Credentials in ProtoGENI

Rob Ricci
University of Utah

16 March, 2010

Introduction

- What credentials are used for
- Different types of credentials
- What credentials contain
- Credential delegation
- Example privileges

ProtoGENI credential usage

- *Credentials* are used in conjunction with PKIX to allow secure validation of permissions:
  - X.509 certificates prove that a key is bound to a principal;
  - Credentials prove that permissions have been assigned to that principal.
- Almost all ProtoGENI services should verify *both* certificates and credentials: relying on either alone is inadequate.
- Credentials ONLY EVER increase permission!
  - Presenting them is optional.

Types of credentials

- *"Self"* credentials, proving permission to invoke operations on a certain authority. For instance, a user will generally be able to obtain a self credential on their home slice authority.

- *Slice* and *sliver* credentials, evidence of permission for activity within a particular slice or sliver context with unspecified authorities.

- *Tickets* are also credentials, conveying permission to access designated resources.

- (And potentially many more... credentials are extensible.)

Credential contents

- A `target`: any named ProtoGENI object over which the credential applies. Never changes.
- An `owner`: the (only) principal who may present the credential. Can change, through delegation.
- Type-dependent description:
  - for privileges, a list of pairs of identifying strings and a flag indicating if that privilege may be delegated;
  - for tickets, an `rspec` describing the resources.
- Optional extensions.
- Optional parent credentials (in case of delegation).
- Signature(s) covering all of the above.

Credential `targets`

- Targets are specified with URNs.
- In theory, any named GENI object could be a target.
- In practice, targets are usually slices, slivers, or authorities (e.g. component managers).
- Credentials must *always* ultimately be signed by the authority identified in the URN; no other signature (not even that of the clearinghouse) will do.

Credential `owners`

- Owners are also specified with URNs (potentially of any type).

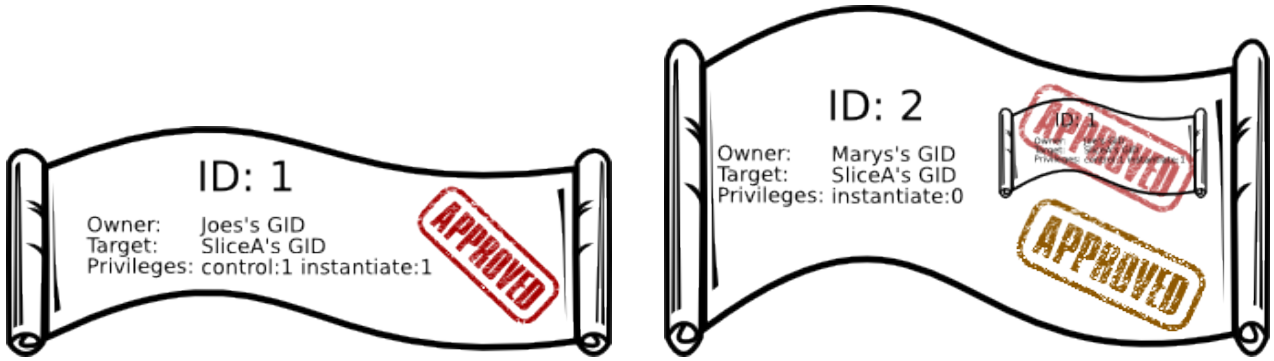
- In practice, owners are usually users, and sometimes authorities or slices.

Credential delegation

- If the credential allows it, the owner may *delegate* the corresponding permissions (or a subset of them) to another principal.

- The signed credential is embedded in a new credential, specifying:
  - the new permissions (no greater than the original);
  - the new owner;
  - a full copy of the original (including its parents, if any);
  - the old owner's signature.

Delegation (cont.)



- Note that delegation is decentralised!

- We should tend to be liberal with delegation policies. If the system discourages delegation (by providing inadequate tools, or restrictive policies), users will be tempted to "roll their own" delegation by sharing private keys, which is much worse!

Permission policies

- Sites are free to choose their own policies (it's not *technically* feasible to enforce distributed policies).

- Our component manager implementations implement a fairly simple default policy.

- We allow relatively simple tweaks to the default policies to allow custom rights, e.g.:

  - impose default resource limits and grant credentials which raise the limit;

  - local "superuser" admin credentials which allow a user to operate on any local sliver.

Example: admin credentials

- Suppose staff members at a particular component manager want to (locally) shut down a misbehaving slice.
- With access to the CM authority's private key, they can locally generate a credential with the CM itself as the target and one of their user accounts as the owner.
- They then use this new credential to invoke the `Shutdown` operation on the slice (instead of the customary slice credential, which they do not have).
- The CM is configured to accept a credential with itself as the target in lieu of a slice credential, so the slice can be shut down on this CM. (Other CMs would typically reject this admin credential.)
- We have script support for generating these credentials, and CM support for validating them.

Example permissions

Unprivileged	Informational	Manipulation

`GetVersion`
`Resolve`
`DiscoverResources`
	`GetSliver`
	`SliverStatus`
		`CreateSliver`
		`DeleteSlice`
		`RestartSliver`
		`RenewSliver`

(none required)	pi info	pi instantiate control

Detailed permissions (SA)

Unprivileged	Informational	Registration	Special

`GetVersion`
`GetCredential`
	`Resolve`
	`DiscoverResources`
	`GetKeys`
		`Register`
		`Remove`
			`BindToSlice`
			`Shutdown`

(none required)	authority resolve	authority refresh	(special)

Detailed permissions (CM)

Unprivileged	Informational	Manipulation	Special

`GetVersion`
`Resolve`
`DiscoverResources`
	`GetSliver`
	`SliverStatus`
	`SliceStatus`
	`SliverTicket`
		`CreateSliver`
		`RedeemTicket`
		`DeleteSlice`
		`DeleteSliver`
		`SplitSliver`
		`UpdateSliver`
		`StartSliver`
			`BindToSlice`
			`Shutdown`

(none required)	pi info	pi instantiate control	(special)