Ephemeral Data

In ZCS 8.8.0

Agenda

1. Motivation & Design Goals
2. Ephemeral Backends
3. Data Migration
4. Known Issues
5. Troubleshooting
6. Questions

Motivation

- LDAP is not suited for rapidly changing data: meant for *read often/change rarely*
- Poor storage solution for auth tokens, which can grow indefinitely
- Leads to poor LDAP performance, causing delays for users
- Validating a token requires fetching all values, then scanning until a match is found
- Server is responsible for handling token expiration
- Last Login Timestamp logging must be artificially throttled

> *"we have some accounts with up to 120,000 auth tokens stored in ldap, over 2 million overall for a couple hundred users"* - Wes Cilldhaire, Sol1 - https://bugzilla.zimbra.com/show_bug.cgi?id=104858

Affected Attributes

Three attributes were chosen as initial candidates for migration to ephemeral storage:

- **zimbraAuthTokens***:* authentication tokens that manage user sessions
- **zimbraCsrfTokenData***:* CSRF tokens that correspond to auth tokens
- **zimbraLastLogonTimestamp***:* currently throttled to avoid frequent writes

Design Goals

- API that better fits existing ephemeral data use cases
- Backend configuration via LDAP attribute
- Ephemeral attribute configuration via *zimbra-attrs.xml*
- Default, backwards-compatible LDAP backend
- High-performance external backend
- Migration tool to move data to new backend
- Ability to create/migrate further ephemeral attributes in the future

Setting the Backend

- Controlled via new config-level **zimbraEphemeralBackendURL** attribute
- Format is **[backend]:[configuration]**
- Default value is **ldap://default**
- SSDB value format is **ssdb:[host:port]**
- Value must resolve to functioning backend; attribute will not be changed otherwise!
- Successfully changing the value issues a *FlushCacheRequest* to all mailbox servers to flush config.
- Must be configured post-upgrade

LDAP Backend

- Default backend that routes data to existing LDAP system
- Same logic, but implemented behind ephemeral API
- All the drawbacks of the old system still exist
- No additional actions necessary for this to work

SSDB backend

What is it?

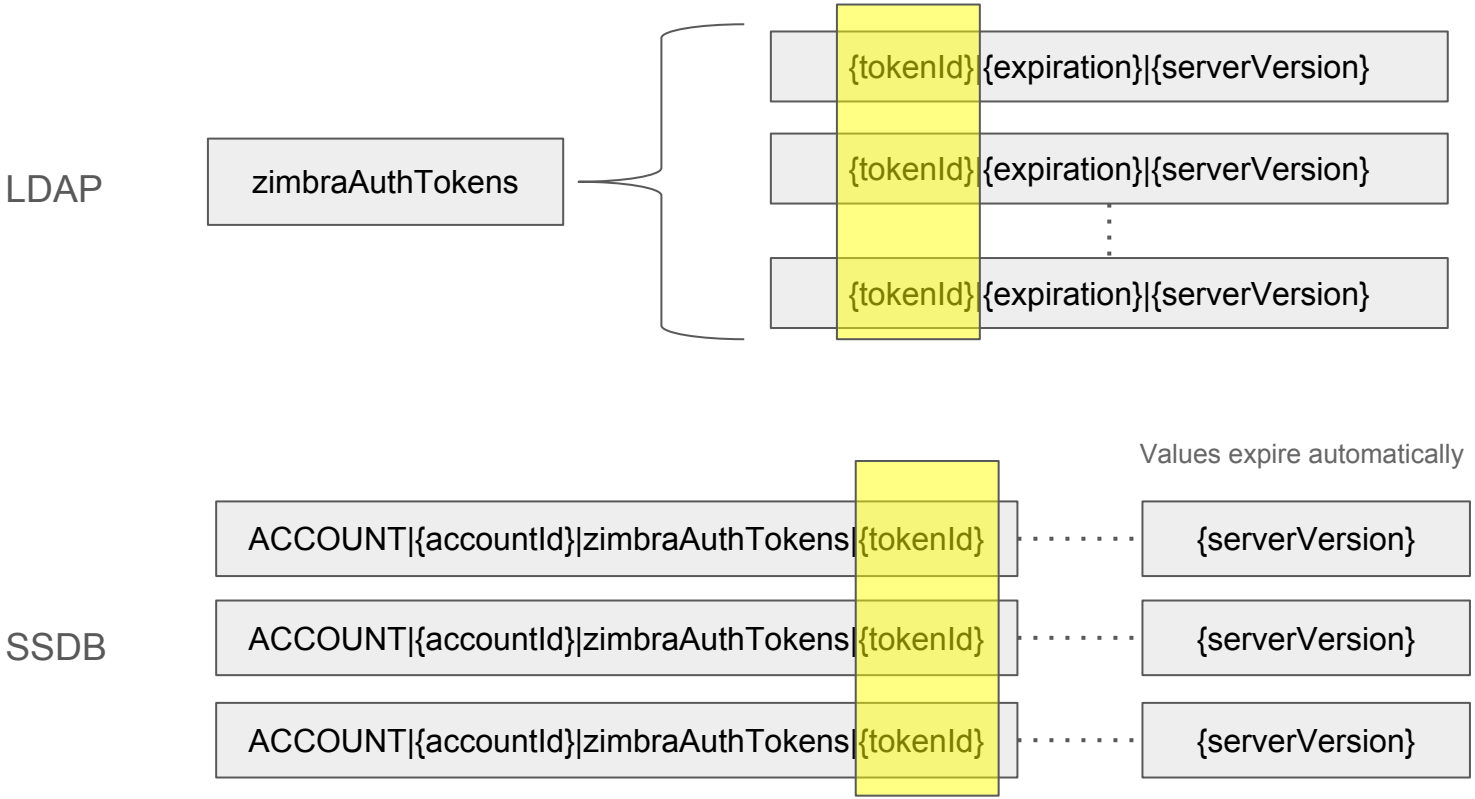
- Performant, Redis-compatible KV store
- Backend implemented in *com_zimbra_ssdb_ephemeral_store* server extension

Better than LDAP because:

- Schemaless, so token data can be encoded in the key for efficient lookups. No need to send all values to mailbox server!
- Can store more data than can fit in server memory
- Supports master/slave, master/master, sharding

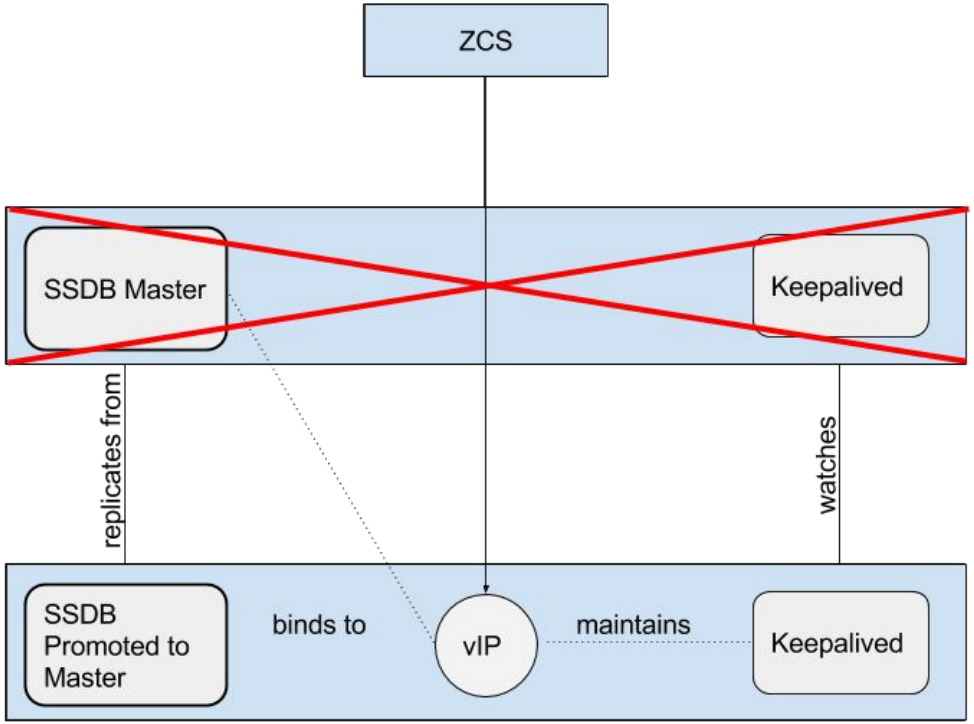
Note: SSDB is not currently included with the installer. Clients are responsible for installing and configuring up their own SSDB deployment. However, including it in the installer is planned.

Auth Token Data Structure Comparison



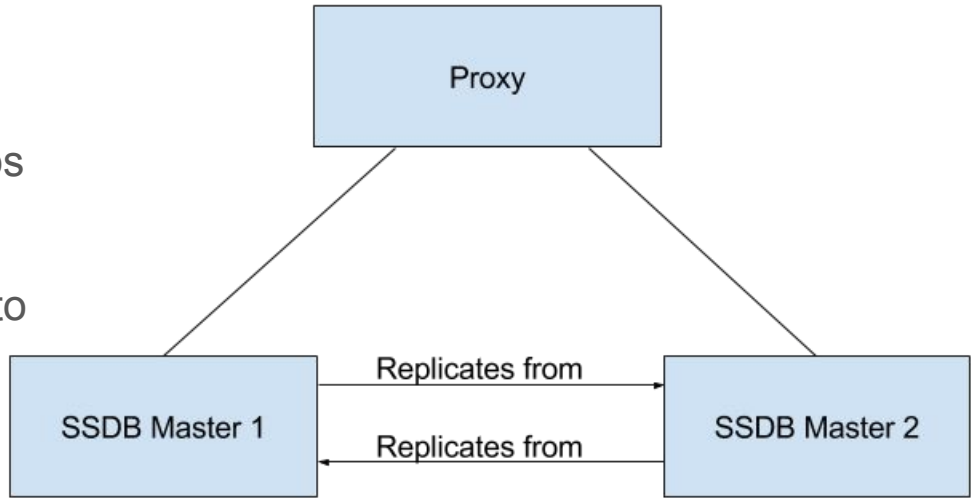
SSDB: Master-Slave

- Keepalived maintains virtual IP bound to SSDB Master
- Slave replicates from Master
- Keepalived on Slave detects failure on Master
- Slave is promoted to Master
- *zimbraEphemeralBackendURL* points to vIP



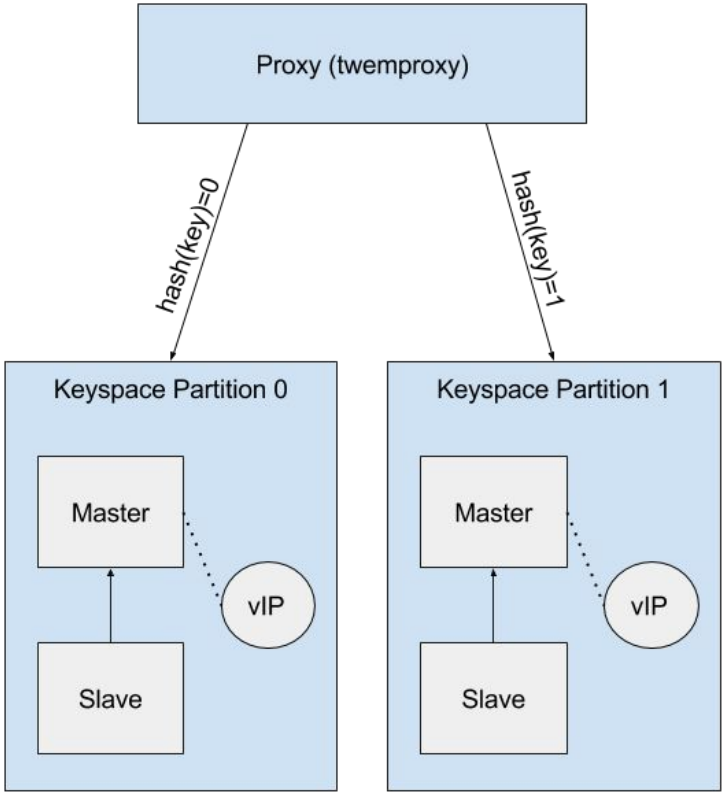
SSDB: Master-Master

- Masters replicate data between one another
- Proxy routes requests to Masters, drops failed servers
- *zimbraEphemeralBackendURL* points to Proxy
- Proxy should itself be highly-available



SSDB: Horizontal Scaling Via Multi-Master

- Multiple Master-Slave pairs
- Each pair stores a partition of the keyspace
- Proxy hashes inputs to partitions
- *zimbraEphemeralBackendURL* points to proxy



SSDB: Recommendations

- We recommend the Master/Slave option for clients whose ephemeral data can fit on one server.
- We do not recommend Master/Master configuration, as it requires a larger overhead with no upside.
- For larger clients, we recommend the Multi-Master configuration.

For more details on configuring SSDB, see the *ssdb-configuration-options* section of the admin guide.

Migration

- **/opt/zimbra/bin/zmmigrateattrs** is provided to move data from LDAP to the backend specified in *zimbraEphemeralBackendURL*
- Should be run immediately after changing the backend.
- Migration is one-way: cannot migrate back to LDAP, or to another backend.
- During migration, the server will look for ephemeral data in both the destination backend and in LDAP.

Migration Options

- **--keep-old**: Leaves old ephemeral data in LDAP after migration. Upside: possible to revert back to LDAP immediately after migration with no data loss. Downside: data will quickly become obsolete.
- **--num-threads**: Specifies how many threads are used for migration. More threads may consume more resources; less threads take more time.
- **--set-flag/--unset-flag**: set/unset the flag that tells the server to use the fallback mechanism (should not be necessary for normal operation)
- **--dry-run**: output the changes that would be made without performing the migration
- **--account**: comma-separated list of accounts to migrate (should not be necessary for normal operation)
- **--debug**: enable debug logging

Migration: CSV output

Each run of **zmmigrateattrs** generates a CSV file in the **/opt/zimbra/data/tmp/** directory.

- Contains migration info for every migrated account.
- If any migrations fail a CSV file report detailing only the errors is also created in the same directory.
- The names of the files are logged at the end of each run

Changing Backends Without Migration

If the backend is changed without migration, authentication data will be inaccessible. Users will need to re-login, and the new tokens will be stored in the new backend.

Potential scenarios:

- Delay between changing to SSDB and running migration script
- Reverting the backend back to LDAP
- Changing backend to a different (future) ephemeral backend

Changing the backend back will restore authentication data.

If migrating between SSDB backends, data should be replicated manually before changing the backend URL.

Migration Failures

- An interrupted ephemeral backend connection will cause the migration to terminate
- Once the issue is resolved, migration should be restarted
- Accounts migrated during the initial run will be skipped
- LDAP fallback mechanism will remain in effect until a migration completes successfully

Other Notes

- Values of *zimbraAuthTokens* and *zimbraCsrfTokenData* are no longer returned as part of the **zmprov ga <account>** call
- If data was migrated with the **--keep-old** option, these attributes *will* be returned, but they will reflect the state of the data at the time of migration
- *zimbraLastLogonTimestamp* is still throttled using *zimbraLastLogonTimestampFrequency* if the ephemeral backend is LDAP
- Documentation is available in the *ephemeraldata* and *ssdb-configuration-options* sections of the admin guide.

Known Issues

- **ZMS-519**: Non-default ephemeral backends currently cannot be used with a split-UI architecture. This is because the UI nodes validate auth tokens in certain scenarios. ZMS-531 is currently in-progress to resolve this.
- **ZMS-539**: During migration, users may experience long login times. This issue is currently under investigation with ZMS-544.

Diagnosing Problems

- Mailbox.log has a new *ephemeral* logger category
- If the ephemeral store cannot be instantiated, a FATAL error will be logged and the runtime will halt, necessitating a mailboxd restart.
- If “FATAL [] system - no EphemeralStore specified” is seen in the logs, it means that the SSDB extension is either not present or not instantiated.
- Ephemeral data in SSDB can be fetched directly using the **ssdb-cli** tool provided with every SSDB installation. This is of limited use, however, since keys are dynamically constructed.

Questions?