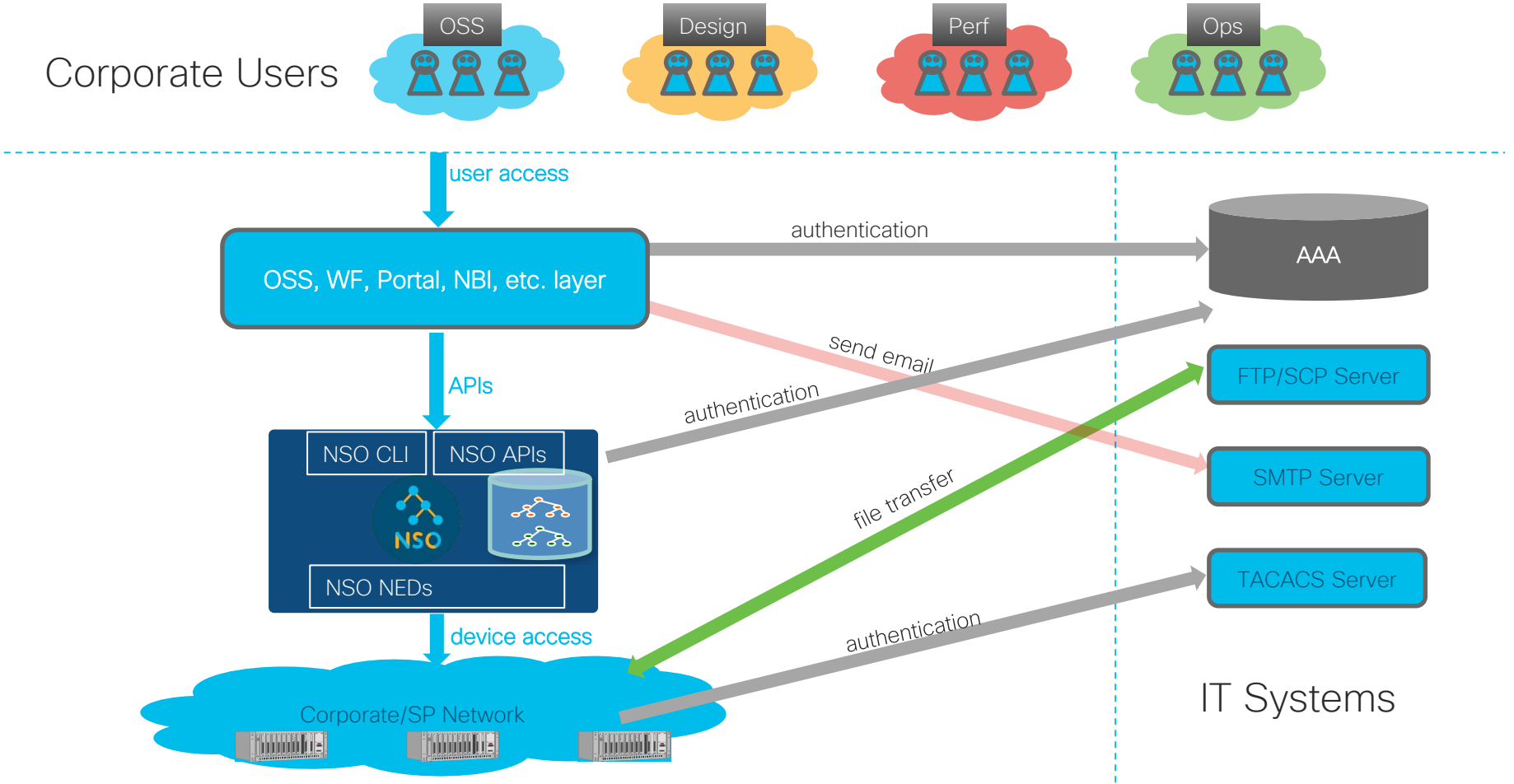
Developer Days
Network Services Orchestrator

NSO Access Control

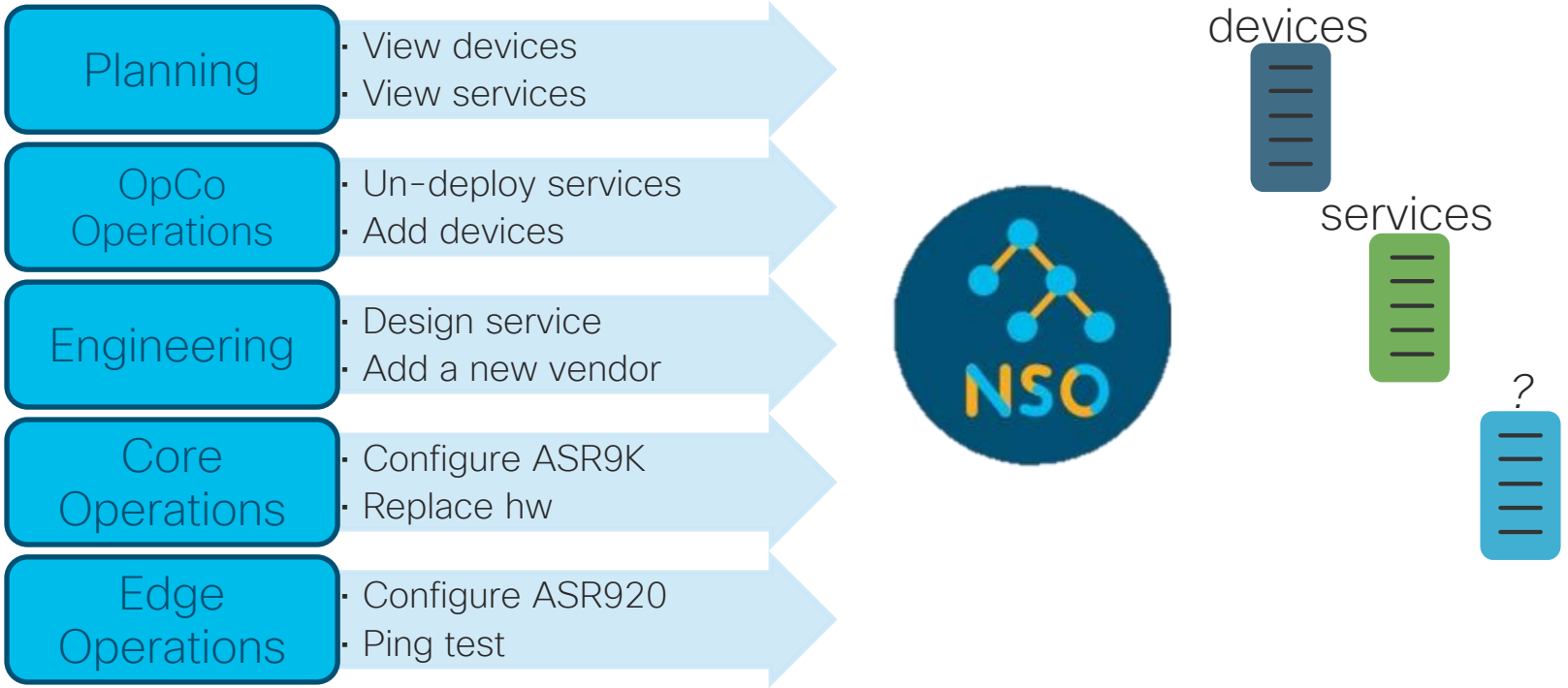
Role-based and Resource-based Access

Fatih Ayvaz
Software Architect, Cisco CX
16 June 2020

Deployment example



Different roles perform different tasks

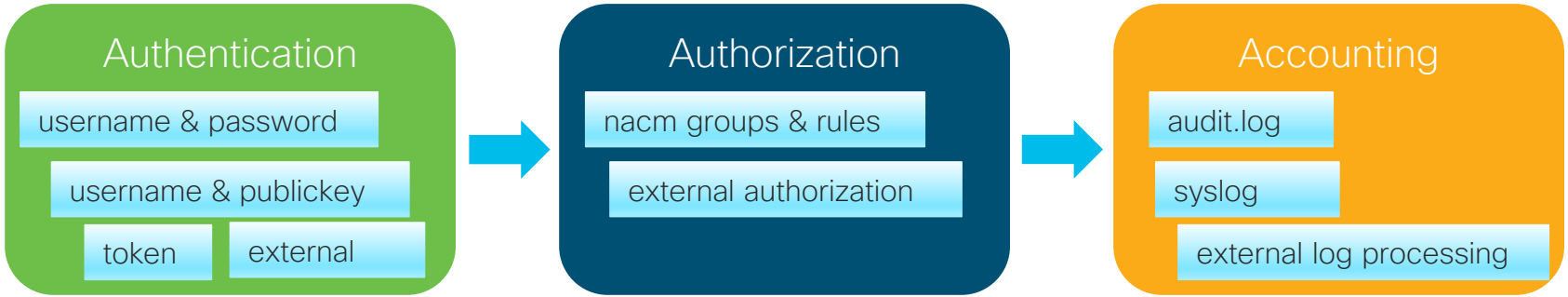


Who is going to access NSO?

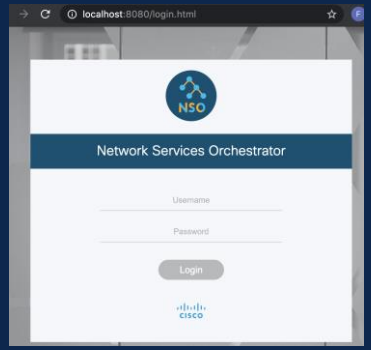
- What the the user roles to use NSO?
  - What are the permissions/restrictions of each role?
  - Is everyone allowed to access (read, write) all devices?
  - How about services?
- Which northbound interfaces will be used to access?
- What information will be required from user to grant access?
- Where are the users and groups associations stored? Which attributes?
- Will all the access interfaces be treated in same way?
- ..

NSO AAA

AAA functions in NSO



Access to NSO




```
FAAYVAZ-M-L3HD:ncs-run faayvaz$ ncs_cli

nsoadmin connected from 127.0.0.1 using
nsoadmin@ncs>
```

RESTCONF

NETCONF

Access Interfaces


```
int maapi_authenticate(int sock, const char *user, const char *pass,
char *groups[], int n);
```

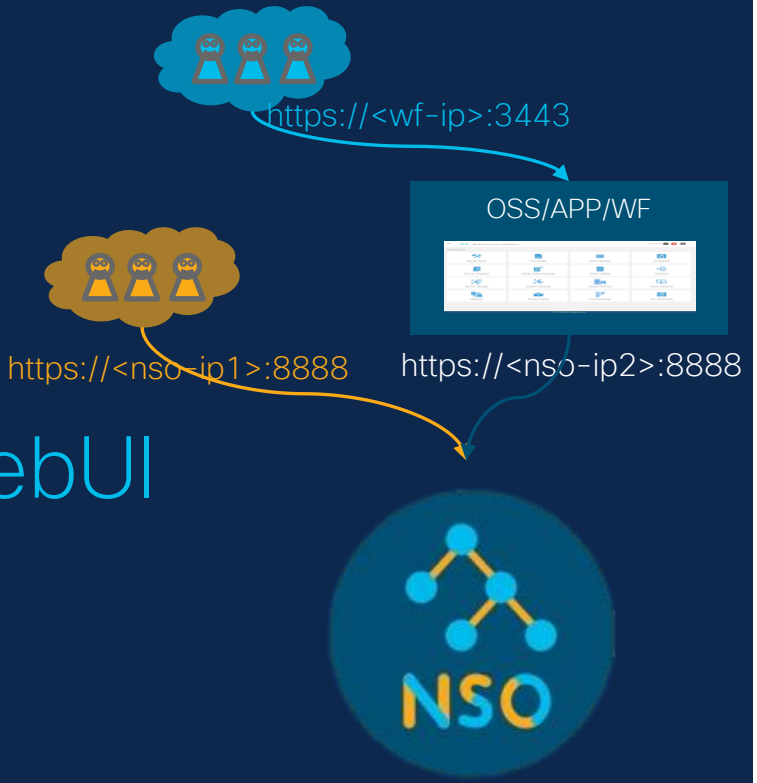
- CLI >> Console, SSH
- NETCONF >> TCP, SSH (built-in), SSH (OpenSSH)
- RESTCONF >> HTTP
- SNMP
- WEBUI >> HTTP, SSL
- maapi_authenticate()
  - >> username and password
  - maapi_authenticate2() >> + src_addr, src_port, context, and prot
  - * aaa/externalAuthentication/includeExtra

Built-in SSH Server

- Modeled in tailf-ncs-ssh.yang
- Supports DSA, RSA, EDDSA (ED25519) key types.
- Controls ssh host keys fetched from devices.
- Access methods using SSH:
  - NETCONF (TCP: 2022)
  - CLI (TCP: 2024)

Configure SSH

- SSH server keys
  - /ncs-config/aaa/ssh-server-key-dir: ${NCS_DIR}/etc/ncs/ssh
- Duration to close ssh session
  - /ncs-config/aaa/ssh-login-grace-time [PT10M]
- Max number of attempts to close ssh session
  - /ncs-config/aaa/ssh-max-auth-tries [unbounded]
- Public key authentication method
  - /ncs-config/aaa/ssh-pubkey-authentication: (none, local, *system)
- User ssh keys:
  - local: /aaa/authentication/users/user{$USER}/ssh_keydir
  - system: $HOME/.ssh



- SSL settings in ncs.conf
  - /ncs-config/webui/transport/ssl/enabled
  - /ncs-config/webui/transport/ssl/key-file (string)

```
<key-file>/etc/ncs/ssl/cert/nso_acme_com.key</key-file>
```

  - /ncs-config/webui/transport/ssl/cert-file (string)

```
<cert-file>/etc/ncs/ssl/cert/nso_acme_com.cer</cert-file>
```

  - /ncs-config/webui/transport/ssl/ca-cert-file (string)

```
<ca-cert-file>/etc/ncs/ssl/cert/CACert.cer</ca-cert-file>
```

  - /ncs-config/webui/transport/ssl/protocols (string)

```
<protocols>tlsv1.2</protocols>
```

- Verify server-side and client-side certs with openssl!

```
NSO$ openssl s_client -connect nso.acme.com:8888 -cert
nso_acme_com.cer -key nso_acme_com.key
```

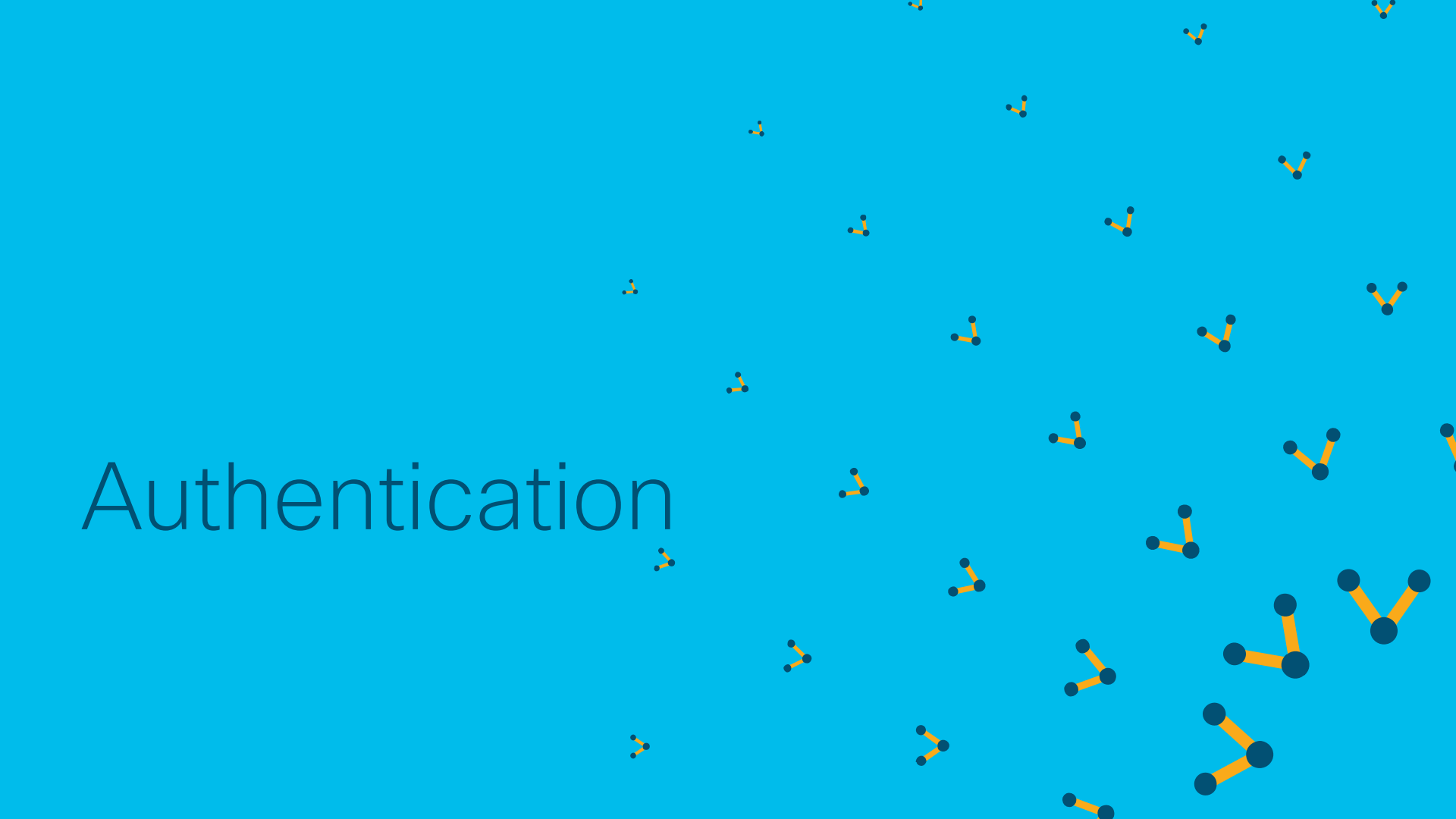
```
WF$ openssl s_client -connect wf.acme.com:3443 -cert
/opt/wf/wf_acme_com.cer -key /opt/wf/wf_acme_com.key
```

- INSTALL client-side certificates!
- Edit /etc/hosts entries for hostnames!

IPC Access

- Client libraries connect. E.g.: ncs_cli, ncs_load, netconf-subsys, etc.
- Users with shell access are trusted (by default)
  - User must be in a linux group which is allowed.
- Configuration options
  - /ncs-config/ncs-ipc-address/ip (ipv4-address | ipv6-address) [127.0.0.1]
  - /ncs-config/ncs-ipc-address/port (port-number) [4569]
- Restricting IP access:
  - /ncs-config/ncs-ipc-access-check/enabled (boolean) [false]
  - /ncs-config/ncs-ipc-access-check/filename (string)
  - The file should be protected via OS file permissions.
  - Client should set environment variable NCS_IPC_ACCESS_FILE.
  - IMPORTANT! if this is set, and ipc-access-check is disabled, client connection will fail!

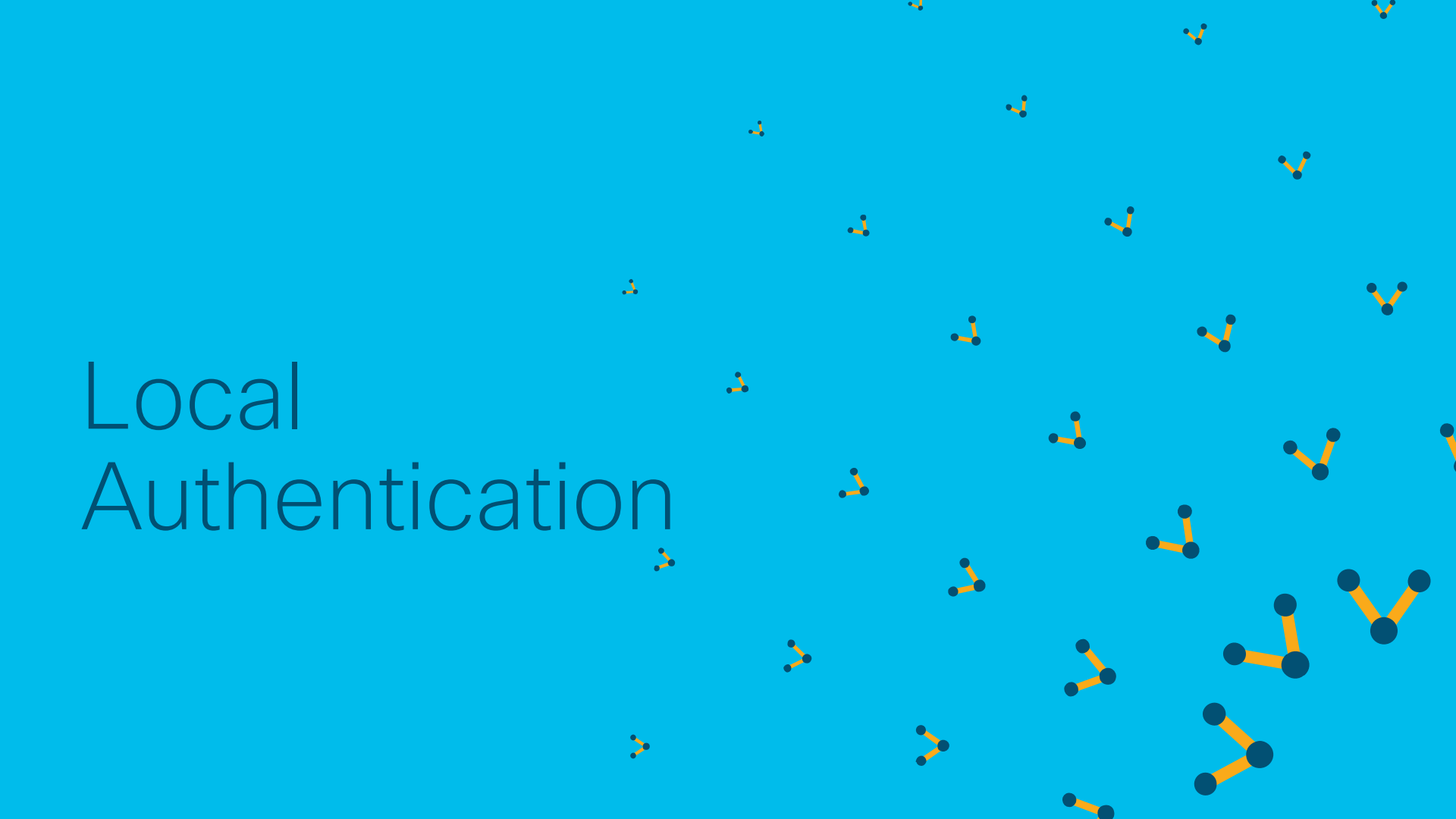
Authentication



Authentication

- Username and password
  - CLI, NETCONF, RESTCONF, SNMP, WebUI
  - External authentication, PAM, local authentication
  - Authentication order in ncs.conf: e.g.
    ```
    <auth-order>external-authentication
    local-authentication</auth-order>
    ```
- Public key
  - CLI, NETCONF
- Token validation
  - RESTCONF
  - External validation

Local Authentication



Local Authentication

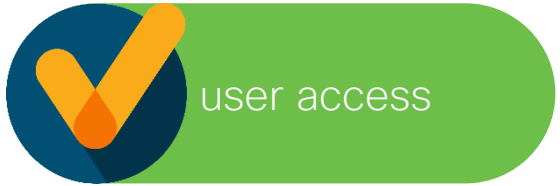

```
  <aaa>
    <ssh-server-key-dir>${NCS_DIR}/etc/ncs/ssh</ssh-server-key-
dir>
    <!-- Depending on OS - and also depending on user
requirements -->
    <!-- the pam service value value must be tuned. -->
    <pam>
      <enabled>true</enabled>
      <service>common-auth</service>
    </pam>
    <external-authentication>
      <enabled>false</enabled>
      <executable>my-test-auth.sh</executable>
    </external-authentication>

    <local-authentication>
      <enabled>true</enabled>
    </local-authentication>

  </aaa>
```

AAA (default for local install) configuration in CDB

```
admin@ncs> show configuration aaa authentication users user admin
uid         65534;
gid         65534;
password
$6$EUPCDnuhJwIFZk9c$m/IKonkOTNm0KbeRb4BTIUsq9I6XrKNbvd3UKowava904mWdbVxvT7C/X8aAKgwb598mrHwS05ewyc5f
/pQfU1;
ssh_keydir /var/ncs/homes/admin/.ssh;
homedir    /var/ncs/homes/admin;
[ok][2020-06-09 12:30:59]
admin@ncs> show configuration aaa authentication users user oper
uid         65534;
gid         65534;
password
$6$Ey9GYFlUcF3TgkZY$rlx5OteS.bRXfzxouX7EEunzKgz5.xR2TWWxsYCR3wKwBKjOJhz7BN68OLulSEk8VRjHhynMskzFR/Sb
zyJYd1;
ssh_keydir /var/ncs/homes/oper/.ssh;
homedir    /var/ncs/homes/oper;
[ok][2020-06-09 12:35:28]
admin@ncs>
```

Verification of username/password authentication

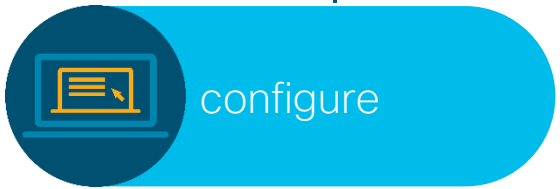

```
ncs> show configuration aaa authentication users user admin
password   $6$EUPCDnuhJwIFZk9c$m//pQfU1;
ssh_keydir /var/ncs/homes/admin/.ssh;
```



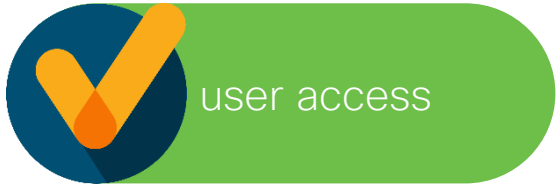

```
$ ssh admin@localhost -p 2024
admin@localhost's password:****

admin connected from 127.0.0.1 using ssh on FAAYVAZ-M-J0S2
admin@ncs>
```


```
[withheld]/0 login failed via cli from 127.0.0.1:57488 with ssh:
Couldn't read "/var/ncs/homes/admin/.ssh/authorized_keys2": "no
such file or directory"
admin/0 local authentication succeeded via cli from 127.0.0.1:57488
with ssh, member of groups: admin
admin/0 logged in via cli from 127.0.0.1:57488 with ssh using local
authentication
admin/50 assigned to groups: admin
admin/50 created new session via cli from 127.0.0.1:57488 with ssh
```

Disable public key for oper




```
admin@ncs% set aaa authentication users user oper ssh_keydir ""
```




```
$ ssh oper@localhost -p 2024
oper@localhost's password:****

oper connected from 127.0.0.1 using ssh on FAAYVAZ-M-J0S2
oper@ncs>
```


```
<INFO> .. audit user: oper/0 local authentication succeeded via
cli from 127.0.0.1:58713 with ssh, member of groups: oper
<INFO> .. audit user: oper/0 logged in via cli from
127.0.0.1:58713 with ssh using local authentication
<INFO> .. audit user: oper/54 assigned to groups: oper
<INFO> .. audit user: oper/54 created new session via cli from
127.0.0.1:58713 with ssh
```

Public Key Authentication

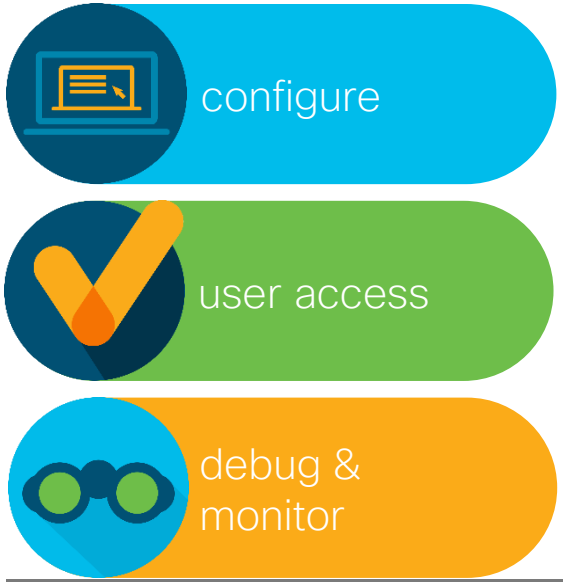
Enable public key for a “local user"

```
admin@ncs% set aaa authentication users user faayvaz ssh_keydir "/Users/faayvaz/.ssh"
Value for 'uid' (<int>): 65534
Value for 'gid' (<int>): 65534
Value for 'password' (<hash digest string>): ****
Value for 'homedir' (<string>): "/Users/faayvaz"
admin@ncs% commit
```

```
cat /Users/faayvaz/.ssh/id_rsa.pub >> /Users/faayvaz/.ssh/authorized_keys

FAAYVAZ-M-J0S2: faayvaz$ ssh faayvaz@localhost -p 2024
faayvaz connected from 127.0.0.1 using ssh on FAAYVAZ-M-J0S2
faayvaz@ncs>
```

```
<INFO> 9-May-2020::12:57:52.973 FAAYVAZ-M-J0S2 ncs[95259]: audit user: faayvaz/0 logged in via cli
from 127.0.0.1:60007 with ssh using publickey authentication
<INFO> 9-May-2020::12:57:52.981 FAAYVAZ-M-J0S2 ncs[95259]: audit user: faayvaz/58 assigned to groups:
<INFO> 9-May-2020::12:57:52.981 FAAYVAZ-M-J0S2 ncs[95259]: audit user: faayvaz/58 created new session
via cli from 127.0.0.1:60007 with ssh
```

Enable public key for a non-local user




```
admin@ncs% delete aaa authentication users user faayvaz
admin@ncs% commit
```

```
$ ssh faayvaz@localhost -p 2024

faayvaz connected from 127.0.0.1 using ssh on FAAYVAZ-M-J0S2
faayvaz@ncs>
```

```
<INFO> 9-May-2020::13:43:01.390 FAAYVAZ-M-J0S2 ncs[95259]: audit user: faayvaz/0 logged in via cli from 127.0.0.1:63124 with ssh using publickey authentication
<INFO> 9-May-2020::13:43:01.392 FAAYVAZ-M-J0S2 ncs[95259]: audit user: faayvaz/61 assigned to groups: _appserveradm,staff,admin,_appserverusr,_lpadmin
<INFO> 9-May-2020::13:43:01.392 FAAYVAZ-M-J0S2 ncs[95259]: audit user: faayvaz/61 created new session via cli from 127.0.0.1:63124 with ssh
```

External Authentication

External authentication

- LDAP, RADIUS, TACACS
- Python scripting

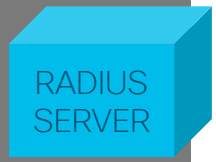
```
<aaa>
  <ssh-server-key-dir>${NCS_DIR}/etc/ncs/ssh</ssh-server-key-dir>
  <external-authentication>
    <enabled>true</enabled>
    <executable>/Users/faayvaz/NSO-5.3-20200108/external-authentication-for-demo.py</executable>
  </external-authentication>
  <auth-order>external-authentication local-authentication</auth-order>
</aaa>
```

```
$ ls -al /Users/faayvaz/NSO-5.3-20200108/external-authentication-for-demo.py
lrwxr-xr-x  1 faayvaz  staff  77 Jan 13 13:18 /Users/faayvaz/NSO-5.3-20200108/external-
authentication-for-demo.py -> /Users/faayvaz/Documents/CiscoLive/CL2020/external-authentication-for-
demo.py
```

External authentication with RADIUS

- Add NSO IP address as a RADIUS NAS client on RADIUS server
- Get the radius secret

```
Cisco Access Registrar 5.1.0.10 Configuration Utility
--> cd /radius/clients
--> add nso-15 "" radius <NSO-IP> cisco NAS
--> cd /radius/userlists/nsoUsers/
--> add test5 "" test5 TRUE "" prNSO1
--> ls -R test5
[ test5 ]
    Name = test5
    Password = <encrypted>
    BaseProfile~ = prNSO1
--> ls -R /radius/profiles/prNSO1
[ /Radius/Profiles/prNSO1 ]
    Name = prNSO1
    Description =
    Attributes/
          Callback-Id = "ncsoper 1010 500 501 502"
--> save
```




```
<external-authentication>
  <enabled>true</enabled>
  <executable>${NCS_RUN_DIR}/radauth.py</executable>
  rnal-authentication>
```

```
o-faayvaz:~$ ssh test5@<NSO-IP> -p 2024
test5@172.16.13.15's password:
test5 connected from 172.16.13.15 using ssh on nso-
faayvaz
test5@ncs>


--- audit.log ---
<INFO> audit user: test5/0 Logged in over ssh using
externalauth, member of groups: ncsoper
<INFO> audit user: test5/97 assigned to groups: ncsoper
```

radauth.py


```
def get_credentials():
  # read username and password from stdin
  # comes in [username;password;]\n format

  #Remove [ and ], split on ; and assign to username and password
  username = c.replace("\\n","").strip('"[]').split(";")[0]
  password = c.replace("\\n","").strip('"[]').split(";")[1]
  return (username, password)

def check_credentials(username, password):
   radhost = 'RADIUSSERVER:1812'
   radsecret = 'radius-secret'
   # build and send radclient command
   radcommand = 'echo \"User-Name=%s,User-Password=%s\" | radclient %s auth %s' %
(username,password,radhost,radsecret)
   radresponse = subprocess.Popen(radcommand, stdout=subprocess.PIPE, shell=True)
   (reply, err) = radresponse.communicate()

   if reply.find("Callback-Id") > 0 :
       authinfo = reply.split('"')
       accept = "accept %s /home/%s" % (authinfo[1],username)
       acceptstr = "Returning: %s\n" % accept
       return accept
```

External authentication with TACACS

- Add NSO IP address as a TACACS client
- Get the tacacs shared key

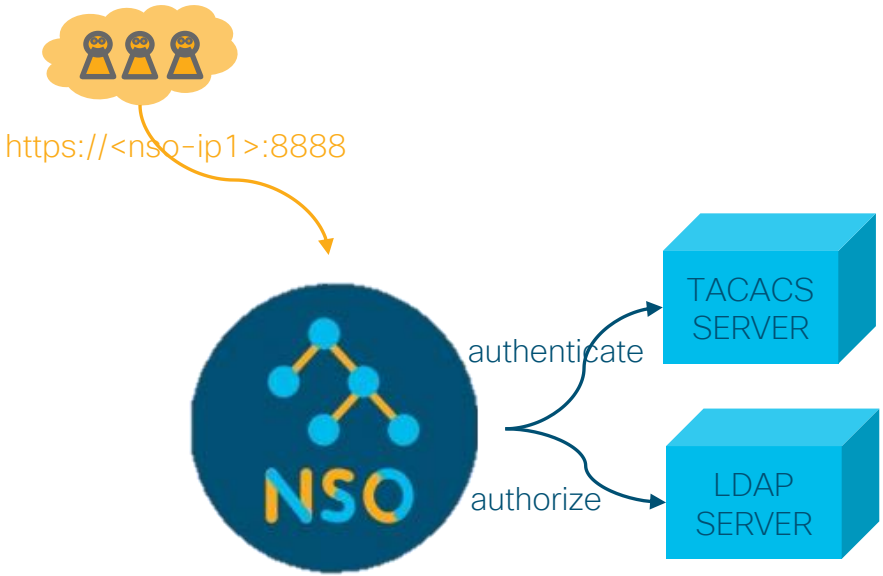
```
'cisco-av-pair=shell:domains = group1'
```


```
<external-authentication>
  <enabled>true</enabled>
  <executable>${NCS_RUN_DIR}/authTACACS.py</executable>
</external-authentication>
```

authTACACS.py

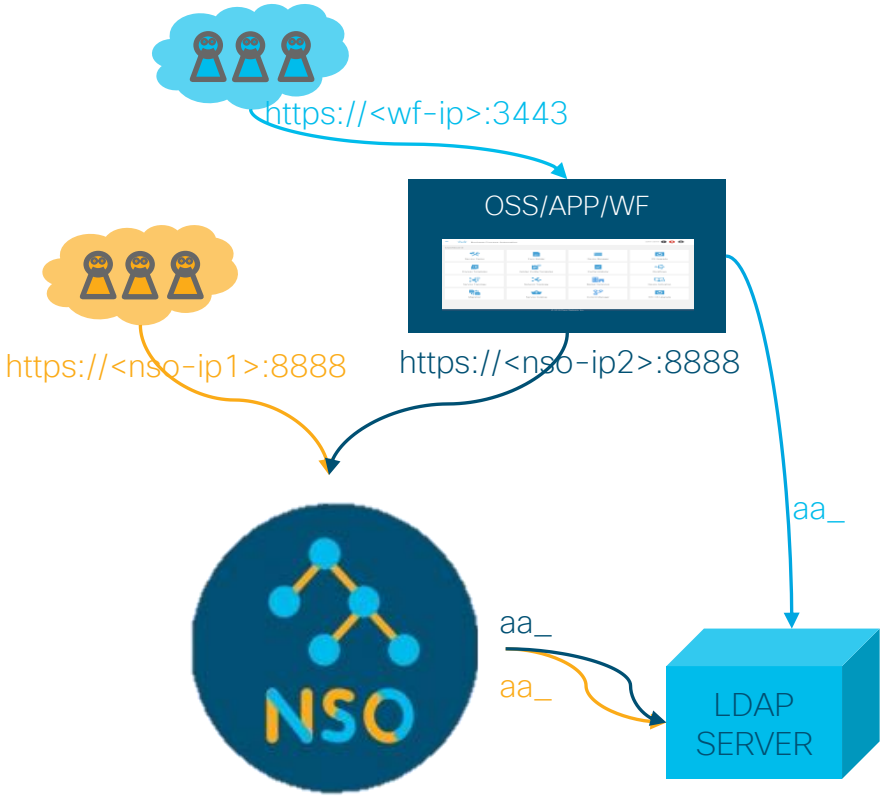

```
credentialstring = sys.stdin.readline()[:-2][1:]
credentials = credentialstring.split(';’)
user = credentials[0]
password = credentials[1]

tacacs_host = '10.A.B.C'
shared_key = '***'
cli = TACACSClient(tacacs_host, 49, shared_key,
                           timeout=60, family=socket.AF_INET)
authenticate = cli.authenticate(username, password)

if authenticate.valid:
   response = cli.authorize(username, arguments=cmds)
   regex = re.compile(r".*domains=(\S+)")
   groups += regex.search(arg).groups()[0] + " "
   accept = "accept {} 1004 1004 /opt/ncs/ncs-run\n".format(groups)
   print(accept)
else:
    print "reject invalid password\n"
```

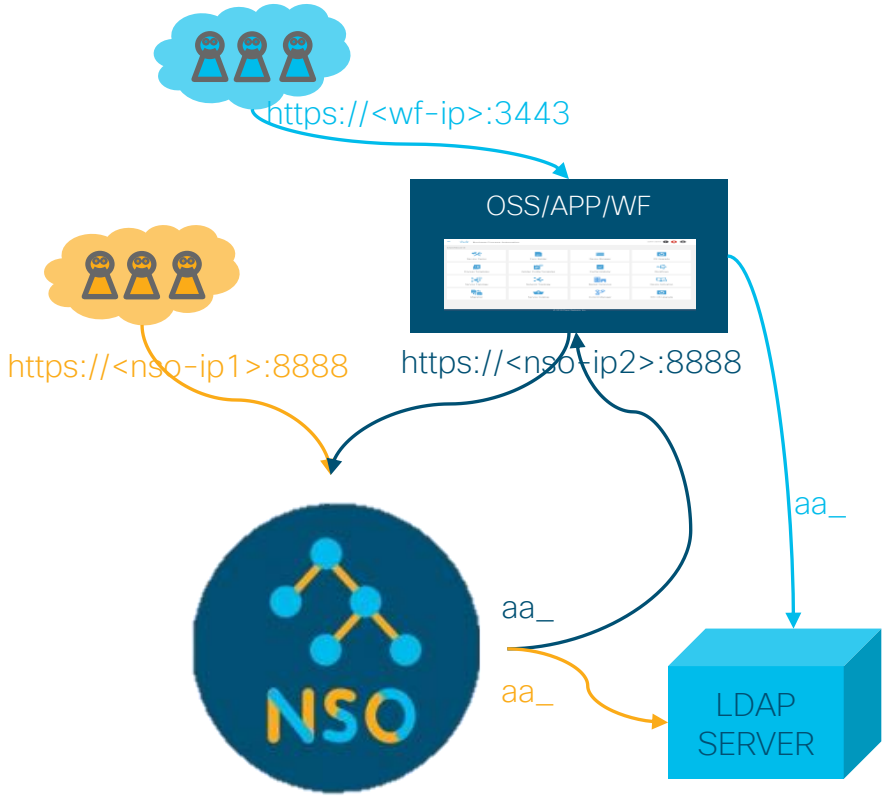
External authentication with multiple servers



- What is the requirement to authenticate/authorize with another server?
- When can this be useful?

How about this?





When authentication happens

- External authentication (with a custom python script) via LDAP, RADIUS, TACACS:
  - "accept $groups $uid $gid $supplementary_gids $HOME\n”
  - To return a token: "accept_token $groups $uid $gid $supplementary_gids $HOME $token\n”
  - Other options: accept_info, accept_warning, accept_token_info, accept_token_warning
- External token validation (RESTCONF only):
  - "accept $groups $uid $gid $supplementary_gids $HOME $USER\n”
  - To return a token: "accept_token $groups $uid $gid $supplementary_gids $HOME $USER $token\n”
  - Other options: accept_info, accept_warning, accept_token_info, accept_token_warning
- Monitor in audit.log file:
  - demouser1/0 logged in via netconf from 127.0.0.1:55779 with ssh using publickey authentication
  - nsoadmin/0 logged in via webui from 127.0.0.1:52135 with http using local authentication
  - demouser1/0 logged in via rest from 127.0.0.1:55363 with http using externalvalidation authentication

When authentication fails

- If authentication/validation fails:
  - “reject”: will try next method (in auth-order or validation-order)
  - “abort”: fails immediately!
  - /ncs-config/aaa/audit-user-name (always | known | never) [known]

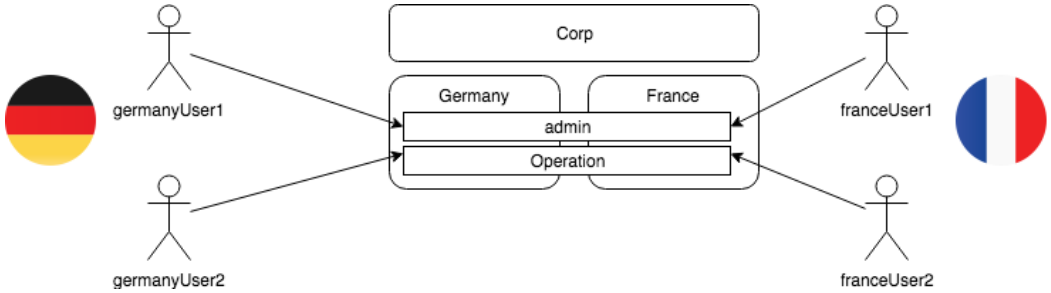
```
try:
   ldap_client.simple_bind_s(ldap_username, password)
except ldap.INVALID_CREDENTIALS:
    if username == ”privileged-user":
       print("reject INVALID_CREDENTIALS: privileged-user : invalid ldap credentials")
    else:
       print("abort INVALID_CREDENTIALS: invalid ldap credentials")
except ldap.CONSTRAINT_VIOLATION:
    print("reject CONSTRAINT_VIOLATION: ldap constraint violation")
except ldap.SERVER_DOWN:
    print("reject SERVER_DOWN: ldap server not accessible")
except ldap.LDAPError:
    print("abort LDAPError")
```

NSO Authorization

Authorization on NSO

- What is authorized and how?
  - Authorization of commands (cmdrule)
    - CLI and WebUI commands and operations
  - Authorization of data access (rule)
    - RPC
    - Notifications
    - Data nodes
  - Group membership
    - Role based authorization
    - /nacm/groups
    - /etc/group
    - (+) any group from authentication
    - /ncs-config/aaa/default-group, if empty and set so!
  - NACM (RFC 8341)
    - Hyperlink: https://tools.ietf.org/html/rfc8341
  - Tail-f ACM
    - CLI commands
    - Support for ”context”
  - Tail-f NCS ACM
    - NACM options for services

Example scenario: multi-tenant users/groups



LDAP Structure

Group	User
Corp	corpUser1 corpUser2
Germany	germanyUser1 germanyUser2
France	franceUser1 franceUser2

Group membership: /etc/passwd

```
[nsoadmin@nso-01 ncs]$ grep nsoadmin /etc/passwd
nsoadmin:x:1000:1000::/home/nsoadmin:/bin/bash
[nsoadmin@nso-01 ncs]$ grep 1000 /etc/group
nsoadmin:x:1000:
[nsoadmin@nso-01 ncs]$ id
uid=1000(nsoadmin) gid=1000(nsoadmin) groups=1000(nsoadmin)
```

```
[nsoadmin@nso-01 ncs]$ ncs_cli
nsoadmin connected from 10.1.1.1 using ssh on nso-01
nsoadmin@nso-s1> conf
                 ^
% Invalid input detected at '^' marker.
nsoadmin@nso-s1> <TAB>
Possible completions:
  exit - Exit the management session
  quit - Exit the management session
nsoadmin@nso-s1>
-- audit.log --
<INFO> 29-May-2020::06:46:50.621 nso-01 ncs[1305]: audit user: nsoadmin/471065 assigned to groups: nsoadmin
```

Group membership: /etc/passwd & /nacm/groups

```
[nsoadmin@nso-01 ncs]$ ncs_cli -u admin
admin connected from 10.1.1.1 using ssh on nso-01
admin@nso-s1> show configuration nacm groups group ncs
Possible completions:
  ncsadmin  ncsoper
admin@nso-s1> show configuration nacm groups group ncsadmin
user-name [ admin private root system ];
```

```
-- audit.log --
<INFO> 29-May-2020::06:46:50.621 nso-01 ncs[1305]: audit user: nsoadmin/471065 assigned to groups: nsoadmin

<INFO> 29-May-2020::06:49:49.243 nso-01 ncs[1305]: audit user: admin/471112 assigned to groups: ncsadmin,nsoadmin
<INFO> 29-May-2020::06:50:24.818 nso-01 ncs[1305]: audit user: admin/471112 CLI 'show configuration nacm groups group
ncsadmin'
```

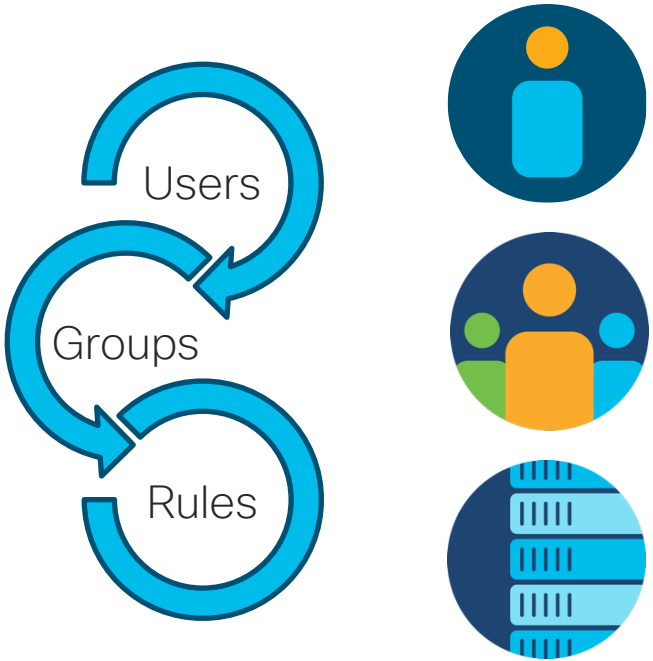
External group assignments can be disabled

```
admin@nso-s1% set nacm enable-external-groups false
admin@nso-s1% commit
```

```
[nsoadmin@nso-01 ncs]$ ncs_cli
nsoadmin connected from 10.1.1.1 using ssh on nso-01
nsoadmin@nso-s1> <TAB>
Possible completions:
  exit - Exit the management session
  quit - Exit the management session
nsoadmin@nso-s1> exit
[nsoadmin@nso-01 ncs]$ ncs_cli -u admin
admin connected from 10.1.1.1 using ssh on nso-01
admin@nso-s1> exit

-- audit.log --
<INFO> 29-May-2020::07:04:42.344 nso-01 ncs[1305]: audit user: nsoadmin/471345 assigned to groups:
<INFO> 29-May-2020::07:04:52.737 nso-01 ncs[1305]: audit user: admin/471348 assigned to groups: ncsadmin
```

NACM Overview

Cisco NSO's AAA Model



- AAA Users
- NACM Groups
- NACM Rule-lists
- NACM Rule Structure
  - Enablement
  - Rule-lists
  - Path statements

NACM Rule Types

- Module Rule: e.g. -> module-name = id-allocator
  - Controls access for definitions in a specific YANG module, identified by its name.
- Protocol Operation Rule: e.g. -> rpc-name = edit-config
  - Controls access for a specific protocol operation, identified by its YANG module and name.
- Data Node Rule: e.g. -> path = /devices/device[name='devIOS-0']
  - Controls access for a specific data node and its descendants, identified by its path location within the conceptual XML document for the data node.
- Notification Rule: e.g. -> notification-name = sys-config-change
  - Controls access for a specific notification event type, identified by its YANG module and name.

NSO cmdrule

- Where do these apply?
  - CLI commands

```
cmdrule c-logout action deny
  command logout
!
cmdrule j-logout action deny
  command "request system logout"
 !
```

  - WebUI functions and operations

```
cmdrule permit-jsonrpc-action action permit
  command "::jsonrpc:: action"
cmdrule permit-jsonrpc-run_action action permit
  command "::jsonrpc:: run_action"
cmdrule permit-jsonrpc-logout action permit
  command "::jsonrpc:: logout”
cmdrule permit-jsonrpc-delete action permit
  command "::jsonrpc:: delete"
```

```
augment /nacm:nacm/nacm:rule-list {
    list cmdrule {
      key "name";
      ordered-by user;

      leaf name {

      leaf context {
        default "*";

      leaf command {
        default "*";

      leaf access-operations {
         default "*";

      leaf action {
         mandatory true;

      leaf log-if-permit {

      leaf comment {
    }
  }
```

Authorization order

- For cmdrule
  - Check /nacm/enable-nacm
  - Traverse rule-list to match groups
    - Traverse cmdrule rules
  - if read; check /nacm/cmd-read-default
  - if exec; check /nacm/cmd/exec-default

- For rule
  - Check /nacm/enable-nacm
  - Traverse rule-list to match groups
    - Traverse rule rules
  - Check NACM extensions in data models:
    - *: data model: nacm:default-deny-all
    - CUD: data model: nacm:default-deny-write
  - if read; check /nacm/read-default
  - if CUD; check /nacm/write-default
  - if exec; check /nacm/exec-default

Order is important!

```
set nacm rule-list rdemogroup35 group demogroup3

set nacm rule-list rdemogroup35 cmdrule request-message-deny command "request message" action deny
access-operations exec

set nacm rule-list rdemogroup35 cmdrule all-cmd-any action permit context * log-if-permit
```

```
demouser3@ncs> request message nsoadmin hi
[ok][2020-05-29 14:19:06]
demouser3@ncs>
nsoadmin@ncs>
Message from demouser3@FAAYVAZ-M-J0S2 at 2020-05-29 14:19:06...
hi

<DEBUG> 29-May-2020::14:19:06.232 User: demouser3[demogroup3,demogroupN] Command Rule "rdemogroup3/all-cmd-any" triggered
full_match accept for "request message nsoadmin hi" op execute

nsoadmin@ncs% move nacm rule-list rdemogroup35 before rdemogroup3
nsoadmin@ncs% commit
demouser3@ncs> request message nsoadmin hi
Aborted: permission denied
[error][2020-05-29 14:22:29]

<DEBUG> 29-May-2020::14:22:29.114 User: demouser3[demogroup3,demogroupN] Command Rule "rdemogroup35/all-cmd-any" triggered
full_match accept for "request message" op read
<DEBUG> 29-May-2020::14:22:29.116 User: demouser3[demogroup3,demogroupN] rejected command "request message nsoadmin hi" op
execute by full_match Command Rule "rdemogroup35/request-message-deny"
```

NACM Default Rule Behavior

When no groups are found (no rule-lists to check) or no matching rules …

- nacm:default-deny-all
- nacm:default-deny-write
- read-default [permit]
- write-default [deny]
- exec-default [permit]
- *cmd-read-default [permit]
- *cmd-exec-default [permit]
- *log-if-permit-default

NSO NACM Rule Format

- Module Name [*]
  - The name of the YANG module where the requested data node is defined.
- Rule Type
  - rpc-name / notification-name / path: If data-node, then path must be checked.
  - “path” (yang:xpath1.0;}: The leaf "path" is an instance-identifier. You should not refer to non-key leafs.
- Access Operations [*]
  - create, *read, update, delete, **exec
    - *read: MUST be permitted, if a **notification** is tied to the node.
    - **exec: MUST be permitted, if an **action** is requested.
- Action
  - permit, deny
- Comment
- *Context
- *Log-if-permit

Module rule example

- The modules loaded:

  admin@ncs> **show status netconf-state capabilities capability**

  capability http://cisco.com/ciscoutils?module=ciscoutils;

  capability http://com/example/l3vpn?module=l3vpn;

  ...

- Deny l3vpn module:
  - **set nacm rule-list rdemogroup3 rule l3vpn-module-deny module l3vpn action deny access-operations ***
  - or:

```
<rule-list>
    <name>rdemogroup3</name>
      <rule>
        <name>l3vpn-module-deny</name>
        <module-name>l3vpn</module-name>
        <access-operations>*</access-operations>
        <action>deny</action>
      </rule>
  </rule-list>
```

```
demouser3@ncs> show vp{hit TAB to auto-complete for vpn}
- - devel.log- -
<DEBUG> 29-May-2020::01:21:56.945 FAAYVAZ-M-J0S2 ncs[24451]: devel-aaa User:
demouser3[demogroup3,demogroupN] rejected data access path /l3vpn:vpn op read due to rule
"rdemogroup3/l3vpn-module-deny"
```

RPC rule example

- Check rpc from netconf capabilities. E.g.: edit-config, delete-config, kill-session
- Deny get-config:

  set nacm rule-list rdemogroup33 group demogroup3

  set nacm rule-list rdemogroup33 rule get-config-deny rpc-name get-config action deny access-operations *

  set nacm rule-list rdemogroup33 cmdrule all-cmd-any action permit context * log-if-permit

```
faayvaz$ netconf-console -u demouser3 -p cisco --host localhost --get-config -x "/devices/device/authgroups"
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <rpc-error>
    <error-type>protocol</error-type>
    <error-tag>access-denied</error-tag>
    <error-severity>error</error-severity>
  </rpc-error>
</rpc-reply>
FAAYVAZ-M-J0S2:yang-explorer faayvaz$

-- devel.log -
<DEBUG> 29-May-2020::01:43:36.813 FAAYVAZ-M-J0S2 ncs[24451]: devel-aaa User:
demouser3[demogroup3,demogroupN] rejected data access path /nc:get-config op execute due to rule
"rdemogroup33/get-config-deny"
```

Usage of “path”

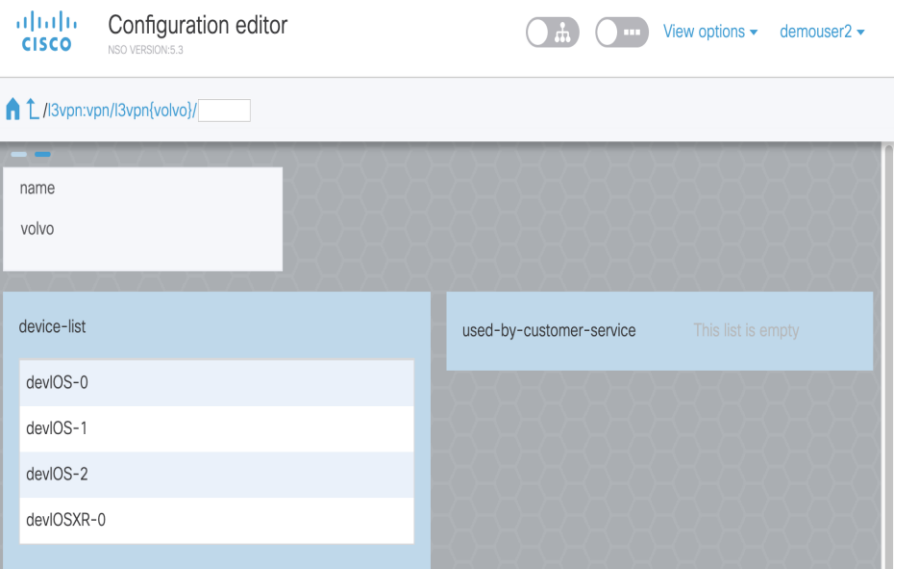
- Tagpaths that are not containing any keys
  - E.g., /ncs/live-device/live-status
- Instantiated key
  - E.g., /devices/device[name=“devIOS-0”]/config/interface
  - E.g., /devices/device/config/interface[name=”eth0”]
- Wildcard at end
  - E.g., /services/web-site/*

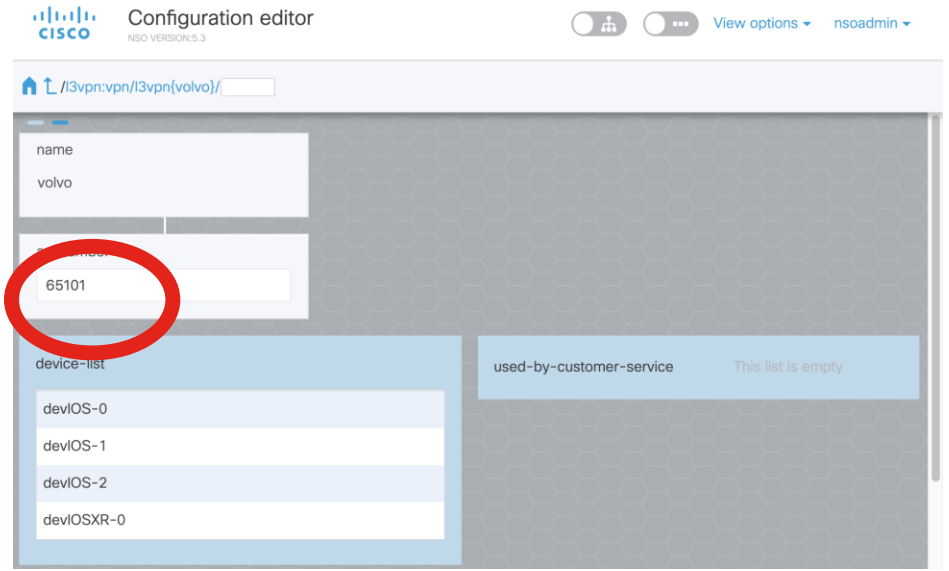
Example for path statement with no-key

- With no-key:

  set nacm rule-list rdemogroup2 group demogroup2

  set nacm rule-list rdemogroup2 rule l3vpn-asn-deny action deny context webui path /vpn/l3vpn/as-number access-operations read





Example for path statement with key

- A specific data node:

  set nacm rule-list device_devIOS rule d_TE412mtu_P_R path /devices/device[name='devIOS-0']/config/ios:interface/TenGigabitEthernet[name='4/1/2’]/mtu access-operations read action permit log-if-permit

<DEBUG> 19-May-2020::12:53:29.923 FAAYVAZ-M-J0S2 ncs[4816]: devel-aaa User: demouser1[demogroup1,demogroupN] Rule "device_devIOS/d_TE412mtu_P_R" triggered data access accept for path /ncs:devices/device{devIOS-0}/config/ios:interface/TenGigabitEthernet{4/1/2}/mtu op read

Example for path statement with wildcard

- A wildcarded list of elements:

  set nacm rule-list rdemogroup2 group demogroup2

  set nacm rule-list rdemogroup2 rule l3vpn-asn-deny action deny context webui path /vpn/l3vpn/as-number access-operations read

  set nacm rule-list rdemogroup2 rule l3vpn-ford-wc-permit action permit context webui path /vpn/l3vpn[name='ford']/* access-operations * log-if-permit

  set nacm rule-list rdemogroup2 rule l3vpn-ford-permit action permit context webui path /vpn/l3vpn[name='ford'] access-operations * log-if-permit

```
<DEBUG> 28-May-2020::22:54:41.264 FAAYVAZ-M-J0S2 ncs[24451]: devel-aaa User:
demouser2[demogroup2,demogroupN] Rule "rdemogroup2/l3vpn-ford-permit" triggered data access accept for path
/l3vpn:vpn/l3vpn{ford}/qos op read
<DEBUG> 28-May-2020::22:54:41.265 FAAYVAZ-M-J0S2 ncs[24451]: devel-aaa User:
demouser2[demogroup2,demogroupN] Rule "rdemogroup2/l3vpn-ford-wc-permit" triggered data access accept for
path /l3vpn:vpn/l3vpn{ford}/qos/qos-policy op read
```

Thank you!

NSO Access Control

Role-based and Resource-based Access

Fatih Ayvaz
Software Architect, Cisco CX
16 June 2020


CISCO