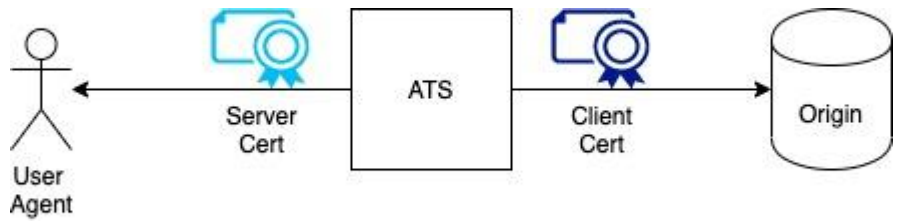
ATS Certificate Loading Plugin Support

Susan Hinrichs
ATS Summit
June 2020

ATS and Certificates

ATS provides both servers certs to user agents and client certs to origins



Current State of the World

- ATS loads server and client certificates from disk
  - On process start
  - On config reload
- Downsides
  - If certificates change on disk, someone must explicitly call config reload. The “hot load” problem
  - If an organization manages certs and keys elsewhere, another step must be introduced to move certs and keys to disk on the ATS server
- The details of any centralized key/cert management is probably very organizational-specific.
  - Plugins are great for such organization-specific business logic.

Previous Efforts on cert loading

Last year (or 2 years ago?), Zeyuan updated the core infrastructure and added Plugin APIs to allow for “hot loading” of client and server certificates

```
TSReturnCode TSSslClientCertUpdate(const char *cert_path, const char *key_path)

TSReturnCode TSSslServerCertUpdate(const char *cert_path, const char *key_path)
```

The idea was that an external tool would watch for changes on disk (or in a Key Server) and signal ATS that a file was ready to reload or feed in new PEM data.

- Limit complexity in ATS process

This worked in testing. But we got distracted, Zeyuan left, and this never got merged back into our ATS7 and deployed

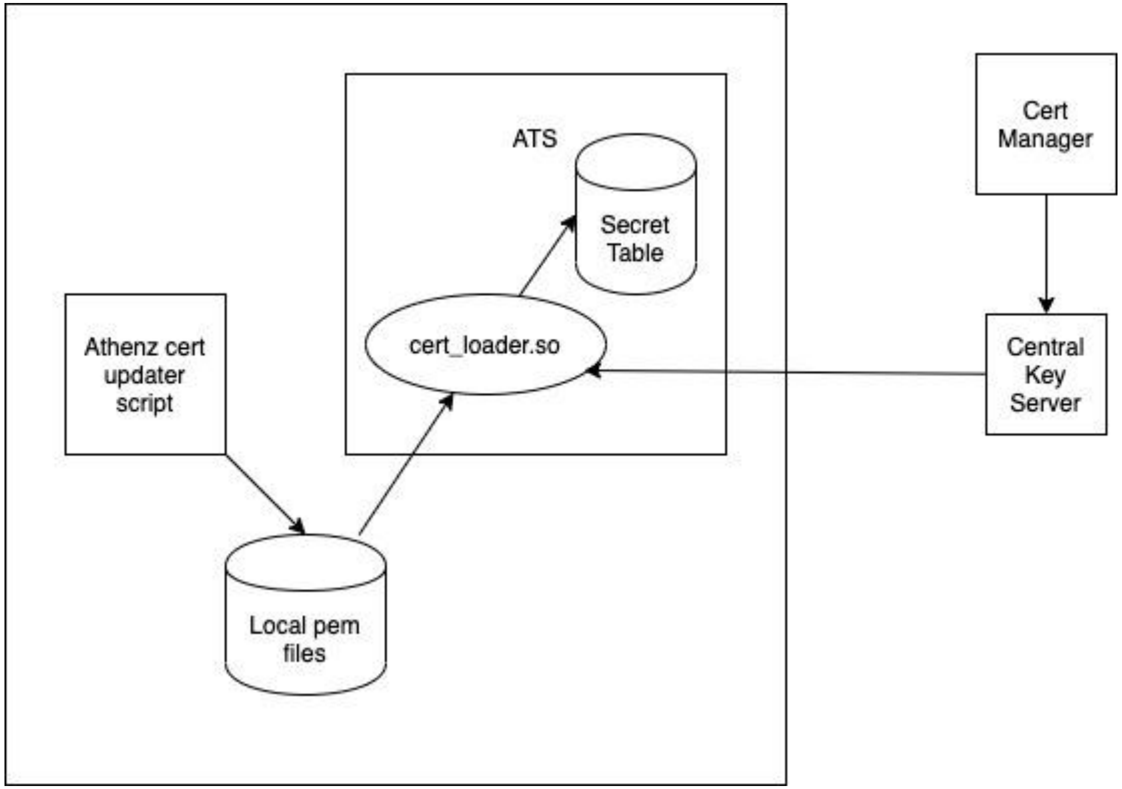
Limitations with previous approach

- Good approach for updates
- Not viable if cert/key is never on disk
  - Initial load won’t work
- Implementation did not sufficiently track the policy relationships for server certificates
  - E.g dual cert relationships
  - Our new implementation work still benefited from Zeyuan’s reorganization for safely updating SSL_CTX tables without a config reload.

Motivations

- For Server Certificates we need to pull keying information from a Central Key Server
    - Our cert management system places updated certs and keys in the Central Key Server
    - Auto-renewals get placed in the Central Key Server
- Need to automatically detect and load new certs and keys “hot loading”
    - Avoid need for explicit system reload or config start
        - Avoid another point of ops coordination
    - Athenz client certificates used for authentication/authorization lifetime less than a day
    - Server cert lifetimes are continually being reduced
        - Our default is now 30 days, and still aiming lower

ATS Cert Loader Block Diagram



New Plugin API Hook

Added the idea of a ***secret***. The secrets are used for the certificate and key PEM data.

Hook **TS_LIFECYCLE_SSL_SECRET_HOOK**, triggered on load/reload for each certificate/key referenced in the ATS policy (both client and server certs).

The names are the full path from the config. Plugin can use any subset of that name.

```
Int fetch_secret(TSCont cont, TSEvent event, void *edata) {
  TSSecretID *id = static_cast<TSSecretID *>(edata);
  std::string cert_name{id->cert_name, id->cert_name_len};
  std::string key_name{id->key_name, id->key_name_len};
```

New Plugin API’s for config load/reload

From the hook, the plugin can update the Secret state via **TSSslSecretSet**. When the hook returns, the data should be present for the core SSL_CTX creation. If the secret data is not present after calling the hook, the core reads from disk as before.

```
TSReturnCode
TSSslSecretSet(const char *secret_name, int secret_name_length, const char *secret_data, int
secret_data_len)
```

The plugin can also use **TSSslSecretGet** to see what is already in the Secret state for that name.

```
TSReturnCode
TSSslSecretGet(const char *secret_name, int secret_name_length, const char
**secret_data_return, int *secret_data_len)
```

New Plugin API for hot reload

Plugin can run periodically to look for new secret data.

When detected, Plugin updates the secret state with **TSSslSecretSet**.

In addition, it needs to signal to the core that it should recompute the associated SSL_CTX with **TSSslSecretUpdate**

- For client SSL_CTX, simply removes the current entry. On the next access a new SSL_CTX will be created with the new secret data
- For server SSL_CTX, the core looks at the associated policy and creates a new SSL_CTX to update the server SSL_CTX table entry.

Cert_loader plugin

Uses the new hook and TS API's to pull data from Central Key Server or disk. Plugin connects to Central Key Server and periodically pulls information about the registered key groups. On each hook call

- Take the secret names and removes the path to get the Central Key Server key names.
- If it is is found in Central Key Server, call TSSslSecretSet
- Otherwise, use the full path and look for the data on disk.
- If it is found on disk, call TSSslSecretSet

Save information about the secrets loaded from Central Key Server or disk with last version or file modification time.

Cert_loader Plugin Update

Periodically, refetch the Central Key Server key group info. Time interval is a plugin config option.

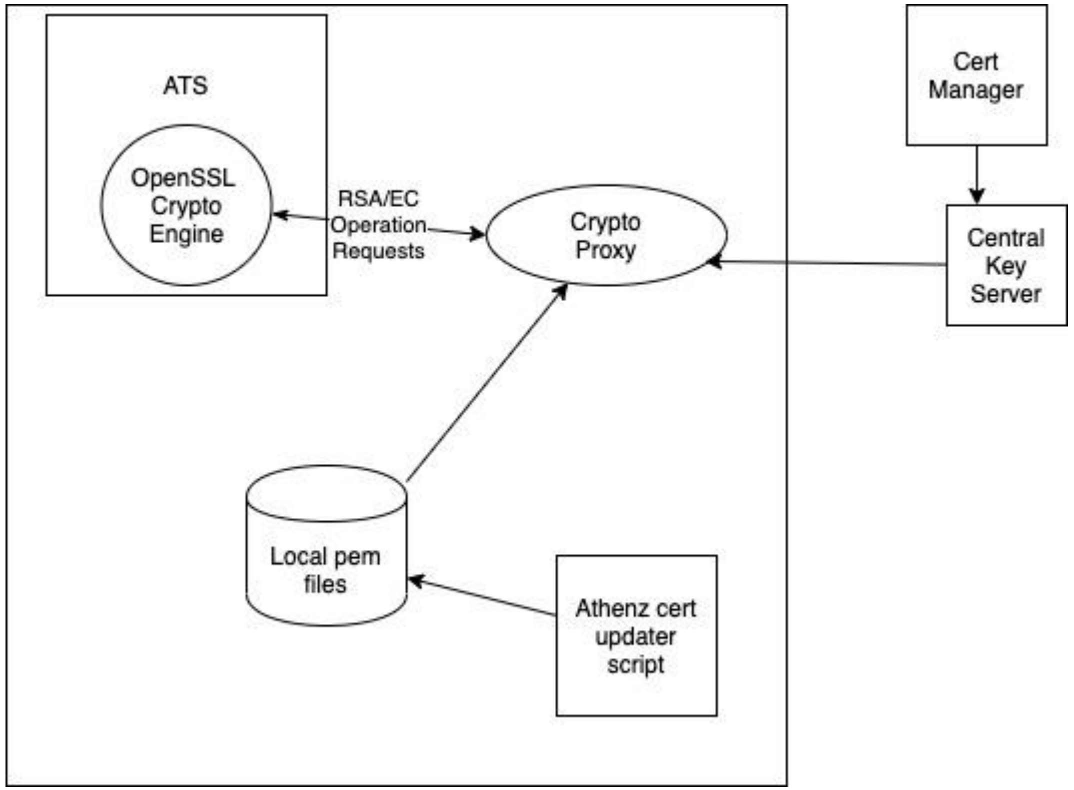
Look through the list of previously loaded secrets and see if there are newer versions in Central Key Server. If so call TSSslSecretSet and TSSslSecretUpdate.

Do the same thing with the list of secrets from disk. See if any of the files have newer modification times and call TSSslSecretSet and TSSslSecretUpdate as necessary.

Status

- PR for plugin updates https://github.com/apache/trafficserver/pull/6609
  - Need to update the PR
  - PR includes a test plugin to exercise the API via autest
- Vinith has tested both loading and updating from disk and Central Key Server
  - Getting ready for wider deployment with ATS9

Maybe Eventually - Move out to Crypto Proxy


ATS
OpenSSL Crypto Engine
RSA/EC Operation Requests
Crypto Proxy
Cert Manager
Central Key Server
Local pem files
Athenz cert updater script