X.509 Certificate-Based Authentication & Authorization

• What is X.509 Certificate-Base Authentication?
• Why do we want it, or need it?
• How we did it (Show-N-Tell)
• But ...!
What Is X.509 Certificate-Base Authentication

• Commonly known as 2-way SSL or TLS
• X.509 certificate is a digital identity
  • Binds a public key to an identity (i.e. user, device, etc)
  • Identity owns the corresponding private key
  • Issued/certified by a Certification Authority (CA)
  • Analogous to your driver license or passport
• Validation is based on mutual trust
  • Trusted Certification Authority (CA) hierarchy
  • Each party can trust a different CA hierarchy
  • Analogous to driver license issuer (i.e. State DMV)
• Public-Key Infrastructure (PKI) properties
  • Confidentiality, Integrity, Authentication, and Non-Repudiation
X.509 Certificate
Why Do We Want Or Need It?

• **Use Cases**
  • I want to be able to use my existing enterprise X.509 certificate to login to the cloud.
  • Our department mandates the use of Common Access Card (CAC) to authenticate to our private cloud.
  • All our users are managed in corporate LDAP directory, having to create and manage the service user accounts in our corporate LDAP directory may not work for us. It would be awesome if we can replace the service users with X.509 certificates.
  • We do not want our service users to be able to login to Horizon.
  • Oops did we just inadvertently show the admin password to our vendor’s support guy when we went through one of the conf files? It would be nice if we don’t have to put any passwords into the conf files.

• **Automation**
  • No need to put passwords in conf files.
  • Certificate can bind to a device (i.e. IP)

• **Something *You* Have**
  • Devices are becoming more personal (i.e. your smart phone, PDA, ultrabook, etc)
  • World is moving on from passwords

• **Single Credential For Multiple Clouds?**
  • Not exactly SSO, but similar experience
How We Did It

• Horizon Login
  • Based on WebSSO
  • No code changes. Configuration changes only.

• Keystone (auth_token) Middleware
  • Based on X509Tokenless auth plugin
  • [https://review.openstack.org/#/c/283905/](https://review.openstack.org/#/c/283905/)

• CLI and Lib
  • X509 auth plugin for keystoneauth (formerly python-keystoneclient)
  • Patch yet to be submitted upstream
Horizon Login

User's Browser → Horizon → Keystone:
1. Select "X.509 SSL Certificate" Login
2. HTTP Redirect to Keystone 2-Way SSL Endpoint
3. 2-Way SSL Handshake
4. JavaScript Code contains HTTP Redirect to Horizon
5. Federated Unscoped Token in Header
6. Execute JavaScript
7. Login with Unscoped Token
8. List User's Federated Projects
9. Federated Projects
10. Rescoped Token Request
11. Project Scoped Token
12. Login Completed
Step 1: Configure Keystone

```
[federation]
#
# From keystone
#
# Entrypoint for the federation backend driver in the keystone.federation
# namespace. (string value)
#driver = sql
#
# Value to be used when filtering assertion parameters from the environment.
# (string value)
#assertion_prefix =
#
# Value to be used to obtain the entity ID of the Identity Provider from the
# environment (e.g. if using the mod_shib plugin this value is 'shib-Identity-
# Provider'). (string value)
#remote_id_attribute = SSL_CLIENT_I_DN_CN
#
# A domain name that is reserved to allow federated ephemeral users to have a
# domain concept. Note that an admin will not be able to create a domain with
# this name or update an existing domain to this name. You are not advised to
# change this value unless you really have to. (string value)
#federated_domain_name = Federated
#
# A list of trusted dashboard hosts. Before accepting a Single Sign-On request
# to return a token, the origin host must be a member of the trusted_dashboard
# list. This configuration option may be repeated for multiple values. For
# example: trusted_dashboard=http://acme.com/auth/webss
# trusted_dashboard=http://beta.com/auth/webss
# trusted_dashboard=https://10.0.2.15/dashboard/auth/webss
trusted_dashboard = https://10.0.2.15/dashboard/auth/webss/
#
# Location of Single Sign-On callback handler, will return a token to a trusted
# dashboard host. (string value)
#sso_callback_template = /etc/keystone/sso_callback_template.html
```

```
[auth]
#
# From keystone
#
# Allowed authentication methods. (list value)
#methods = external,password,token,oauth1,x509
#x509 = keystone.auth.pluginsmapped.Mapped
```

/etc/keystone/keystone.conf
Step 2: Configure Apache

/etc/apache2/sites-available/keystone
Step 3: Create Service Provider, Protocol & Mapping

```bash
# create an identity provider for X.509
openstack identity provider create --remote-id "Signing CA" --description "For X.509 2-way SSL Single Sign-On" X509

# create the mapping
openstack mapping create --rules $PROJECT_HOME/mapping.json ssl_auth

# create the protocol
openstack federation protocol create --identity-provider X509 --mapping ssl_auth X509
```

```
{
    "local": [
        {
            "user": {
                "type": "local",
                "name": "{}",
                "domain": {
                    "name": "{}"
                }
            },
            "type": "local"
        }
    ],
    "remote": [
        {
            "type": "SSL_CLIENT_S_DN_CN"
        },
        {
            "type": "SSL_CLIENT_S_DN_O"
        },
        {
            "type": "SSL_CLIENT_I_DN",
            "any_one_of": []
        }
    ]
}
```

Certificate Viewer: "imported certificate"

This certificate has been verified for the following uses:

- SSL Server Certificate

- Common Name (CN) - Signing CA
- Organization (O) - Tenant
- Organizational Unit (OU) - Keystore
- Serial Number - 01

- Issued To
  - Common Name (CN) - Signing CA
  - Organization (O) - Tenant
  - Organizational Unit (OU) - Keystore

- Issued By
  - Common Name (CN) - Signing CA

- Period of Validity
  - Begin On: 04/13/2016
  - Expires On: 04/13/2018

- Fingerprints
  - SHA-256 Fingerprint: 9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B
  - SHA1 Fingerprint: 9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B:9B
Step 4: Create Domain, User, Project

```bash
# create the domain
openstack domain create Helion

# create the user in the domain, no password needed
openstack user create --enable --domain Helion demo

# create a project in the domain
openstack project create --enable --domain Helion demo_project

# assign some role to the user so he can do some damage
openstack role add --user demo --user-domain Helion --project demo_project --project-domain Helion admin
```
Step 5: Test With cURL
Step 6: Configure Horizon

```
OPENSTACK_API_VERSIONS={"identity":3}
OPENSTACK_KEYSTONE_URL="https://10.6.2.15:5000/v3"
OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
OPENSTACK_SSL_CACERT = '/etc/keystone/ssl/cacert.pem'
WEBSSO_ENABLED = True
WEBSSO_INITIAL_CHOICE = "credentials"
WEBSSO_CHOICES = (  
    ("credentials", "Keystone Credentials"),  
    ("x509", "X.509 SSL Certificate"),
)
AVAILABLE_REGIONS = [
    ('https://10.6.2.15:5000/v3', 'direct-keystone'),
    ('https://10.6.2.15:35358/v3', 'proxied-keystone'),
]
```
Walkthrough
Walkthrough
Walkthrough
Walkthrough
Keystone Middleware – Typical Deployment
Keystone Middleware – With Client SSL Auth
Walkthrough

curl -v -s -k -X POST \
--cert /home/sam/dev/demo/certs/x509client.cer \
--key /home/sam/dev/demo/certs/x509client.key \
--cacert /etc/apache2/ssl/apache.cer \
-H 'Content-Type: application/json' \
-d '{"auth": {
  "scope": {
    "project": {
      "domain": {
        "name": "HPE"
      },
      "name": "Identity"
    },
    "identity": {"x509": {},
      "methods": ["x509"]
    }
  }
},
https://10.0.2.15:5000/v3/auth/tokens
Walkthrough

```bash
openstack --os-auth-url https://10.0.2.15:5000/v3
  --os-cacert /etc/apache2/ssl/apache.cer
  --verify
  --os-cert /home/sam/dev/demo/certs/x509client.cer
  --os-key /home/sam/dev/demo/certs/x509client.key
  --os-auth-type x509
  --os-project-id a2651747f25e41cf980d79c8061a0fde
server list
```
Walkthrough

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Status</th>
<th>Networks</th>
</tr>
</thead>
<tbody>
<tr>
<td>61b6752c-7473-468c-acbc-e4965c730098</td>
<td>sam_server_2</td>
<td>ACTIVE</td>
<td>private=10.0.0.3</td>
</tr>
</tbody>
</table>
Walkthrough

/etc/nova/nova.conf

```
[keystone_auth_token]
memcached_servers = 10.0.2.15:11211
signing_dir = /var/cache/nova
certfile = /home/sam/dev/x509/x509service.cer
keyfile = /home/sam/dev/x509/x509service.key
cafile = /etc/apache2/ssl/apache.cer
auth_url = https://10.0.2.15:5000
auth_plugin = x509tokenlessauth
domain_name = HPE
```
Walkthrough

/etc/keystone/keystone.conf

[tokenless_auth]
trusted_issuer = emailAddress=service_signer@hpe.com,CN=service_signer,OU=Helion,O=HPE,ST=California,C=US
trusted_issuer = emailAddress=intermediateca@hpe.com,CN=intermediateca,OU=Engineering,O=HPE,ST=California,C=US

[auth]
methods = external,password,token,oauth1,x509
x509 = keystone.auth.plugins.x509.X509
But ...

- Deployment Considerations
  - SSL Termination
  - SSL Pass-Through
  - Hybrid Approach - SSL Termination at the LB, New SSL Connection to the Backend
  - Vendor considerations (i.e. Apache versus Nginx, HAProxy versus some specialized hardware on steroid)

- Operational Considerations
  - Certificate management versus password management tradeoffs
  - Provisioning, revocation, rotation, etc
  - Different classes of certificates

- Multi-Cloud, Multi-Region Considerations

- Keystone Limitations
  - remote_id_attribute, it should be managed programmatically instead of in static keystone.conf
  - Always map to an unscoped token. We should allow mapping to scoped token.
SSL Termination
Typical Deployment
Example: HAPProxy

frontend http-frontend
    #balance leastconn
    mode http
    option forwardfor
    bind 10.0.2.15:335358 ssl crt /etc/keystone/ssl/keystone-all.pem ca-file /etc/keystone/ssl/cacert.pem verify optional
    http-request set-header X-SSL %[ssl_fc]
    http-request set-header X-SSL-Client-Verify %[ssl_c_verify]
    http-request set-header X-SSL-Client-SHA1 %{+q}[ssl_c_sha1]
    http-request set-header X-SSL-Client-DN %{+q}[ssl_c_s_dn]
    http-request set-header X-SSL-Client-CN %{+q}[ssl_c_s_dn(cn)]
    http-request set-header X-SSL-Client-O %{+q}[ssl_c_s_dn(o)]
    http-request set-header X-SSL-Issuer %{+q}[ssl_c_i_dn]
    http-request set-header X-SSL-Issuer-CN %{+q}[ssl_c_i_dn(cn)]
    regadd X-Forwarded-Proto: \https if { ssl_fc }
default_backend keystone

backend keystone
    server keystone-public 10.0.2.15:5001 check
Example: Apache

```xml
<VirtualHost *:5001>
  WSGIDaemonProcess keystone-public-http processes=5 threads=1 user=gyee display-name=%{GROUP}
  WSGIScriptAlias / /usr/local/bin/keystone-wsgi-public
  WSGIApplicationGroup %{GLOBAL}
  WSGIPassAuthorization On
  ErrorLogFormat "%M"
  ErrorLog /var/log/apache2/keystone.log
  CustomLog /var/log/apache2/keystone_access.log keystone_combined

  ServerName ssldemo

  SetEnvIf X-SSL-Issuer "^(.*)$" SSL_CLIENT_I_DN=$0
  SetEnvIf X-SSL-Issuer-CN "^(.*)$" SSL_CLIENT_I_DN_CN=$0
  SetEnvIf X-SSL-Client-CN "^(.*)$" SSL_CLIENT_S_DN_CN=$0
  SetEnvIf X-SSL-Client-0 "^(.*)$" SSL_CLIENT_S_DN_O=$0
</VirtualHost>
```
SSL Pass-Through
Hybrid Approach
Resources

- [http://docs.openstack.org/developer/keystone/extensions/websso.html](http://docs.openstack.org/developer/keystone/extensions/websso.html)
- [http://docs.openstack.org/developer/keystonemiddleware/middlewarearchitecture.html](http://docs.openstack.org/developer/keystonemiddleware/middlewarearchitecture.html)
- [http://docs.openstack.org/developer/python-keystoneclient/authentication-plugins.html](http://docs.openstack.org/developer/python-keystoneclient/authentication-plugins.html)
- [http://docs.openstack.org/developer/keystone/](http://docs.openstack.org/developer/keystone/)
- [http://docs.openstack.org/](http://docs.openstack.org/)
Where To Get Help?

• IRC (freenode) - #openstack-keystone
• Mailing Lists (https://wiki.openstack.org/wiki/Mailing_Lists)
  • openstack-dev
    • Don’t forget to put the [keystone] tag in the subject.
  • openstack-operators
• Presenters (We accept beer vouchers!)
  • Guang Yee
    • guang.yee@hpe.com
    • IRC handle: gyee
  • Sam Leong
    • chio-fai-sam.leong@hpe.com
    • IRC handle: samleon