-TechOverview20v03-11511.pdf

[Pri04] T. Priebe and E.B. Fernandez, J.I. Mehlau, G. Pernul. "*A Pattern System For Access Control*". "citeseer.ist.psu.edu/priebe04pattern.html, DBSec 2004: 235-249.

[Sch06] M. Schumacher, E. Fernandez, D. Hybertson, F. Buschmann, and P. Sommerlad, *Security Patterns: Integrating Security and Systems Engineering*, Wiley 2006.