Aggregate Manager API Proposed Updates

Aaron Helsinger
March 13, 2012
www.geni.net

Session Goals

- Discuss AM API revision proposals
- Adopt those we agree on
- Postpone those we don't
- Discuss future proposals
- Discussions will continue via email, and at the Coding Sprint

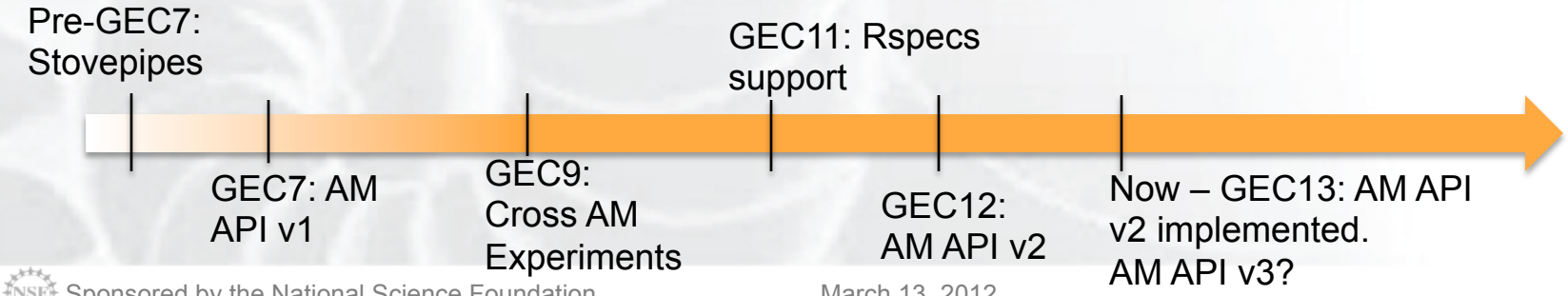
Agenda

- (5 minutes): Getting to AM API v3
- (25 minutes): Minimal AM API v3 Changes
  - Change D: Addressable Slivers
  - Change F1: Sliver States
  - Change I4: Option donotstart in CreateSlivers
  - Change F2: StartSlivers
  - Change G: Allow other Credentials
  - Change H: Clarify CreateSlivers
  - Change I1: Return expires from SliversStatus and CreateSlivers
  - Change I2: Return SSH public keys in SliversStatus
  - Change K: Standardize slice names, usernames
- (20 minutes): Proposals on the bubble
  - Change K: Require UUID and email in certs; Slice Ided by UUID;
  - Change L: Simplify SFA slice privileges
- (10 minutes): UpdateSlivers
- (10 minutes): Tickets
- (15 minutes): Jon Duerig: ProtoGENI perspective
- (10 minutes): Nick Bastin: Alternative proposals
- (20 minutes): Discussion
- (5 minutes): Next Steps

AM API Status Review

- Pre-GEC7: GENI was just stovepipe systems.
- AM API V1 Published at GEC7
- GEC9, AM API enabled cross-technology experiments.
- GEC11: implemented changes to specify RSpec schemas.

- GEC12: Adopted **AM API v2**
  - **now implemented by most AMs**



AM API Acceptance Tests

- GCF 1.6.1 includes scripts for testing aggregate compliance with the AM API:
  - v1 plus Change Set A (use GENI Rspecs)
  - v2 (basic tests only)
- We will add tests for v3 when adopted and defined.
- These tests will help ensure racks are AM API compliant.

Goals for AM API v3

- Specify unclear things
  - Semantics of CreateSliver, definition of sliver, sliver states
- Provide consistent experimenter info across AMs
  - Sliver expiration times, SSH login info
  - Slice name and username restrictions
- Allow for other credentials

- Define something solid that aggregates can implement this summer

API v3 Proposal Status

- Many proposals were emailed to geni-dev
- Vigorous debate – Thank you
- But….
- Too much too soon to agree to it all, let alone implement it all quickly

- **Plan**:
  - Adopt for v3 the things we can agree to
  - Continue discussing other ideas

Dividing the v3 Proposals

- **Minimal AM API v3**
  - Harden & clarify AM API v2
  - Adopt today
- **On the bubble**
  - UUID & email in certs, simplify slice privileges
  - Discuss today. Maybe adopt?
- **Discuss for AM API v4**
  - More complex or contentious topics
  - UpdateSlivers, Tickets
    - These are key experimenter needs

“Minimal” API v3 Proposal

- Addressable slivers: **Define sliver**, pluralize method names
  - But do not add the ability to delete or renew individual slivers.
- **Define sliver states**
- Add **"doNotStart" option** to CreateSlivers
- Include a way to start slivers: **new method StartSlivers**
- **Allow other credentials**
- Clarify that calling CreateSlivers with existing slivers is an error
- **Return an expiration** in SliversStatus for the set of resources at an aggregate.
  - But only 1 time, for all the slivers at the AM in this slice.
- **Return expiration** in CreateSlivers
- **Return SSH public keys** in SliversStatus
- Adopt **standardized slice names** and username rules/restrictions

Proposals to Keep Discussing

- On the bubble:
  - **UUID** and email address requirement for certs
  - Standardized **privilege names** in credentials
- For AM API v4+
  - **UpdateSlivers**
  - Ability to **Delete and Renew individual slivers**
  - **Tickets**
  - **ActOnSlivers** method
  - **Proxy** AM
  - Making slices identified by UUID
  - Cert **revocation** proposal

Biggest Experimenter Request!

Consider each Minimal v3 proposal

- We'll consider each part of the minimal AM API v3 proposal in turn.
- If there is debate, it is 'on the bubble'.
- When we finish, we'll have changes for AM API v3
  - Details TBD, e.g. at the Coding Sprint on Thursday
- Then consider changes 'on the bubble'.

Change Set D: Slivers and sliver groups

- **Issue**: API has the wrong definition of sliver
- **Proposal**:
  - Define sliver: an aggregate defined grouping of resources within a slice
  - CreateSliver → CreateSlivers
  - RenewSliver → RenewSlivers
  - DeleteSliver → DeleteSlivers
  - SliverStatus → SliversStatus

Change Set D Details

- **Issue**: API has the wrong definition of sliver
  - API methods should be operating on many slivers potentially
  - Ideally experimenters would operate on individual slivers
- ProtoGENI definitions for comparison:
  - AggregateSliver: all resources in the slice at an aggregate
    - This is what CreateSliver creates
  - ComponentSliver: things with a sliver_id, roughly 1 resource in each

Change D Details: Sliver Definition

- A *Sliver* is
  - an aggregate defined grouping of resources within a slice at this aggregate,
  - which can be used as an argument to methods.
  - The AM defines 1 or more of these groupings to satisfy a given resource request for a slice.
  - All reserved resources are directly contained by exactly 1 such sliver container, which is in precisely 1 slice.

Change Set D Details: Implications

- Rename some methods
  - CreateSliver → CreateSlivers
  - RenewSliver → RenewSlivers
  - DeleteSliver → DeleteSlivers
  - SliverStatus → SliversStatus

- Not now: Some methods take slice *or* sliver urn
  - AM does the right thing
  - Some slivers may not be independently deleted – AM may fail the request
  - Postponed due to disagreement

Convention: Grey text for postponed items from original proposals

Change F: Sliver States & StartSlivers

- **Issue**: Allocation and sliver states unclear

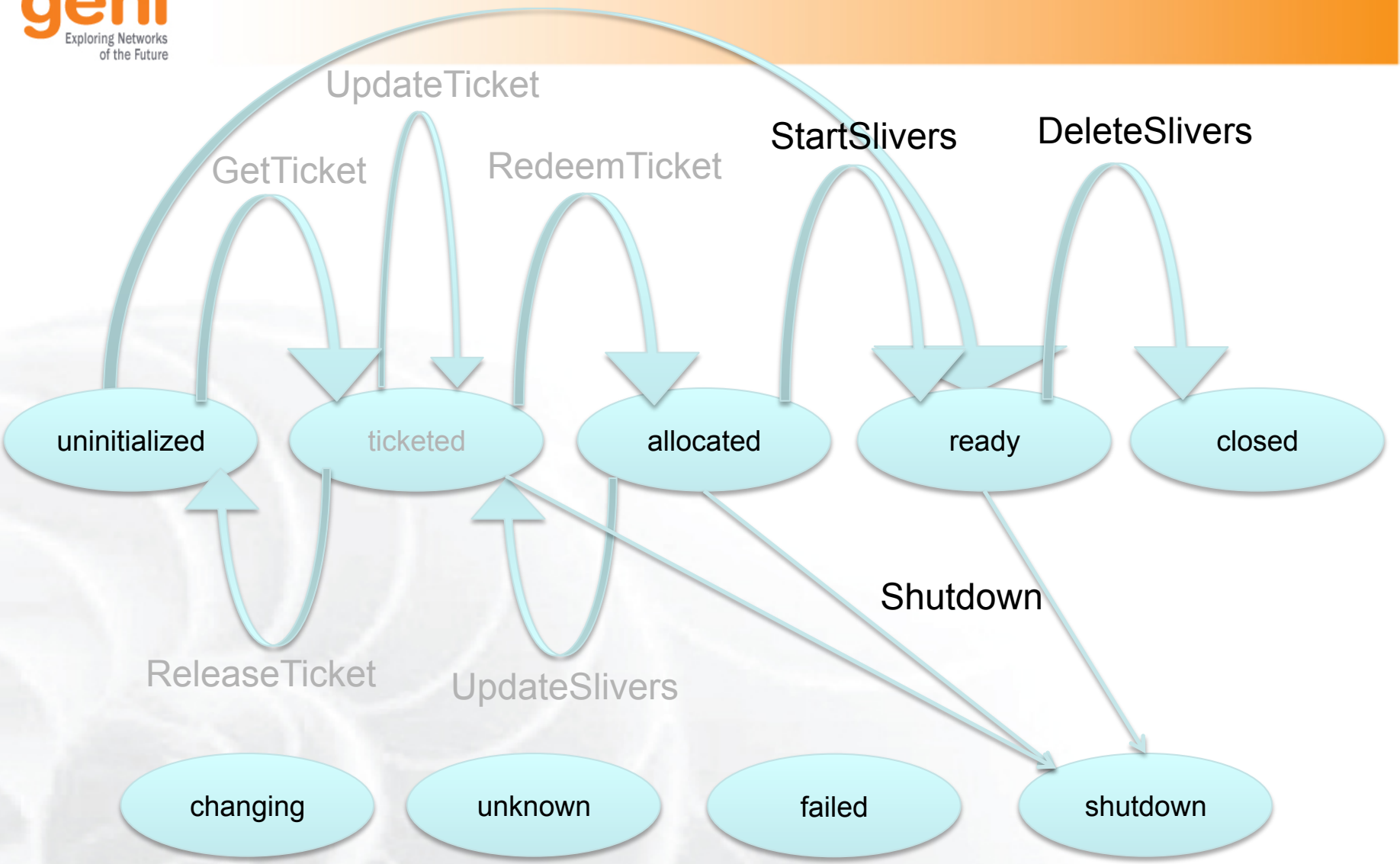
- **Proposal summary**:
  - `configuring` → `changing`, which can be used in many other cases, in returns from SliversStatus
  - New states `uninitialized`, `ticketed`, `allocated`, `closed`, and `shutdown` are added
  - Etc…
  - New method:
  - `struct StartSlivers(string slice_urn, string credentials[], struct options)`

Change F Details: Sliver States

- AM API methods logically change the state of the slivers at this AM.
- The API is not clear what experimenters should expect.
- The API does not provide easy ways for experimenters to control when and how states change.
- There is in particular no way to move slivers through states and change them in ways otherwise undefined by the API.

Change F1 Details: Define Sliver States

- State changes by method:
  - GetTicket: `uninitialized` → `ticketed`
  - UpdateTicket: `ticketed` → `ticketed`
  - ReleaseTicket: `ticketed` → `uninitialized` (or `allocated` if this was an update)
  - RedeemTicket: `ticketed` → `allocated`
  - CreateSlivers: `uninitialized` → `allocated`
    - And then → `ready` via `changing` if the `geni_donotstart` option is not supplied
  - StartSlivers: `allocated` → `ready` (via `changing`)
  - UpdateSlivers: `allocated` or `ready` → `ticketed`
  - DeleteSlivers: `ready` or `allocated` (or `changing`, etc) → `closed` (not `ticketed`)
  - Shutdown: `allocated` or `ready` → `shutdown`



'changing,' 'unknown', and 'failed' may be reached from any prior state. 'failed' is permanent, 'changing' and 'unknown' are transient. Operator action is required to move from 'failed' or 'shutdown'.

Change F1 Detailed Summary

- Summary of changes:
  - `configuring` → `changing`, which can be used in many other cases, in returns from SliversStatus
  - New states `uninitialized`, `ticketed`, `allocated`, `closed`, and `shutdown` are added
  - State transitions for each method are defined
  - `am_specific_status` optional return defined
  - `geni_status` is returned by SliversStatus, RedeemTicket, UpdateSlivers, CreateSlivers, and ActOnSlivers (if all relevant change sets are adopted).
  - Reminder: Aggregates can always expose non-standard methods at same port/protocol using same certificates and credentials

Change F2 Details: StartSlivers

- If resources from CreateSlivers were left `allocated` (`geni_donotstart` was supplied), then this method is how the resources are 'started'.
- This may be a no-op, or may require imaging and booting a VM.
- This method moves the slivers (all in the slice at this AM) from `allocated` via `changing` to `ready`.
- The method returns immediately; the 'starting' operation completes asynchronously.
- `struct StartSlivers(string slice_urn, string credentials[], struct options)`
- Not now: ActOnSlivers or sliver ioctl

Change I4: Optional donotstart

- **Issue**: CreateSlivers auto starts resources
- **Proposal**: New `geni_donotstart` option

Change I4 Details: Optional donotstart

- **Issue**: CreateSlivers auto starts resources
  - Experimenters may want more control.
  - Without tickets (transactions), an unstarted resource may be our best ‘cheap reservation’.
- **Proposal**: New `geni_donotstart` option
  - Note that ‘start’ may not mean anything for some resources
  - Note that we need a way to start slivers – hence the StartSlivers method.

Change Set G: Credentials are General

- **Issue**: The API says credentials follow a certain format.

- **Proposal**
  - Credentials are any signed document.

Change Set G Details: Credentials are General

- **Issue**: The API says credentials follow a certain format.
- ABAC uses different credentials.
- Aggregates might want other tokens.

- **Proposal**
  - Credentials are any signed document.
  - Aggregates must pick out the credentials they understand.
  - Aggregates must continue to accept current credentials.

Change H: 2nd CreateSlivers is an Error

- CreateSlivers specifies the slice contents at an AM.
- **Issue**: What happens if you call it twice, without calling DeleteSlivers?

- **Proposal**: it is an error
  - Treating it as UpdateSlivers is confusing to experimenters.

  - Note aggregates could accept an option to provide other semantics.

Change I: Return more Info

- **Issue**:
  - Sliver expiration is only returned by RenewSlivers.
  - SSH login info hard to find
- **Proposal**:
  - SliversStatus returns sliver expiration as `geni_expires`.
  - CreateSlivers returns it too. Change the return value from an Rspec to a struct
  - SliversStatus returns a `users` struct on any resource that has login info

Change I Details: Return more Info

- **Issue**: Sliver expiration is only returned by RenewSlivers.
- **Proposal**:
  - SliversStatus returns sliver expiration as `geni_expires`.
  - CreateSlivers returns it too. Change the return value from an Rspec to a struct:

```
{
        string rspec,
        string geni_expires,
        string geni_status,
        <others>
}
```

Change I Details continued

- **Issue**: Node login information is hard to find.
  - Returned in varying ways across aggregates
- **Proposal**: SliversStatus returns `users` struct for each resource that supports login:

```
'users' => [{'urn'   => $user1_urn.
             'login' => $login,
             'protocol' => [ssh, or ?],
             'port' => [22 or ?],
             'keys'  => [...] },
            {'urn'   => $user2_urn.
             'login' => $login,
             'protocol' => [ssh, or ?],
             'port' => [22 or ?],
             'keys'  => [...] }
           ]
```

Keys are SSH public keys

Change K: Standardize certs

- **Issue**: AM API requires particular certificates.

- **Proposal**: Standardize slice name, username, and a few certificate fields

Change K Details: Standardize certs

- **Issue**: AM API requires particular certificates.
- Certain useful elements are not standardized:

  → interoperability problems

  – What are valid slice names and usernames?

- Other useful elements are missing:
  – Distinct serial numbers to support CRLs, etc
  – Not now: slice & cert UUID and email addresses

Change K Details continued

- **Proposal**:
  - URN remains required in the cert Subject Alternative Name field (this is not a change)
  - serialNum is required to be unique per authority over time
  - Slice names must be <= 19 characters, alphanumeric plus hyphen
  - Usernames: <=8 characters, alphanumeric or underscore (not hyphen)
  - Not now: require UUID&email in certs; slices IDed by UUID, require CRL support

Minimal changes implications

- Must delete whole slice at an AM to add a node, change the OS on a single node
- No transactions/tickets/negotiation
- These are the top experimenter desires!
- These remain topics for discussion…

Adopt “Minimal” API v3 Proposal?

- Addressable slivers: **Define sliver**, pluralize method names
  - But do not add the ability to delete or renew individual slivers.
- **Define sliver states**
- Add **"doNotStart" option** to CreateSlivers
- Include a way to start slivers: **new method StartSlivers**
- **Allow other credentials**
- Clarify that calling CreateSlivers with existing slivers is an error
- **Return an expiration** in SliversStatus for the set of resources at an aggregate.
  - But only 1 time, for all the slivers at the AM in this slice.
- **Return SSH public keys** in SliversStatus
- **Return expiration** in CreateSlivers
- Adopt **standardized slice names** and username rules/restrictions

Proposals on the Bubble

- Some changes seemed less critical, more contentious
- We can discuss these now, consider adding them to AM API v3

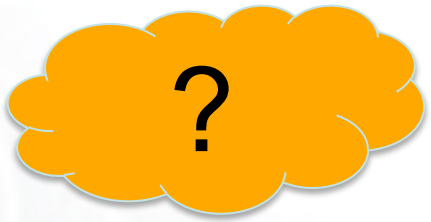
Change L: Simplify Slice privileges

- **Issue**:
  - SFA specifies privileges ‘bind’, ‘embed’, ‘pi’, etc.
  - Implementations today also use ‘*’ in varying ways
  - Aggregates have to support it all: a mess
- **Proposal**:
  - Replace existing privileges
  - CanRead (ListResources, SliversStatus)
  - CanWrite (all others)
  - Existing semantics remain
  - No flag day: accept existing slice credentials pre-change

Change K: Require more in certs

- Require UUID in user certs
  - PG and PL already include it
- Require UUID in slice certs
  - For ABAC
  - PG already includes it
- Make the UUID be a time-invariant ID for slices

Others on the bubble?



Agenda

- (5 minutes): Getting to AM API v3
- (25 minutes): Minimal AM API v3 Changes
  - Change D: Addressable Slivers
  - Change F1: Sliver States
  - Change I4: Option donotstart in CreateSlivers
  - Change F2: StartSlivers
  - Change G: Allow other Credentials
  - Change H: Clarify CreateSlivers
  - Change I1: Return expires from SliversStatus and CreateSlivers
  - Change I2: Return SSH public keys in SliversStatus
  - Change K: Standardize slice names, usernames
- (20 minutes): Proposals on the bubble
  - Change K: Require UUID and email in certs; Slice IDed by UUID;
  - Change L: Simplify SFA slice privileges
- **(10 minutes): UpdateSlivers**
- (10 minutes): Tickets
- (15 minutes): Jon Duerig: ProtoGENI perspective
- (10 minutes): Nick Bastin: Alternative proposals
- (20 minutes): Discussion
- (5 minutes): Next Steps

Other Changes

- Now discuss other changes

- UpdateSlivers
- Tickets aka Transactions
- Others?
  - RSpecs
  - Signed messages

Change Set C: UpdateSliver

- **Change Set C: Add UpdateSlivers**
- **Motivation**
  - Coffee girl: “Make my slice bigger!”
  - “Oops! I didn’t mean the default OS!”
  - “I’m done with this one node.”
  - In current API, you must delete your slice and recreate it – possibly losing state and even resources
- **Approaches** to UpdateSlivers in the community
  - Orca: can modify resources, depending on the resource. Working on add/remove resources
  - PG: UpdateSliver in CMv2 API. Takes request RSpec, returns a ticket
    - Experimenter separately redeems the ticket and restarts the sliver
    - Aggregate computes the difference from the existing state
  - PL: SFA has UpdateSlice, which is a synonym for

Change Set C: Proposal

```
struct UpdateSliver(string slice_urn,
                    string credentials[],
                    string rspec,
                    struct options)
Success return value, a struct:
{
  string ticket (or struct?),
  string geni_status (eg 'ticketed'),
  <others AM specific>
}
```

Note: See tickets proposal

Change Set C: Semantics

- Gives a ticket: what the AM *would* do
  - So experimenter can check while it is reversible
- Takes a **full** Rspec – not a diff
  - Sometimes calculating a diff may be impossible. AMs might return UNSUPPORTED
  - Semantics are really to apply a bunch of changes
  - Should we instead support 2 modes: complete and diff?
  - Maybe return is both complete and diff?

Change C semantics (cont)

- Can specify reservation length with `geni_end_time`
- Requires **ActOnSlivers** or RestartSlivers too.
  - AM allocates the resources. Experimenter controls when resources are ‘started’ or ‘restarted’.
  - Without ActOnSlivers or Start/Stop, AM would auto-start as in CreateSlivers

Change Set C: Discussion Points

- Full Rspec or a diff?
  - Who calculates the delta?
  - Support BOTH deltas AND full Rspecs?
  - Are mods hard to express? Merge?
- Should the return include the AM computed diff?

We must discuss!

Change Set E: Tickets

- What are tickets?
  - A promise to deliver (like a reservation)
  - A way to do **transactions**
  - Possibly: Proof of purchase (like a receipt that can be turned in for your goods)
  - Possibly: A way to reference and trade resources without physically moving boxes
  - A feature which, in name, appears in Orca, SFA, PlanetLab, and ProtoGENI
- Benefits and features of tickets
  - Negotiated reservations, or **coordinated reservations** across aggregates (like 2 phase commit)
    - EG: Allow experimenter to check manifest looks OK before committing
    - Or allow experimenter to ensure 2 AMs can both allocate nodes on the same link
  - Support **scheduled reservations**
  - Possibly: Support brokers to allocate, consolidate resources on behalf of others

Change Set E: Implementations

- Implementations in the community
  - Concept defined in SFA as a signed RSpec that promises to allocate resources.
  - ProtoGENI CM interface constructs them similarly, using them for 2 phase commit
  - Orca uses leases and tickets extensively for brokering, lending resources and for negotiated reservations / 2 phase commit. But they are structured differently.
- Parts of a proposal
  - Semantics
  - Structure
  - Methods

Change E: Tickets

- Options / Questions
  - Are tickets **signed**? Or just a set of information?
  - Are tickets **tied to a slice**?
  - Are tickets tied to an experimenter?
  - Are tickets **diffs or full state** requests?
  - Are tickets **delegatable**?

Change E: Ticket Options

- ProtoGENI style:
  - GetTicket, RedeemTicket, et al
  - Ticket is a signed struct, fully specifying state
- Broker style:
  - Like PG style but…
  - Ticket is a partial state – tickets are additive
  - Tickets can be delegated
  - AMs must honor presented tickets
- Transactions:
  - StartTransaction, CommitTransaction
  - Ticket is a struct of information
  - AM is authority on the transaction
  - Tickets could be additive or full

Agenda

- (5 minutes): Getting to AM API v3
- (25 minutes): Minimal AM API v3 Changes
  - Change D: Addressable Slivers
  - Change F1: Sliver States
  - Change I4: Option donotstart in CreateSlivers
  - Change F2: StartSlivers
  - Change G: Allow other Credentials
  - Change H: Clarify CreateSlivers
  - Change I1: Return expires from SliversStatus and CreateSlivers
  - Change I2: Return SSH public keys in SliversStatus
  - Change K: Standardize slice names, usernames
- (20 minutes): Proposals on the bubble
  - Change K: Require UUID and email in certs; Slice Ided by UUID;
  - Change L: Simplify SFA slice privileges
- (10 minutes): UpdateSlivers
- (10 minutes): Tickets
- **(15 minutes): Jon Duerig: ProtoGENI perspective**
- (10 minutes): Nick Bastin: Alternative proposals
- (20 minutes): Discussion
- (5 minutes): Next Steps

Discussion

Status Review

- Adopted multiple changes for AM API v3
- Discussed other changes
- Further Discussion?

Future Work

- Come to the Coding Sprint Thursday 1:30pm to work out details
- Finalize AM API v3 change sets
- Implement AM API v3 by GEC14?
  - AMs and clients
- Discuss other changes for AM API v4+