Bifrost Installation and Configuration

Content

Introduction

The Bifrost Middleware translates secure authenticated REST API calls from external components such as the SecureX Orchestrator to calls against on-prem devices, currently ISE over pxGrid and ERS API, Stealthwatch Enterprise over its API, or the Active Directory over LDAP

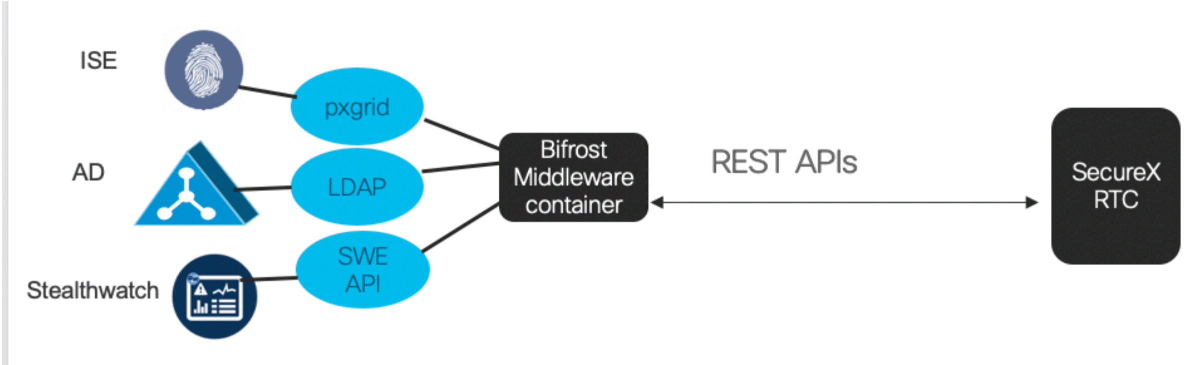


Figure 1 Bifrost Middleware - Overview

Currently the following calls are available

getUserByIP

Retrieves ISE user info given IP address

getUserInfoByIP

Retrieves information from ISE and Active Directory given an IP address

getUserInfoByUser

Retrieves information from Active Directory given a username

getFlowsByIP

Retrieves flows from Stealthwatch Enterprise given an IP address

getANCpolicies

Retrieves the ANC policies defined on ISE

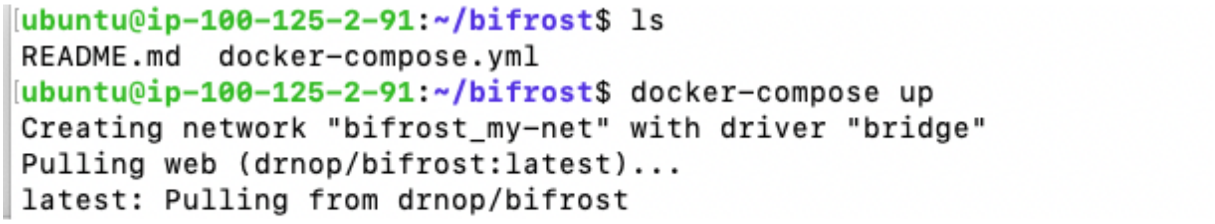
setANCpolicy

Sets (or clears) and ANC policy given IP or MAC address.

The APIs are documented below.

Installation

On a machine with docker and docker-compose installed, download the docker-compose.yml file from the github repository and run **docker-compose**.



```
ubuntu@ip-100-125-2-91:~/bifrost$ ls
README.md  docker-compose.yml
ubuntu@ip-100-125-2-91:~/bifrost$ docker-compose up
Creating network "bifrost_my-net" with driver "bridge"
Pulling web (drnop/bifrost:latest)...
latest: Pulling from drnop/bifrost
```

Figure 2 Installation with docker-compose

Configuration

1. Login to Bifrost Middleware Web GUI

Login to the middleware by browsing to https://<your ip address>. You will have to accept certificate warning.

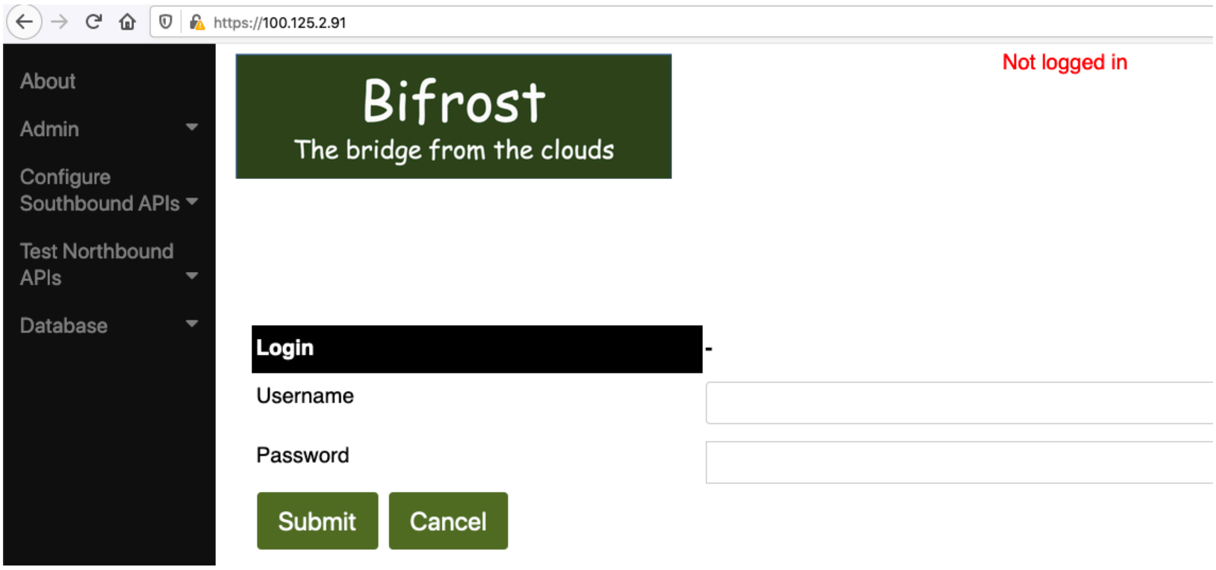


Figure 3 Logging in to the Web UI

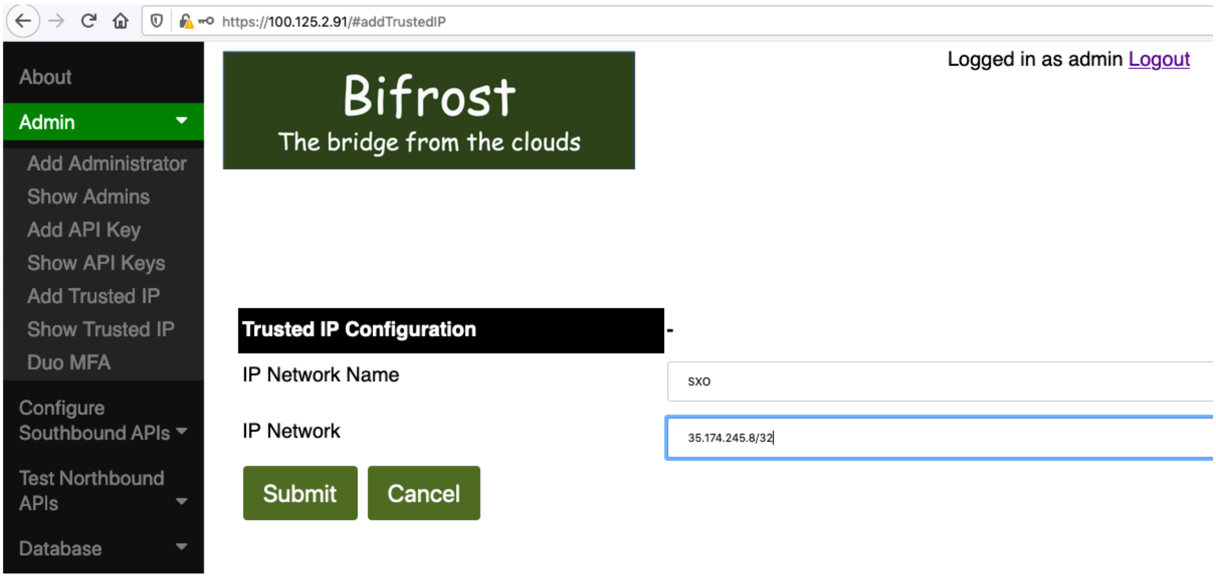
Username admin.
Password C!sco123.

2. Hardening: Configure Trusted Networks.

By default, all IP addresses can connect to the middleware. Change this by

- Add a trusted network for your admins (e.g. bastion hosts)
- Add a trusted network for your SXO
- Delete the default network

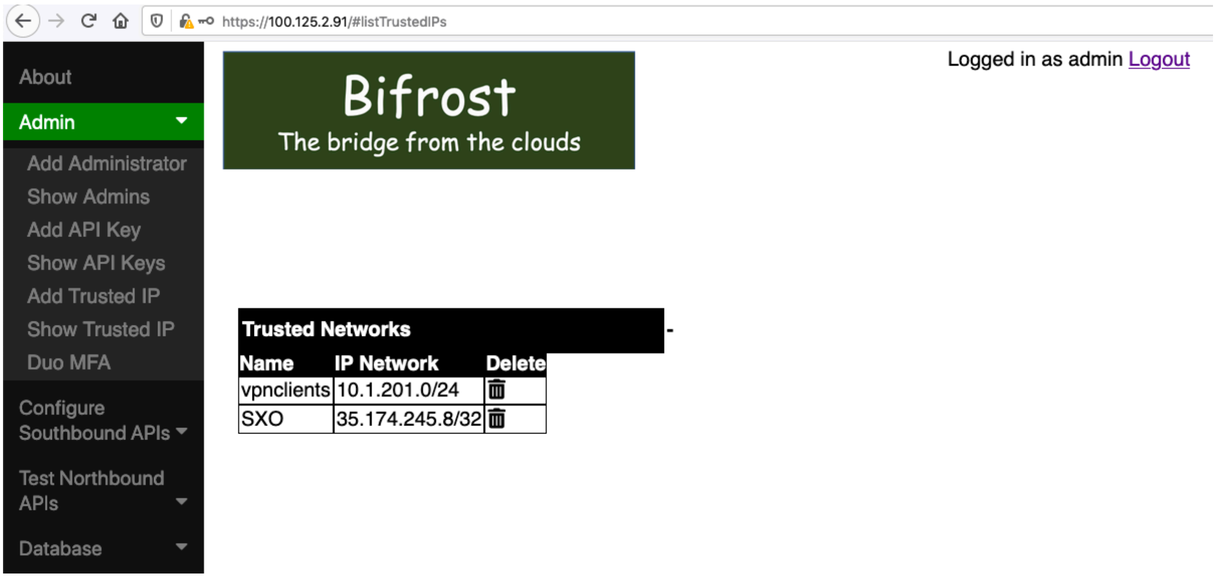
You can find out the public IP of the SXO by creating a trivial workflow that connects to a REST API such as ipify.com



Trusted IP Configuration

IP Network Name	sxo
IP Network	35.174.245.8/32

Figure 4 Adding a Trusted IP



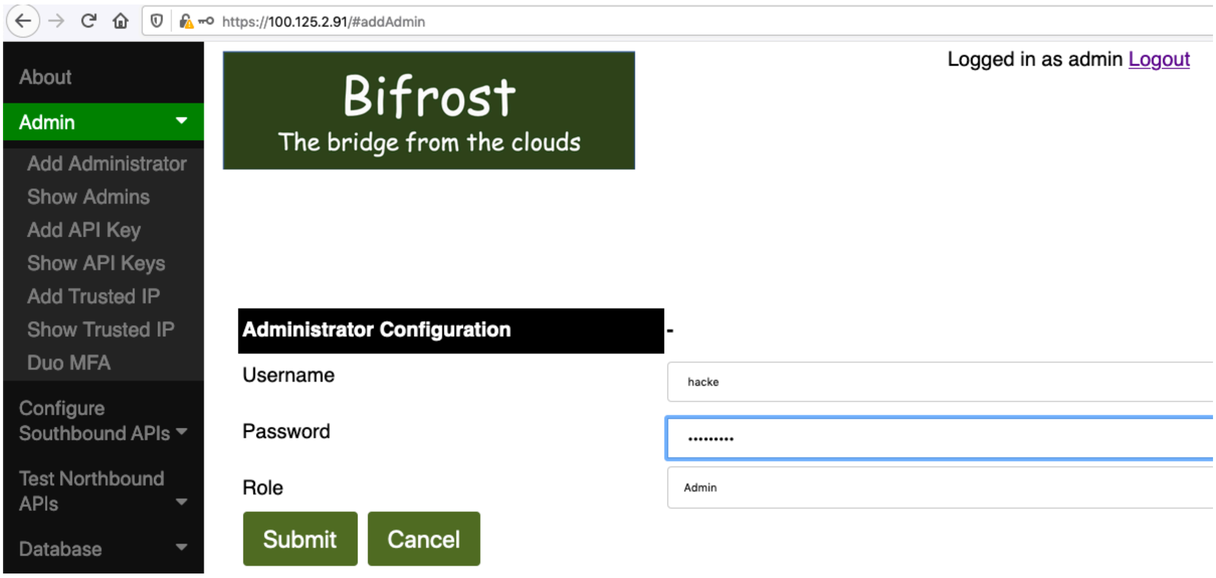
Trusted Networks

Name	IP Network	Delete
vpnclients	10.1.201.0/24	🗑
SXO	35.174.245.8/32	🗑

Figure 5 After adding trusted IPs and deleting default

3. Hardening: Add New Admin User and default admin

Add a new admin user with a password.



https://100.125.2.91/#addAdmin

Administrator Configuration -

Username: hacke

Password: ••••••••••

Role: Admin

Figure 6 Adding a new admin user

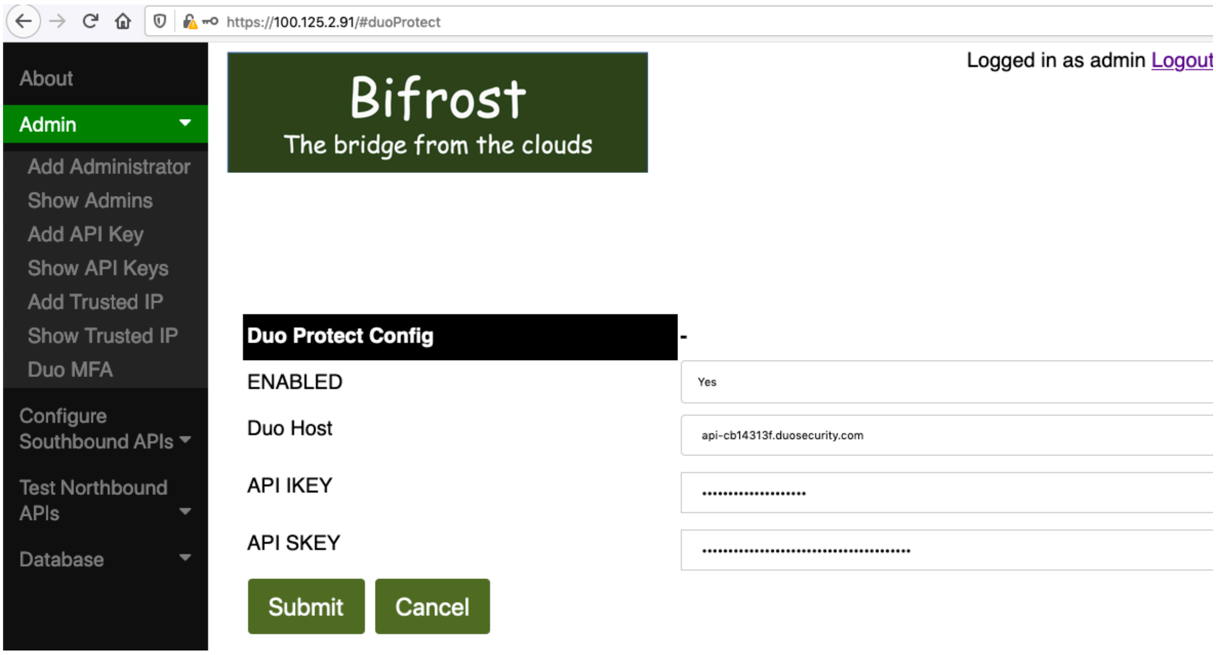
Delete the old admin user (admin).

It is advisable to first logout and test login as the newly created user, before deleting admin account. If you want to apply Duo MFA as in step 5, it is best to wait until after Duo MFA has been implemented and tested before deleting the admin user.

4. Hardening: Configure Duo Protect to enable MFA for users.

The middleware supports MFA with Duo.

Enter the duo host, I-KEY and S-KEY (from the Duo Portal configuration).



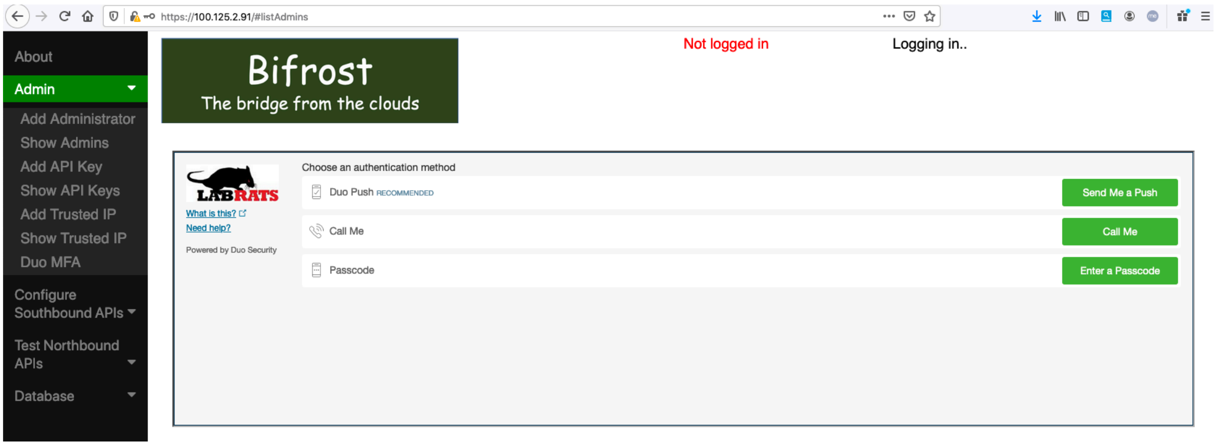
Logged in as admin Logout

Duo Protect Config -

ENABLED	Yes
Duo Host	api-cb14313f.duosecurity.com
API IKEY	•••••••••••••••••••••
API SKEY	•••

Figure 7 Duo Protect Configuration

After configuring Duo Protect, all users will require to use MFA except any user called “admin”. You can test by logging out and in again with any username except “admin”.



Not logged in

Logging in..

Choose an authentication method

Duo Push RECOMMENDED

Call Me

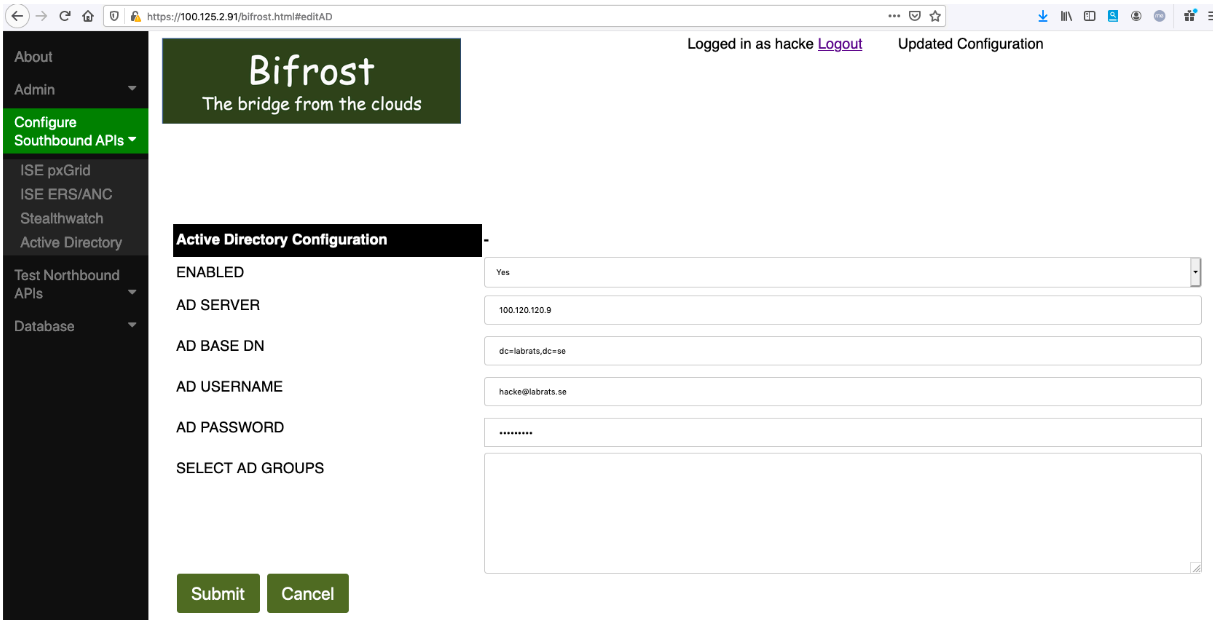
Passcode

Figure 8 Logging in with Duo MFA

5. Configure and Test the Active Directory Interface

Configure the Active Directory interface with

- Enabled Yes
- Server IP
- Base DN
- The username and password of a user that can browse the Active Directory



Active Directory Configuration -

ENABLED: Yes

AD SERVER: 100.120.120.9

AD BASE DN: dc=labrats,dc=se

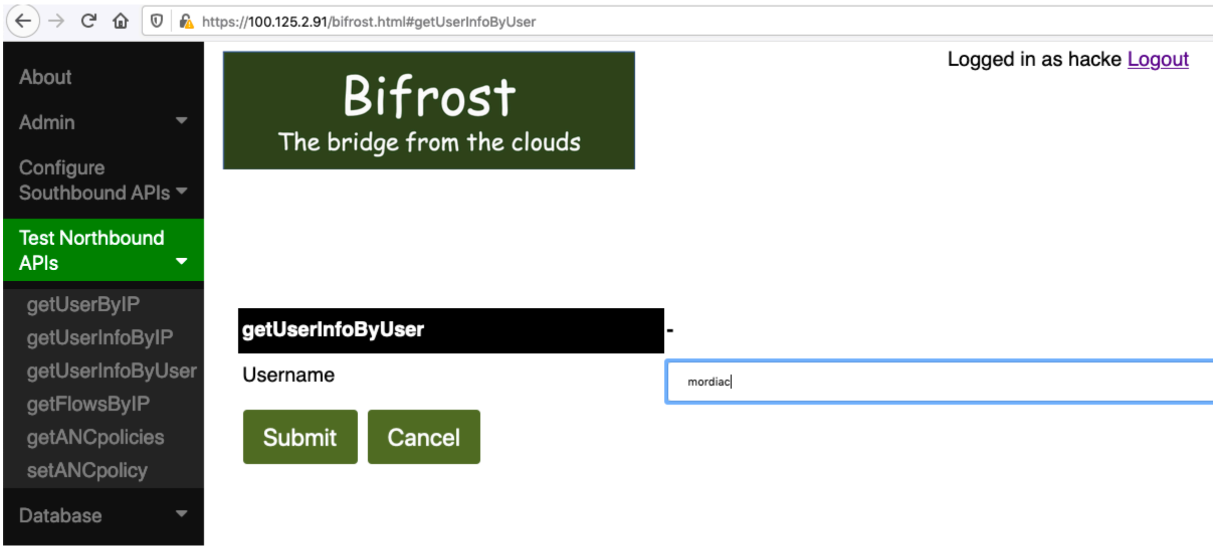
AD USERNAME: hacke@labrats.se

AD PASSWORD: ••••••••••

SELECT AD GROUPS:

Figure 9 Configuring Active Directory

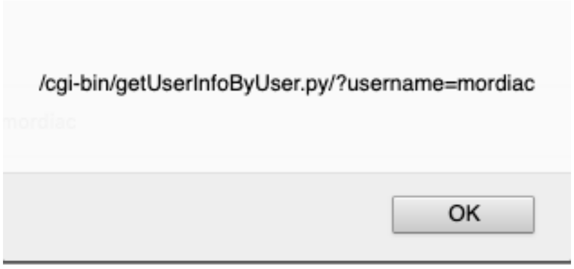
You can test the Active Directory Interface by searching for a user that exists in thee active directory. Test Northbound APIs/getUserInfoByUser and specify a valid username in the AD. You will get an alert showing the API request, and the response returned from the middleware.



getUserInfoByUser -

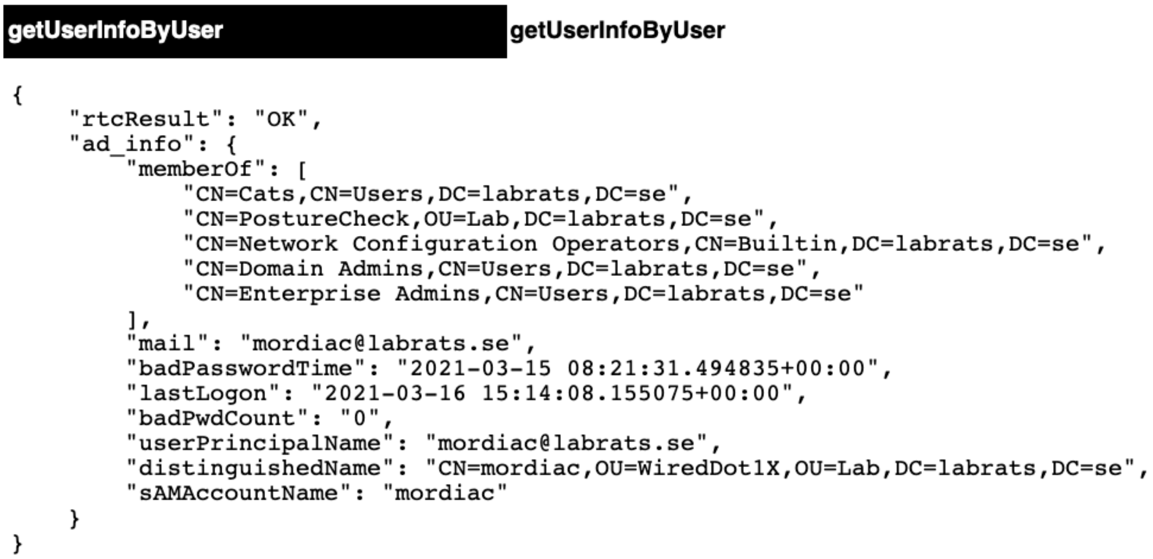
Username: mordiac

Figure 10 Test AD configuration



/cgi-bin/getUserInfoByUser.py/?username=mordiac

mordiac



getUserInfoByUser getUserInfoByUser

```
{
    "rtcResult": "OK",
    "ad_info": {
        "memberOf": [
            "CN=Cats,CN=Users,DC=labrats,DC=se",
            "CN=PostureCheck,OU=Lab,DC=labrats,DC=se",
            "CN=Network Configuration Operators,CN=Builtin,DC=labrats,DC=se",
            "CN=Domain Admins,CN=Users,DC=labrats,DC=se",
            "CN=Enterprise Admins,CN=Users,DC=labrats,DC=se"
        ],
        "mail": "mordiac@labrats.se",
        "badPasswordTime": "2021-03-15 08:21:31.494835+00:00",
        "lastLogon": "2021-03-16 15:14:08.155075+00:00",
        "badPwdCount": "0",
        "userPrincipalName": "mordiac@labrats.se",
        "distinguishedName": "CN=mordiac,OU=WiredDot1X,OU=Lab,DC=labrats,DC=se",
        "sAMAccountName": "mordiac"
    }
}
```

Figure 11 Test AD result (displaying groups and other info)

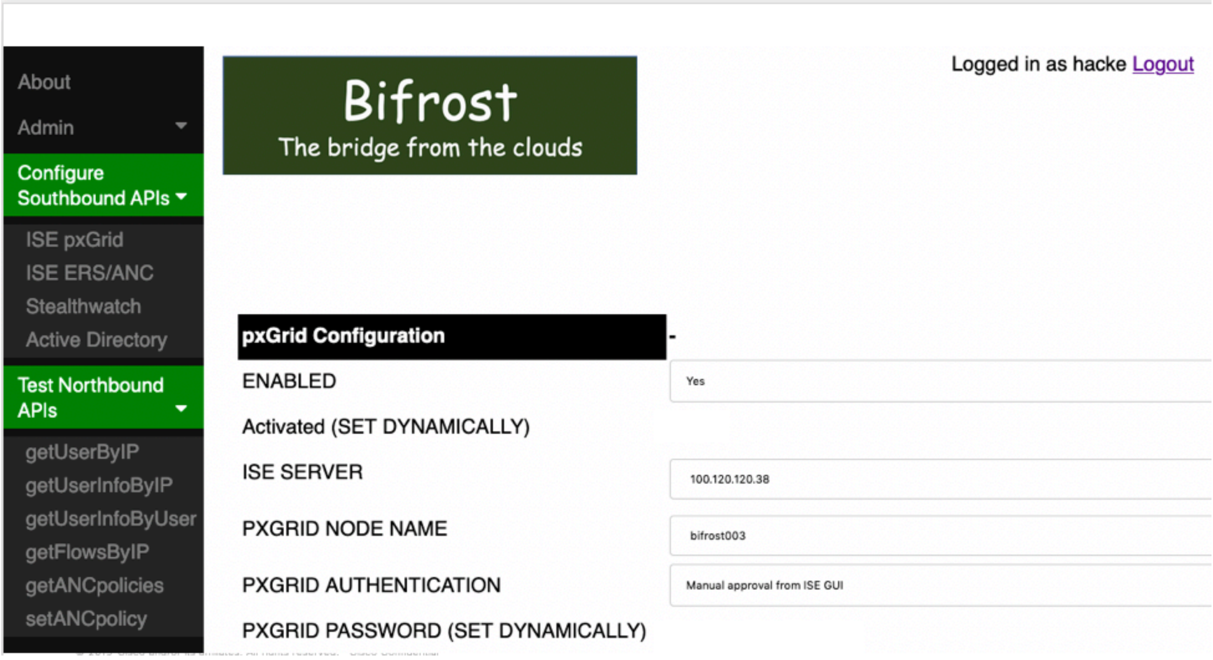
6. Configure and Test the ISE pxGrid Interface

The middleware currently only supports authentication with (dynamically generated) pre-shared key to the pxGrid bus. Manual Approval in the ISE GUI is therefore required.

Prepare by opening a separate tab in your browser and login to ISE, browsing to the page with pxGrid clients (ISE : Administration/pxGrid Services)

In the middleware, configure the ISE API by specifying

- Enabled Yes
- IP address of ISE server
- Nodename of pxGrid client, e.g. the middleware (must be unique)

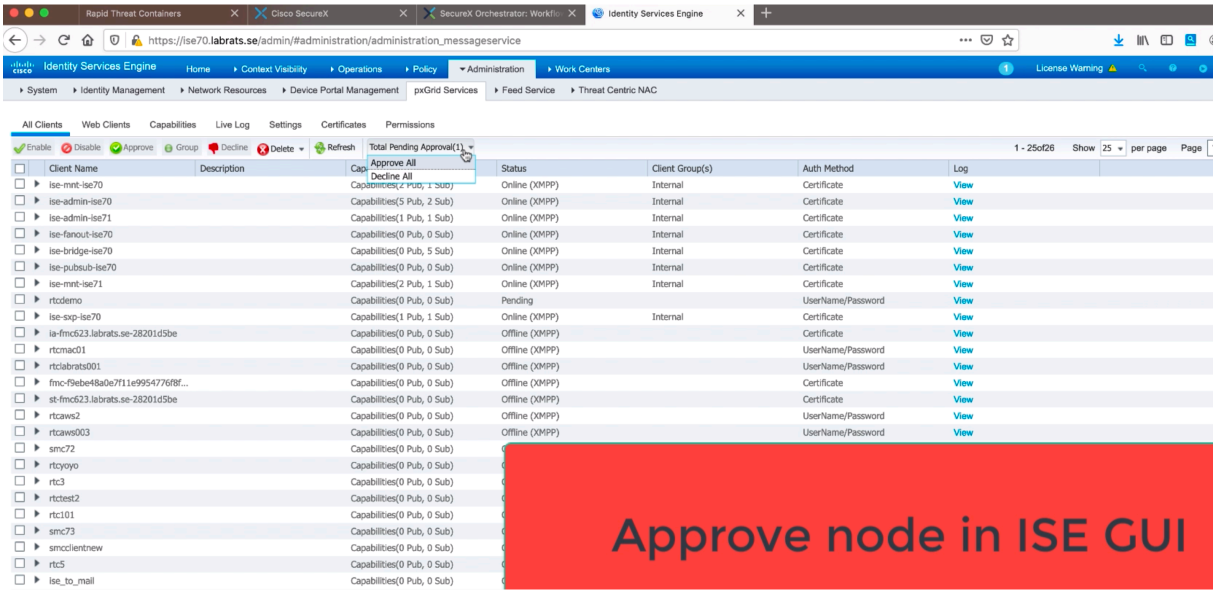


pxGrid Configuration

ENABLED	Yes
Activated (SET DYNAMICALLY)	
ISE SERVER	100.120.120.38
PXGRID NODE NAME	bifrost003
PXGRID AUTHENTICATION	Manual approval from ISE GUI
PXGRID PASSWORD (SET DYNAMICALLY)	

Figure 12 Configure ISE

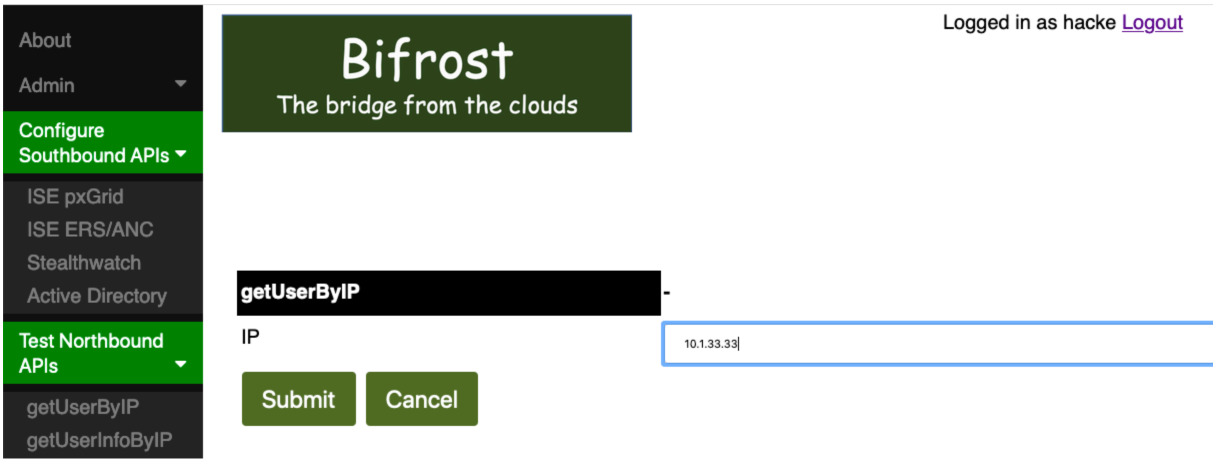
After submitting the ISE configuration, you must approve the client in the ISE GUI. This has to be done within 4 minutes from configuring ISE in the middleware.



□	Client Name	Description	Capabilities	Status	Client Group(s)	Auth Method	Log
□	ise-mnt-ise70		Capabilities(2 Pub, 1 Sub)	Online (XMPP)	Internal	Certificate	View
□	ise-admin-ise70		Capabilities(5 Pub, 2 Sub)	Online (XMPP)	Internal	Certificate	View
□	ise-admin-ise71		Capabilities(1 Pub, 1 Sub)	Online (XMPP)	Internal	Certificate	View
□	ise-fanout-ise70		Capabilities(0 Pub, 0 Sub)	Online (XMPP)	Internal	Certificate	View
□	ise-bridge-ise70		Capabilities(0 Pub, 5 Sub)	Online (XMPP)	Internal	Certificate	View
□	ise-pubsub-ise70		Capabilities(0 Pub, 0 Sub)	Online (XMPP)	Internal	Certificate	View
□	ise-mnt-ise71		Capabilities(2 Pub, 1 Sub)	Online (XMPP)	Internal	Certificate	View
□	rtcdemo		Capabilities(0 Pub, 0 Sub)	Pending		UserName/Password	View
□	ise-sxp-ise70		Capabilities(1 Pub, 1 Sub)	Online (XMPP)	Internal	Certificate	View
□	ia-fmc623.labrats.se-28201d5be		Capabilities(0 Pub, 0 Sub)	Offline (XMPP)		Certificate	View
□	rtcmac01		Capabilities(0 Pub, 0 Sub)	Offline (XMPP)		UserName/Password	View
□	rtclabrats001		Capabilities(0 Pub, 0 Sub)	Offline (XMPP)		UserName/Password	View
□	fmc-f9ebe48a0e7f11e9954776f8f...		Capabilities(0 Pub, 0 Sub)	Offline (XMPP)		Certificate	View
□	st-fmc623.labrats.se-28201d5be		Capabilities(0 Pub, 0 Sub)	Offline (XMPP)		Certificate	View
□	rtcaws2		Capabilities(0 Pub, 0 Sub)	Offline (XMPP)		UserName/Password	View
□	rtcaws003		Capabilities(0 Pub, 0 Sub)	Offline (XMPP)		UserName/Password	View
□	smc72		Capabilities(0 Pub, 0 Sub)				
□	rtcyoyo		Capabilities(0 Pub, 0 Sub)				
□	rtc3		Capabilities(0 Pub, 0 Sub)				
□	rtctest2		Capabilities(0 Pub, 0 Sub)				
□	rtc101		Capabilities(0 Pub, 0 Sub)				
□	smc73		Capabilities(0 Pub, 0 Sub)				
□	smcclientnew		Capabilities(0 Pub, 0 Sub)				
□	rtc5		Capabilities(0 Pub, 0 Sub)				
□	ise_to_mail		Capabilities(0 Pub, 0 Sub)				

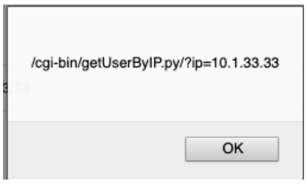
Figure 13 Approve pxGrid node in ISE GUI

You can test the ISE pxGrid connection by searching ISE API for an IP address, which should return the session data for that IP. Test Northbound APIs/getUserByIP.

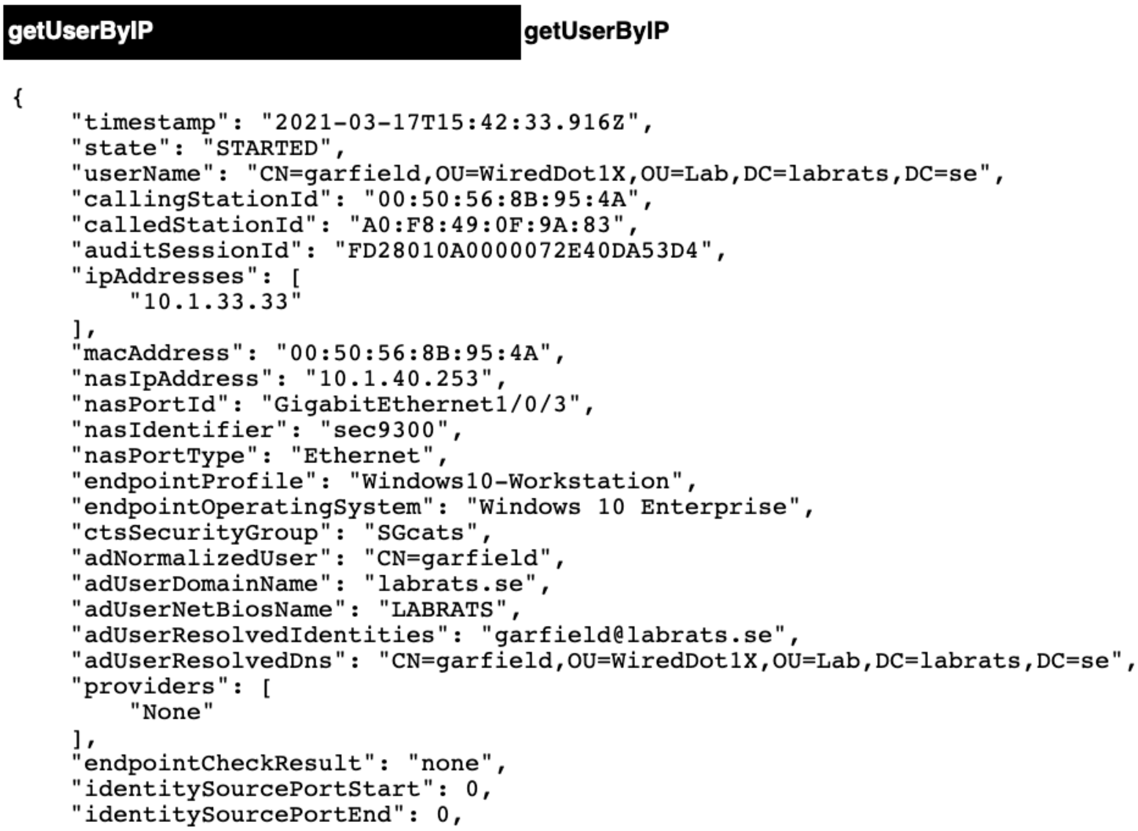


IP 10.1.33.33

Figure 14 Testing ISE pxGrid API



/cgi-bin/getUserByIP.py/?ip=10.1.33.33



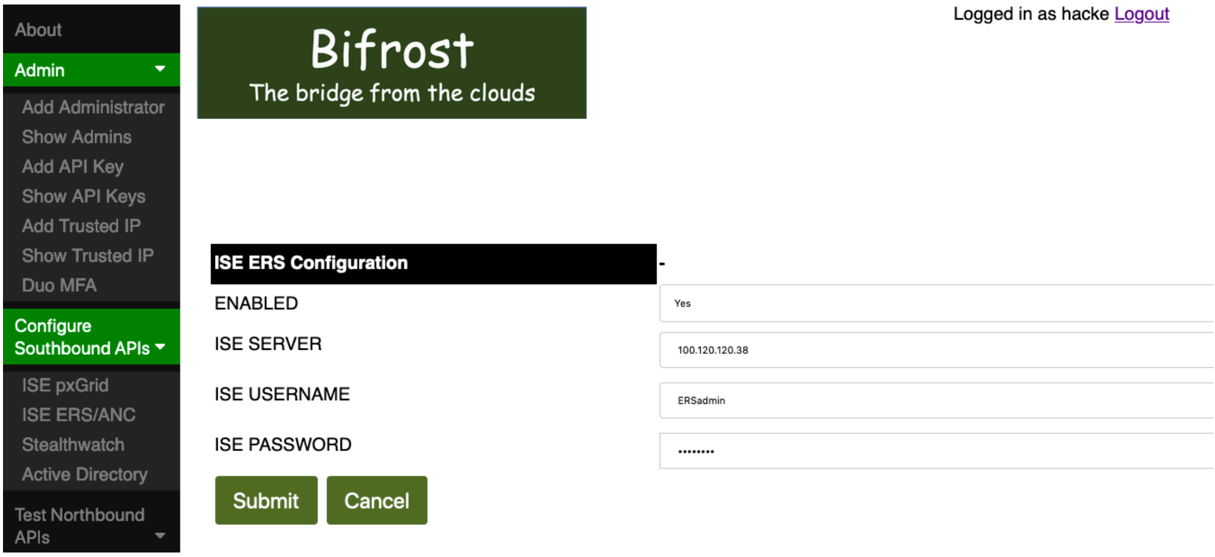
getUserByIP getUserByIP

```
{
    "timestamp": "2021-03-17T15:42:33.916Z",
    "state": "STARTED",
    "userName": "CN=garfield,OU=WiredDot1X,OU=Lab,DC=labrats,DC=se",
    "callingStationId": "00:50:56:8B:95:4A",
    "calledStationId": "A0:F8:49:0F:9A:83",
    "auditSessionId": "FD28010A0000072E40DA53D4",
    "ipAddresses": [
        "10.1.33.33"
    ],
    "macAddress": "00:50:56:8B:95:4A",
    "nasIpAddress": "10.1.40.253",
    "nasPortId": "GigabitEthernet1/0/3",
    "nasIdentifier": "sec9300",
    "nasPortType": "Ethernet",
    "endpointProfile": "Windows10-Workstation",
    "endpointOperatingSystem": "Windows 10 Enterprise",
    "ctsSecurityGroup": "SGcats",
    "adNormalizedUser": "CN=garfield",
    "adUserDomainName": "labrats.se",
    "adUserNetBiosName": "LABRATS",
    "adUserResolvedIdentities": "garfield@labrats.se",
    "adUserResolvedDns": "CN=garfield,OU=WiredDot1X,OU=Lab,DC=labrats,DC=se",
    "providers": [
        "None"
    ],
    "endpointCheckResult": "none",
    "identitySourcePortStart": 0,
    "identitySourcePortEnd": 0,
```

Figure 15 Testing ISE pxGrid API, output

7. Configure ISE ERS API

The ISE ERS API is used by the middleware to set and clear ANC policies (which can be done without a pxGrid configuration).



ISE ERS Configuration

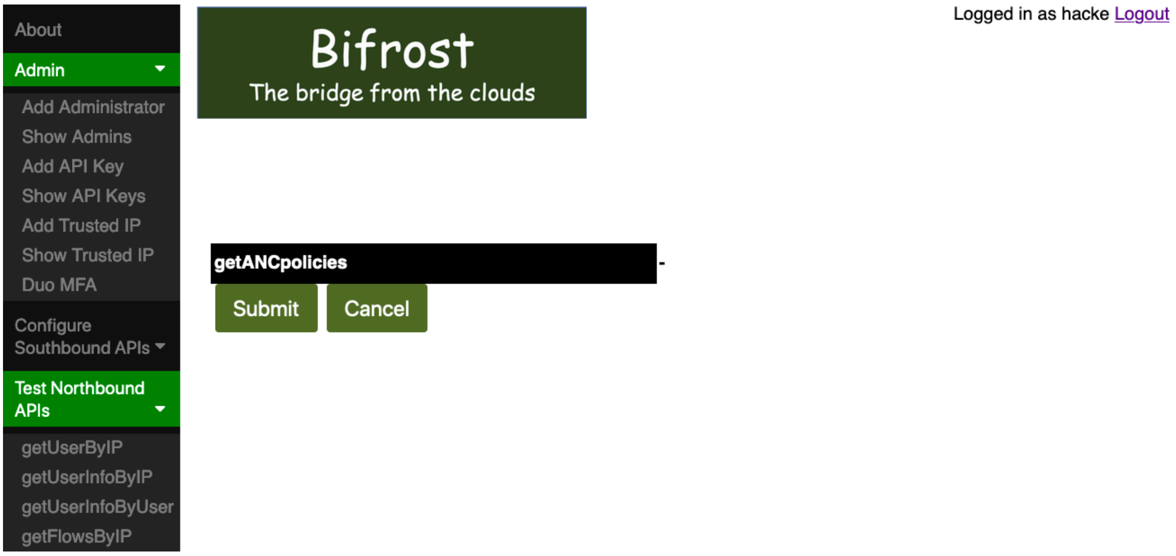
ENABLED Yes

ISE SERVER 100.120.120.38

ISE USERNAME ERSadmin

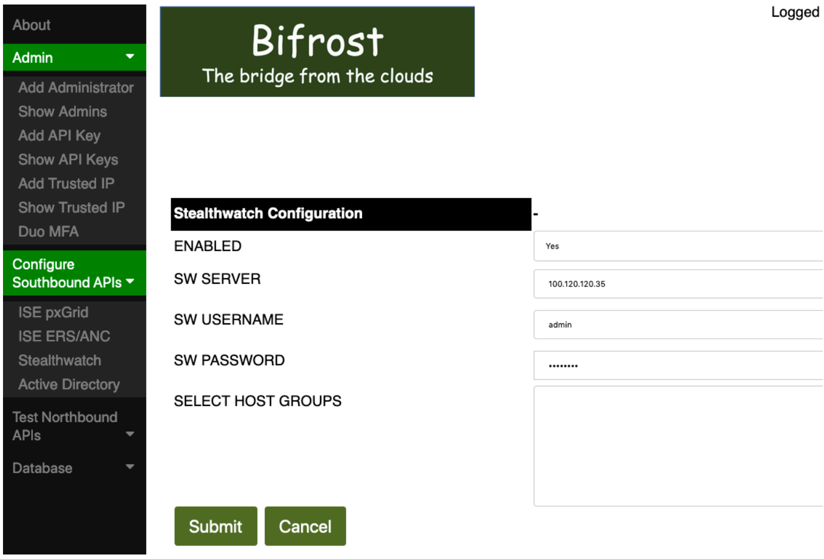
ISE PASSWORD ••••••••

The API can be tested with Test NorthBound APIs/getANCpolicies



8. Configure Stealthwatch API

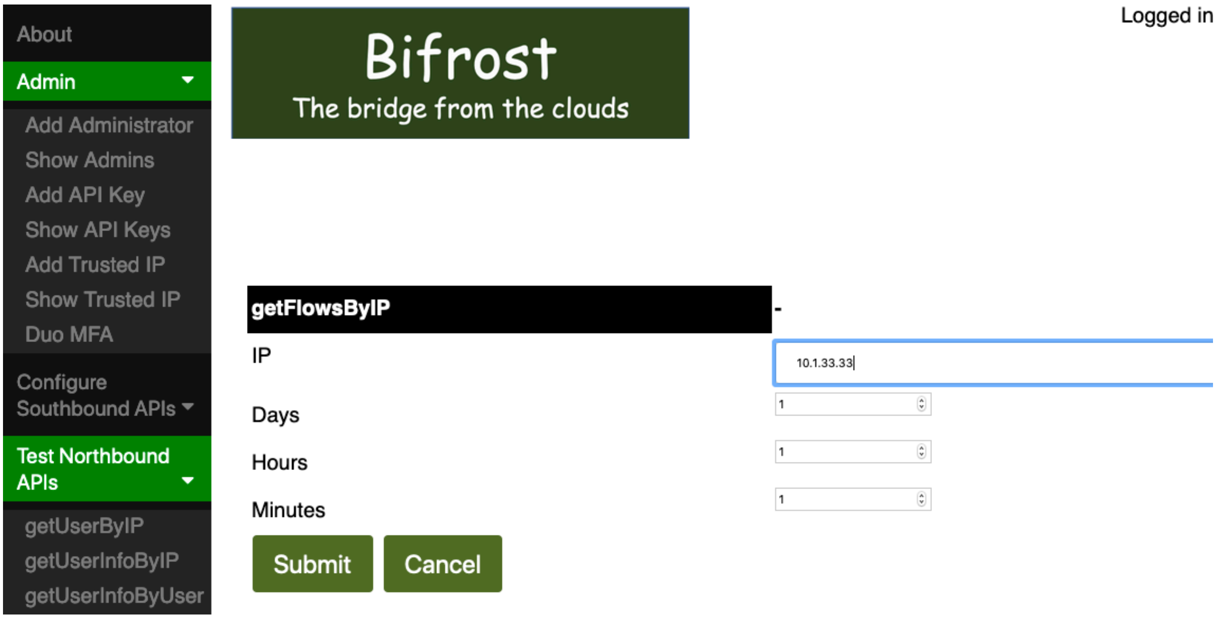
The Stealthwatch API is used to retrieve flows for a specific IP address.



Stealthwatch Configuration

ENABLED	Yes
SW SERVER	100.120.120.35
SW USERNAME	admin
SW PASSWORD	....
SELECT HOST GROUPS	

The configuration can be tested with Test Northbound APIs/getFlowsByIP.

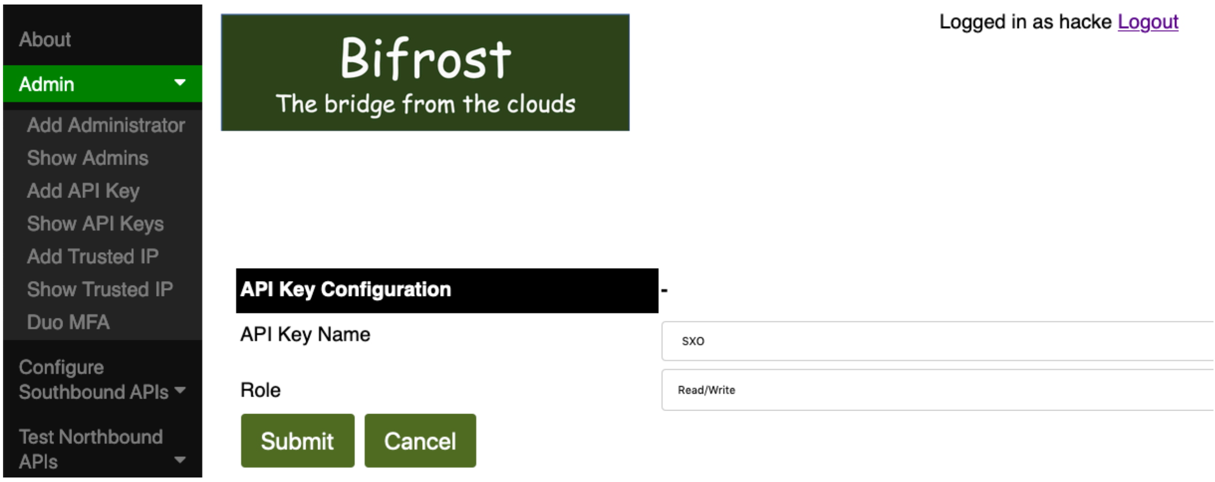


getFlowsByIP

IP	10.1.33.33
Days	1
Hours	1
Minutes	1

9. Generate API key

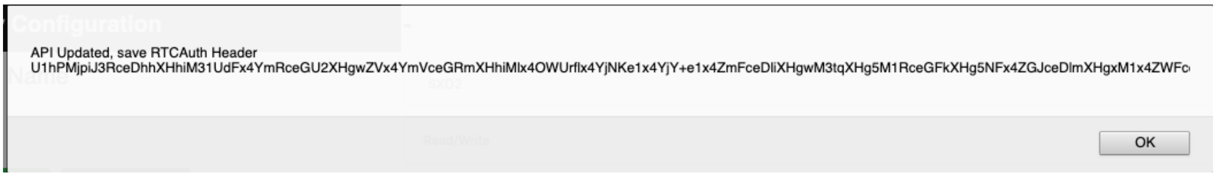
The API calls are authenticated with an API key. Generate the API key and copy it to notepad or similar.



API Key Configuration

API Key Name: SXO

Role: Read/Write



API Updated, save RTCAuth Header
U1hPMjpiJ3RceDhhXHhiM31UdFx4YmRceGU2XHgwZVx4YmVceGRmXHhiMlx4OWUrflx4YjNKe1x4YjY+e1x4ZmFceDliXHgwM3tqXHg5M1RceGFkXHg5NFx4ZGJceDlmXHgxM1x4ZWFc

Figure 15 Add API key for SecureX Orchestrator.

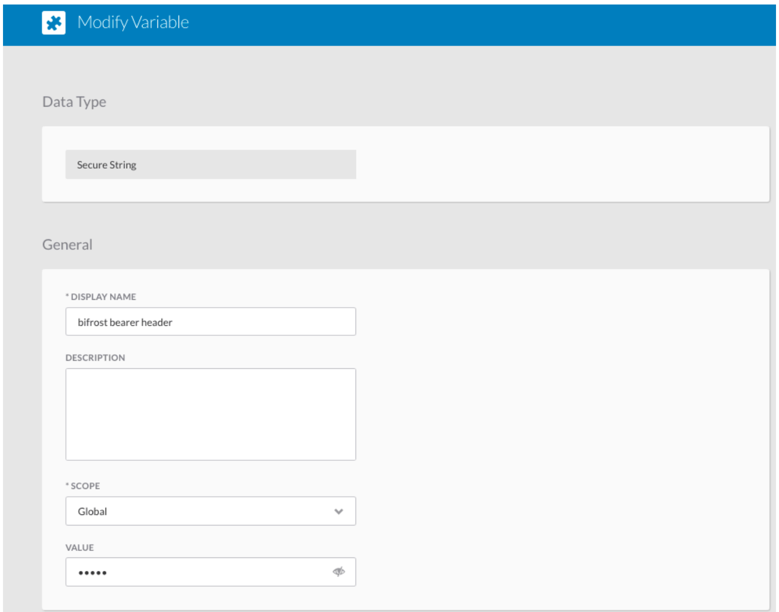
The API key has to be present in all calls in the custom header: Bearer.

Bearer <key>

SecureX Orchestration Configuration

10. SXO – Create Secure String Variable with API Key

In SXO, create a Secure String Variable with the API key from previous step.



Modify Variable

Data Type

Secure String

General

* DISPLAY NAME

bifrost bearer header

DESCRIPTION

* SCOPE

Global

VALUE

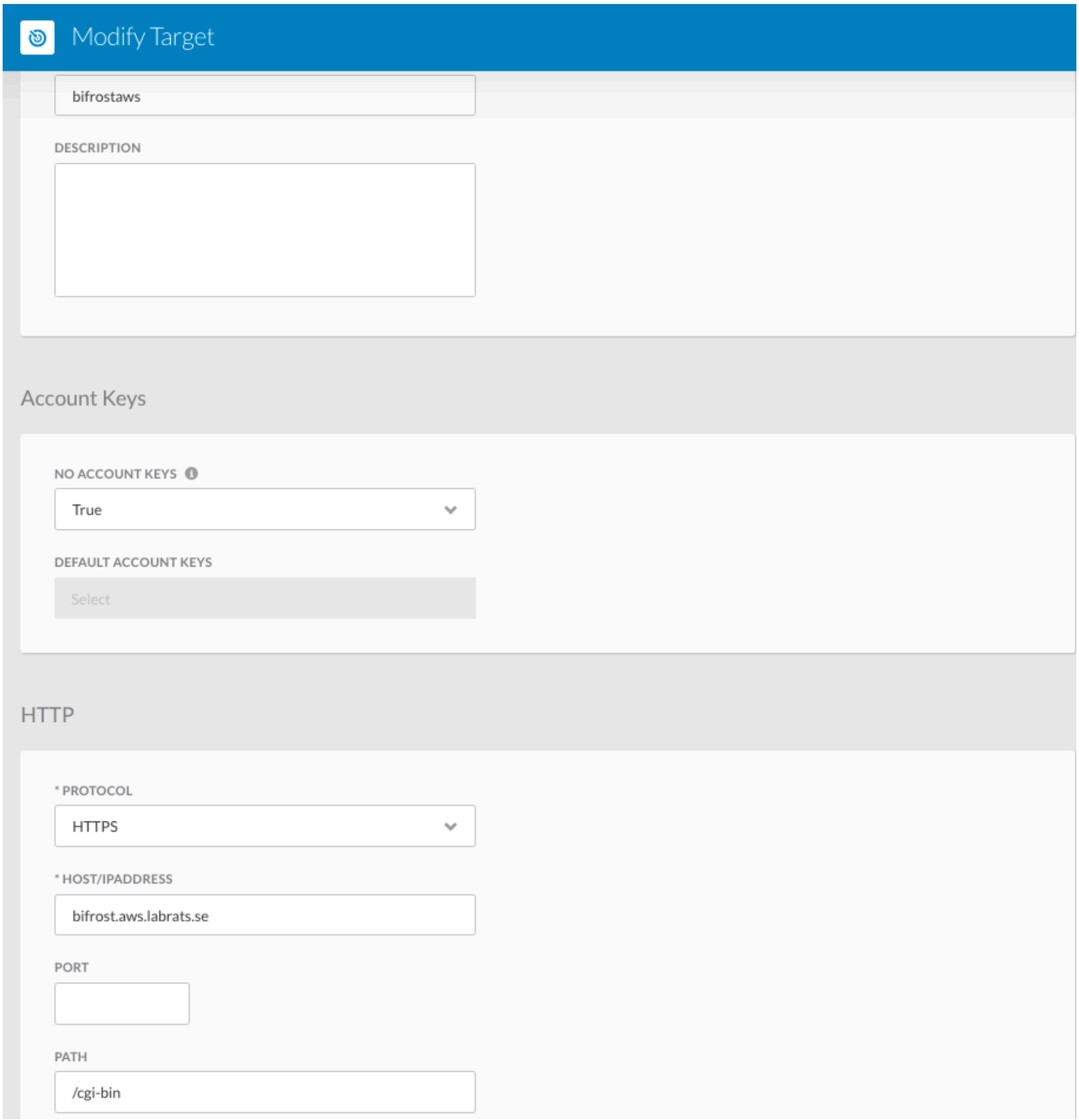
•••••

Figure 16 Create Secure String Variable with API key

11. SXO – Create Target

In SecureX Orchestration, create a Target that points to your Bifrost Middleware.

- Protocol HTTPS
- Specify hostname/IP of your Middleware
- Path /cgi-bin
- Check Disable Server Certificate Validation



Modify Target

bifrostaws

DESCRIPTION

Account Keys

NO ACCOUNT KEYS

True

DEFAULT ACCOUNT KEYS

Select

HTTP

* PROTOCOL

HTTPS

* HOST/IPADDRESS

bifrost.aws.labrats.se

PORT

PATH

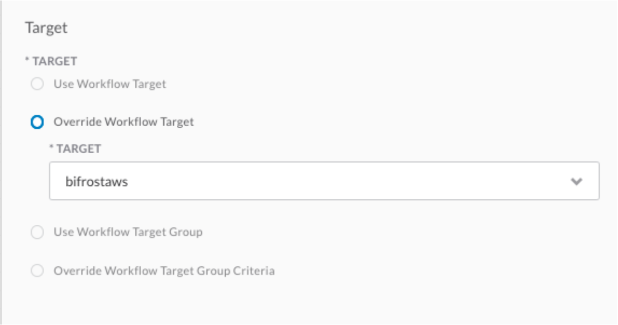
/cgi-bin

Figure 17 SXO - specify target

12. SXO - Creating web requests to the Bifrost Middleware

In SXO when you create a workflow with a web request to the Bifrost middleware, ensure that

Target is set to the target defined in previous step



Target

* TARGET

○ Use Workflow Target

● Override Workflow Target

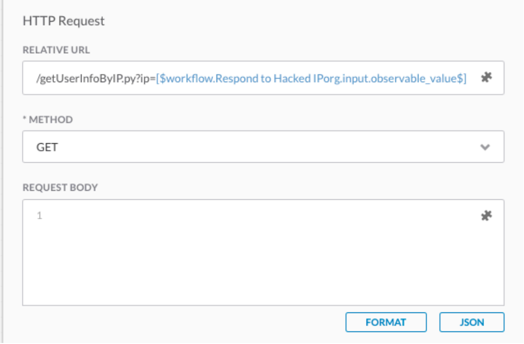
* TARGET

bifrostaws

○ Use Workflow Target Group

○ Override Workflow Target Group Criteria

You specify the url (relative to cgi-bin)



HTTP Request

RELATIVE URL

/getUserInfoByIP.py?ip=[$workflow.Respond to Hacked IPorg.input.observable_value$]

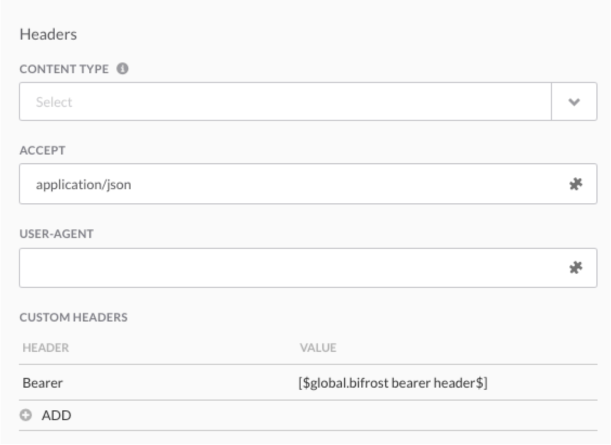
* METHOD

GET

REQUEST BODY

```
1
```

You add a custom header named Bearer with the value of the Secure String from previous step where we set the global variable (which is the API key we created in Bifrost middleware).



Headers

CONTENT TYPE

Select

ACCEPT

application/json

USER-AGENT

CUSTOM HEADERS

HEADER	VALUE
Bearer	[$global.bifrost bearer header$]

API Calls

Base url for all API calls https://hostname/cgi-bin/

Authorization is with custom Bearer header followed by the generated API keys.

API call	Relative URL	Method	Body
Get User By IP	/getUserByIP.py/?ip=…	GET	
Get User Info from IP	/getUserInfoByIP/?ip=…	GET	
Get User Info from username	/getUserInfoByUser/?user=…	GET	
Get flows by IP	getFlowsByIP.py/?ip=ip&days=1&hours=1&minutes=1	GET	
Get ANC policies	getANCpolicies.py	GET	
Apply ANC	setANCpolicy.py	POST	{"action":"Apply Policy","policy":"Qu ","ip":"10.1.33.33"}
Clear ANC	setANCpolicy.py	POST	{"action":"Clear Policy","policy":"Qu ","ip":"10.1.33.33"}