

CISCO *Live!*



#CiscoLive



The bridge to possible

Working with the Intersight API in Go

Chris Gascoigne, Principal Architect
@chrsgascoigne
DEVNET-3004

Cisco Webex App

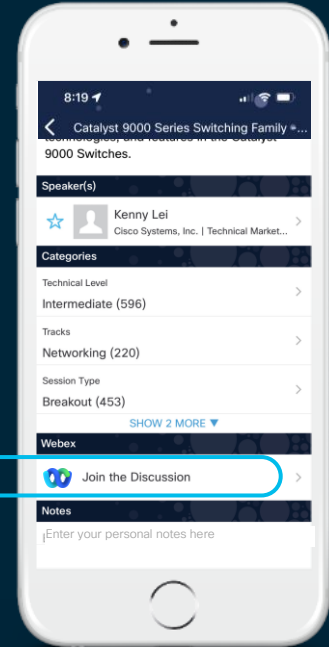
Questions?

Use Cisco Webex App to chat with the speaker after the session

How

- 1 Find this session in the Cisco Live Mobile App
- 2 Click “Join the Discussion”
- 3 Install the Webex App or go directly to the Webex space
- 4 Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until June 17, 2022.



<https://ciscolive.ciscoevents.com/ciscolivebot/#DEVNET-3004>



Agenda

- Why use Go with Intersight?
- A Brief Introduction to Intersight API
- Getting Started with the Intersight Go SDK
- Use Cases and Examples
- Conclusion and Key Takeaways

Why use Go with Intersight?



Language fight!





Choose the right tool for the job

Why Use Go for Automation?



Produce statically linked binaries for any platform



Easy to distribute without dependencies



Great for building tools for CI and automation



Easily distribute tools to other users

Why Use Go for Automation?



Static typing and explicit error handling

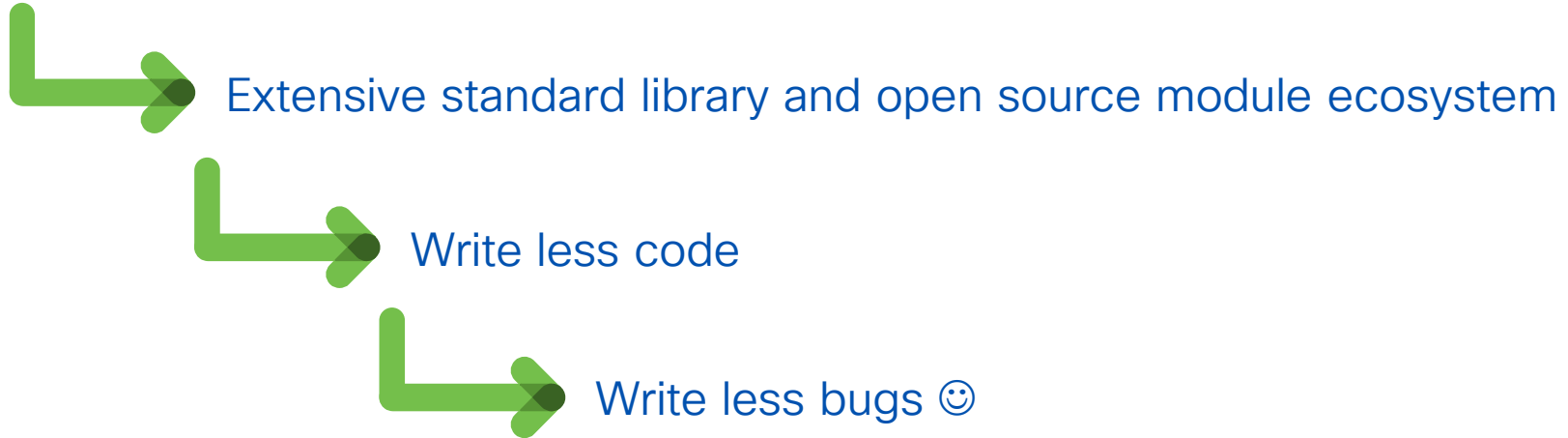


Can make scripts and utilities more robust



Less likely to encounter unexpected corner-cases

Why Use Go for Automation?



Why Use Go for Automation?



Performant with low memory utilisation

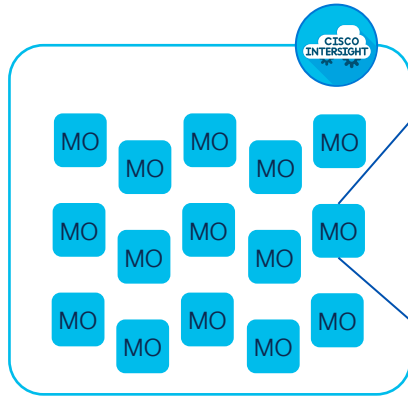


Complete complex tasks more quickly

A Brief Introduction to Intersight API



Intersight API – Managed Objects



```
ClassId: ntp.Policy
Moid: 61e9f9696275722d31e74cfd
ObjectType: ntp.Policy
Organization:
  ClassId: mo.MoRef
  Moid: 5deed7fc6972652d33bc48d0
  ObjectType: organization.Organization
  link: https://...2652d33bc48d0

Name: NTPTest
Description: Example NTP Policy
Enabled: true
NtpServers:
- ntp.esl.cisco.com
Timezone: Australia/Sydney

...
```

(Simplified Example)

Intersight API – Endpoints

The screenshot displays the Cisco Intersight Developer Center API Reference page. The top navigation bar includes 'Intersight Developer Center', 'Guides', 'API Reference', 'Downloads', and 'Support'. The user 'Chris Gascoigne' is logged in. The main content area is titled 'API Reference v1.0.9-6484'. On the left, a search bar is present, and a list of endpoints is shown. The 'ntp/Policies' endpoint is selected, and its details are displayed in the center. The 'GET' method for 'Read a 'ntp.Policy' resource.' is highlighted. The 'Parameters' tab is active, showing the '\$filter { string }' and '\$orderby { string }' query parameters. The 'REST Client' sidebar on the right shows the endpoint '/api/v1/ntp/Policies' and a 'Send' button.

API Reference v1.0.9-6484

GET Read a 'ntp.Policy' resource. REST Client

Parameters Response Model

\$filter { string } query

Filter criteria for the resources to return. A URI with a \$filter query option identifies a subset of the entries from the Collection of Entries. The subset is determined by selecting only the Entries that satisfy the predicate expression specified by the \$filter option. The expression language that is used in \$filter queries supports references to properties and literals. The literal values can be strings enclosed in single quotes, numbers and boolean values (true or false).

\$orderby { string } query

Determines what properties are used to sort the collection of resources.

\$top { integer } query

Specifies the maximum number of resources to return in the response.

REST Client

GET /api/v1/ntp/Policies

+ Query Parameter

Send

Response Text Response Info

1

Intersight API – Authentication

The screenshot shows the Cisco Intersight console interface. On the left is a navigation sidebar with sections: MONITOR, OPERATE, CONFIGURE, and ADMIN. The main content area is titled 'Settings' and contains three tabs: GENERAL, AUTHENTICATION, and ACCESS & PERMISSIONS. The 'API KEYS' sub-tab is selected under the AUTHENTICATION section. The main panel displays 'API Keys' with a search bar, a table header (Description, API Key ID, Purpose, Create..., Email, Role, Identity Provider), and a message 'NO ITEMS AVAILABLE'. A red box highlights the 'Generate API Key' button in the top right corner of the main panel.

Intersight API – Authentication

The screenshot displays the Cisco Intersight user interface. On the left, a navigation sidebar includes sections for MONITOR, OPERATE, CONFIGURE, and ADMIN. The main content area is titled 'API Keys' and features a 'Generate API Key' button in the top right. A modal window titled 'Generate API Key' is open in the center. It contains a 'Description' field with the text 'Cisco Live US Demo'. Below this, the 'API Key Purpose' section has two radio button options: 'API key for OpenAPI schema version 2' (which is selected) and 'API key for OpenAPI schema version 3 (This is a feature in preview and for SDK developer use only)'. At the bottom of the modal, there are two buttons: 'Close' and 'Generate'. The 'Generate' button is highlighted with a red rectangular box.

Intersight API – Authentication

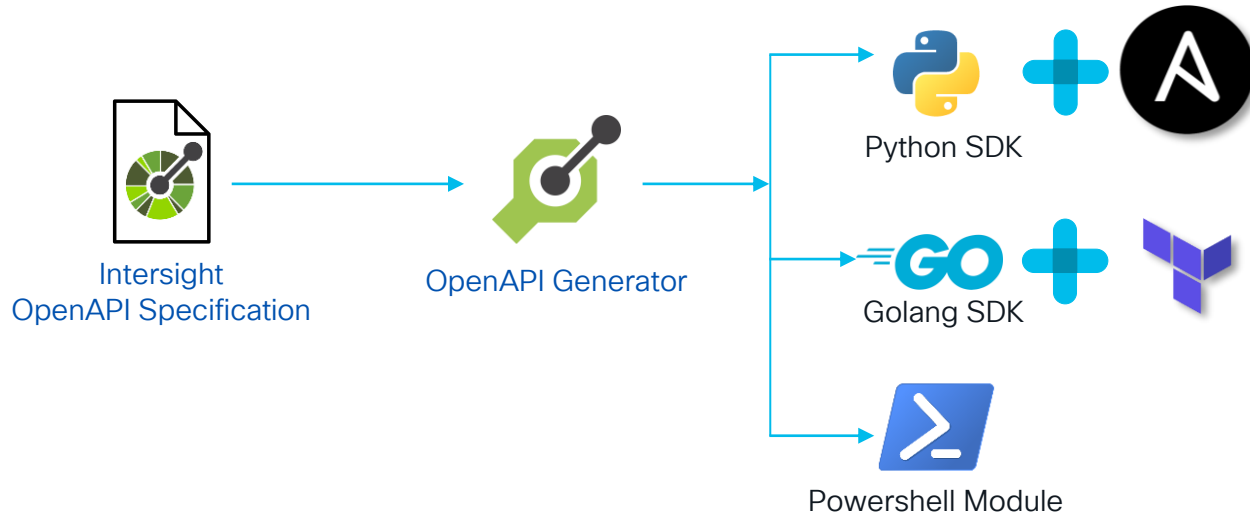
The screenshot shows the Cisco Intersight interface with the 'Generate API Key' dialog box open. The dialog contains the following information:

- API Key ID:** 61970b917564612d333d4d41/61970b917564612d333d4d46/626f14f57564612d333c3b49
- Secret Key:** ---BEGIN RSA PRIVATE KEY---
MIIEowIBAAKCAQEAwrF2ko3D8fC4ENPw2lZy1gzanaNEPfp
qtWZChBlZJqFxxJJK
RgX7S/IM0BaqIEytTeC0soc1TZ9wfxZlZuxyH9a6+h6g7wc
QPR7AUIGY0hnT130t
JkNDCZvKRTc5uayNikU2SbxzcpvFaPzx2dfhoEuTAIZd4nCS
wF6l69Kp1U0zPzxn

Two red arrows point from text boxes to the API Key ID and Secret Key fields:

- Arrow 1 points to the API Key ID field. The text box says: "Will be used to tell Intersight API which API key you are using".
- Arrow 2 points to the Secret Key field. The text box says: "Secret (Private) Key used to sign API requests".

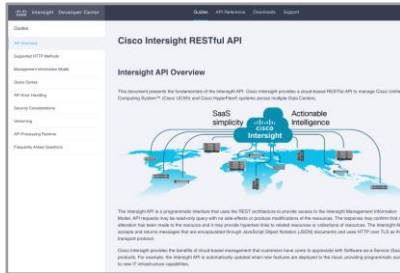
Intersight API – OpenAPI Specification



<https://intersight.com/apidocs/downloads/>

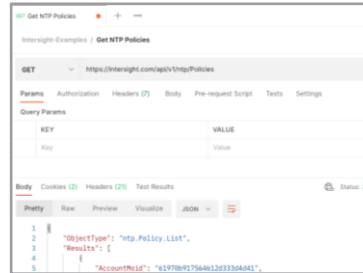
Getting Started With the Intersight API

API Documentation Browser



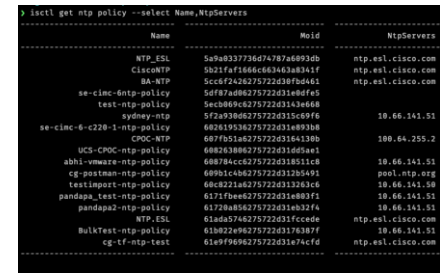
<https://www.intersight.com/apidocs/>

Postman



<https://github.com/CiscoDevNet/intersight-postman>

Command Line (isctl)



<https://github.com/cgascoig/isctl>

DEVNET-1418 – Take the Easy Road to the Intersight API

Getting Started with the Intersight Go SDK





Let's dive into some code!

Let's setup a new project

```
> # Create a new Go module for our code
> mkdir clus-intersight-go-example
> cd clus-intersight-go-example
> go mod init github.com/cgascoig/clus-intersight-go-example
go: creating new go.mod: module github.com/cgascoig/clus-intersight-go-example
go: to add module requirements and sums:
go mod tidy
```

Let's setup a new project

```
> # Create a new Go module for our code
> mkdir clus-intersight-go-example
> cd clus-intersight-go-example
> go mod init github.com/cgascoig/clus-intersight-go-example
go: creating new go.mod: module github.com/cgascoig/clus-intersight-go-example
go: to add module requirements and sums:
go mod tidy

> # Add the CiscoDevNet/intersight-go module to our requirements
> go get github.com/CiscoDevNet/intersight-go@v1.0.9.5808
go: downloading github.com/CiscoDevNet/intersight-go v0.0.0-20220419063803-78b978058f62
go: downloading golang.org/x/oauth2 v0.0.0-20210323180902-22b0adad7558
go: added github.com/CiscoDevNet/intersight-go v0.0.0-20220419063803-78b978058f62
go: added github.com/golang/protobuf v1.4.2
go: added golang.org/x/net v0.0.0-20200822124328-c89045814202
go: added golang.org/x/oauth2 v0.0.0-20210323180902-22b0adad7558
go: added google.golang.org/appengine v1.6.6
go: added google.golang.org/protobuf v1.25.0
```

A simple Intersight Go program (1)

```
package main

import (
    "context"
    "fmt"
    "log"
    "os"

    intersight "github.com/CiscoDevNet/intersight-go"
)
```

main.go (partial code listing)

A simple Intersight Go program (2)

```
func setupIntersightClient(keyID, keyFile string) (*intersight.APIClient, context.Context, error) {  
    config := intersight.NewConfiguration()  
    client := intersight.NewAPIClient(config)  
  
    // Set up the authentication configuration struct  
    authConfig := intersight.HttpSignatureAuth{  
        KeyId: keyID,  
        PrivateKeyPath: keyFile,  
  
        SigningScheme: intersight.HttpSigningSchemeRsaSha256,  
        SignedHeaders: []string{  
            intersight.HttpSignatureParameterRequestTarget, // The special (request-target) parameter  
            "Host", // The Host request header specifies the domain name of the server  
            "Date", // The date and time at which the message was originated.  
            "Digest", // A cryptographic digest of the request body.  
        },  
        SigningAlgorithm: intersight.HttpSigningAlgorithmRsaPKCS1v15,  
    }  
  
    // Get a context that includes the authentication config  
    authCtx, err := authConfig.ContextWithValue(context.Background())  
    if err != nil {  
        return nil, nil, fmt.Errorf("error creating authentication context: %v", err)  
    }  
  
    return client, authCtx, nil  
}
```

main.go (partial code listing)

A simple Intersight Go program (2)

```
func setupIntersightClient(keyID, keyFile string) (*intersight.APIClient, context.Context, error) {
    config := intersight.NewConfiguration()

    client := intersight.NewAPIClient(config)

    // Set up the authentication configuration struct
    authConfig := intersight.HttpSignatureAuth{
        KeyId: keyID,
        PrivateKeyPath: keyFile,

        SigningScheme: intersight.HttpSigningSchemeRsaSha256,
        SignedHeaders: []string{
            intersight.HttpSignatureParameterRequestTarget, // The special (request-target) parameter
            "Host", // The Host request header specifies the domain name of the server
            "Date", // The date and time at which the message was originated.
            "Digest", // A cryptographic digest of the request body.
        },
        SigningAlgorithm: intersight.HttpSigningAlgorithmRsaPKCS1v15,
    }

    // Get a context that includes the authentication config
    authCtx, err := authConfig.ContextWithValue(context.Background())
    if err != nil {
        return nil, nil, fmt.Errorf("error creating authentication context: %v", err)
    }

    return client, authCtx, nil
}
```

main.go (partial code listing)

A simple Intersight Go program (2)

```
func setupIntersightClient(keyID, keyFile string) (*intersight.APIClient, context.Context, error) {
    config := intersight.NewConfiguration()

    client := intersight.NewAPIClient(config)

    // Set up the authentication configuration struct
    authConfig := intersight.HttpSignatureAuth{
        KeyId: keyID,
        PrivateKeyPath: keyFile,

        SigningScheme: intersight.HttpSigningSchemeRsaSha256,
        SignedHeaders: []string{
            intersight.HttpSignatureParameterRequestTarget, // The special (request-target) parameter
            "Host", // The Host request header specifies the domain name of the server
            "Date", // The date and time at which the message was originated.
            "Digest", // A cryptographic digest of the request body.
        },
        SigningAlgorithm: intersight.HttpSigningAlgorithmRsaPKCS1v15,
    }

    // Get a context that includes the authentication config
    authCtx, err := authConfig.ContextWithValue(context.Background())
    if err != nil {
        return nil, nil, fmt.Errorf("error creating authentication context: %v", err)
    }

    return client, authCtx, nil
}
```

main.go (partial code listing)

A simple Intersight Go program (2)

```
func setupIntersightClient(keyID, keyFile string) (*intersight.APIClient, context.Context, error) {
    config := intersight.NewConfiguration()

    client := intersight.NewAPIClient(config)

    // Set up the authentication configuration struct
    authConfig := intersight.HttpSignatureAuth{
        KeyId: keyID,
        PrivateKeyPath: keyFile,

        SigningScheme: intersight.HttpSigningSchemeRsaSha256,
        SignedHeaders: []string{
            intersight.HttpSignatureParameterRequestTarget, // The special (request-target) parameter
            "Host", // The Host request header specifies the domain name of the server
            "Date", // The date and time at which the message was originated.
            "Digest", // A cryptographic digest of the request body.
        },
        SigningAlgorithm: intersight.HttpSigningAlgorithmRsaPKCS1v15,
    }

    // Get a context that includes the authentication config
    authCtx, err := authConfig.ContextWithValue(context.Background())
    if err != nil {
        return nil, nil, fmt.Errorf("error creating authentication context: %v", err)
    }

    return client, authCtx, nil
}
```

main.go (partial code listing)

A simple Intersight Go program (3)

```
func main() {  
    keyID := os.Getenv("IS_KEY_ID")  
    keyFile := os.Getenv("IS_KEY_FILE")  
  
    intersightClient, intersightAuthCtx, err := setupIntersightClient(keyID, keyFile)  
    if err != nil {  
        log.Fatalf("Error setting up Intersight client: %v", err)  
    }  
  
    getNtpPolicyListRequest := intersightClient.NtpApi.GetNtpPolicyList(intersightAuthCtx)  
  
    ntpResult, httpResult, err := getNtpPolicyListRequest.Execute()  
    if err != nil {  
        log.Fatalf("Error making Intersight API call: %v", err)  
    }  
  
    log.Printf("HTTP Status: %v %v", httpResult.StatusCode, httpResult.Status)  
  
    for _, ntpPolicy := range ntpResult.NtpPolicyList.Results {  
        fmt.Printf("NTP Policy: Name=%v Enabled=%v NtpServers=%v\n", *ntpPolicy.Name, *ntpPolicy.Enabled,  
            ntpPolicy.NtpServers)  
    }  
}
```

main.go (partial code listing)

A simple Intersight Go program (3)

```
func main() {  
    keyID := os.Getenv("IS_KEY_ID")  
    keyFile := os.Getenv("IS_KEY_FILE")  
  
    intersightClient, intersightAuthCtx, err := setupIntersightClient(keyID, keyFile)  
    if err != nil {  
        log.Fatalf("Error setting up Intersight client: %v", err)  
    }  
  
    getNtpPolicyListRequest := intersightClient.NtpApi.GetNtpPolicyList(intersightAuthCtx)  
  
    ntpResult, httpResult, err := getNtpPolicyListRequest.Execute()  
    if err != nil {  
        log.Fatalf("Error making Intersight API call: %v", err)  
    }  
  
    log.Printf("HTTP Status: %v %v", httpResult.StatusCode, httpResult.Status)  
  
    for _, ntpPolicy := range ntpResult.NtpPolicyList.Results {  
        fmt.Printf("NTP Policy: Name=%v Enabled=%v NtpServers=%v\n", *ntpPolicy.Name, *ntpPolicy.Enabled,  
            ntpPolicy.NtpServers)  
    }  
}
```

main.go (partial code listing)

A simple Intersight Go program (3)

```
func main() {
    keyID := os.Getenv("IS_KEY_ID")
    keyFile := os.Getenv("IS_KEY_FILE")

    intersightClient, intersightAuthCtx, err := setupIntersightClient(keyID, keyFile)
    if err != nil {
        log.Fatalf("Error setting up Intersight client: %v", err)
    }

    getNtpPolicyListRequest := intersightClient.NtpApi.GetNtpPolicyList(intersightAuthCtx)

    ntpResult, httpResult, err := getNtpPolicyListRequest.Execute()
    if err != nil {
        log.Fatalf("Error making Intersight API call: %v", err)
    }

    log.Printf("HTTP Status: %v %v", httpResult.StatusCode, httpResult.Status)

    for _, ntpPolicy := range ntpResult.NtpPolicyList.Results {
        fmt.Printf("NTP Policy: Name=%v Enabled=%v NtpServers=%v\n", *ntpPolicy.Name, *ntpPolicy.Enabled,
            ntpPolicy.NtpServers)
    }
}
```

main.go (partial code listing)

A simple Intersight Go program (3)

```
func main() {
    keyID := os.Getenv("IS_KEY_ID")
    keyFile := os.Getenv("IS_KEY_FILE")

    intersightClient, intersightAuthCtx, err := setupIntersightClient(keyID, keyFile)
    if err != nil {
        log.Fatalf("Error setting up Intersight client: %v", err)
    }

    getNtpPolicyListRequest := intersightClient.NtpApi.GetNtpPolicyList(intersightAuthCtx)

    ntpResult, httpResult, err := getNtpPolicyListRequest.Execute()
    if err != nil {
        log.Fatalf("Error making Intersight API call: %v", err)
    }

    log.Printf("HTTP Status: %v %v", httpResult.StatusCode, httpResult.Status)

    for _, ntpPolicy := range ntpResult.NtpPolicyList.Results {
        fmt.Printf("NTP Policy: Name=%v Enabled=%v NtpServers=%v\n", *ntpPolicy.Name, *ntpPolicy.Enabled,
            ntpPolicy.NtpServers)
    }
}
```

main.go (partial code listing)

A simple Intersight Go program (3)

```
func main() {
    keyID := os.Getenv("IS_KEY_ID")
    keyFile := os.Getenv("IS_KEY_FILE")

    intersightClient, intersightAuthCtx, err := setupIntersightClient(keyID, keyFile)
    if err != nil {
        log.Fatalf("Error setting up Intersight client: %v", err)
    }

    getNtpPolicyListRequest := intersightClient.NtpApi.GetNtpPolicyList(intersightAuthCtx)

    ntpResult, httpResult, err := getNtpPolicyListRequest.Execute()
    if err != nil {
        log.Fatalf("Error making Intersight API call: %v", err)
    }
    log.Printf("HTTP Status: %v %v", httpResult.StatusCode, httpResult.Status)

    for _, ntpPolicy := range ntpResult.NtpPolicyList.Results {
        fmt.Printf("NTP Policy: Name=%v Enabled=%v NtpServers=%v\n", *ntpPolicy.Name, *ntpPolicy.Enabled,
            ntpPolicy.NtpServers)
    }
}
```

main.go (partial code listing)

A simple Intersight Go program (3)

```
func main() {
    keyID := os.Getenv("IS_KEY_ID")
    keyFile := os.Getenv("IS_KEY_FILE")

    intersightClient, intersightAuthCtx, err := setupIntersightClient(keyID, keyFile)
    if err != nil {
        log.Fatalf("Error setting up Intersight client: %v", err)
    }

    getNtpPolicyListRequest := intersightClient.NtpApi.GetNtpPolicyList(intersightAuthCtx)

    ntpResult, httpResult, err := getNtpPolicyListRequest.Execute()
    if err != nil {
        log.Fatalf("Error making Intersight API call: %v", err)
    }

    log.Printf("HTTP Status: %v %v", httpResult.StatusCode, httpResult.Status)

    for _, ntpPolicy := range ntpResult.NtpPolicyList.Results {
        fmt.Printf("NTP Policy: Name=%v Enabled=%v NtpServers=%v\n", *ntpPolicy.Name, *ntpPolicy.Enabled,
            ntpPolicy.NtpServers)
    }
}
```

main.go (partial code listing)

Build and run!

```
> go build
```

Build and run!

```
> go build  
  
> export IS_KEY_ID=$(cat ~/intersight-cgascoig-clus-demo.keyid.txt)  
> export IS_KEY_FILE=~/intersight-cgascoig-clus-demo.pem
```

Build and run!

```
> go build

> export IS_KEY_ID=$(cat ~/intersight-cgascoig-clus-demo.keyid.txt)
> export IS_KEY_FILE=~/intersight-cgascoig-clus-demo.pem

> ./clus-intersight-go-example
2022/05/03 12:24:01 HTTP Status: 200 200 OK
NTP Policy: Name=NTPTest Enabled=true NtpServers=[ntp.esl.cisco.com]
NTP Policy: Name=NtpTest1 Enabled=true NtpServers=[1.1.1.1 2.2.2.2]
NTP Policy: Name=NtpTest2 Enabled=true NtpServers=[1.1.1.1 2.2.2.2]
NTP Policy: Name=NtpTest3 Enabled=true NtpServers=[1.1.1.1 2.2.2.2]
NTP Policy: Name=NtpTest4 Enabled=true NtpServers=[1.1.1.1 2.2.2.2]
NTP Policy: Name=NtpTest5 Enabled=true NtpServers=[1.1.1.1]
```

What about debugging?



Let's introduce a bug ...

```
func main() {  
    //keyID := os.Getenv("IS_KEY_ID")  
    keyFile := os.Getenv("IS_KEY_FILE")  
  
    intersightClient, intersightAuthCtx, err := setupIntersightClient("INVALID!", keyFile)  
    if err != nil {  
        log.Fatalf("Error setting up Intersight client: %v", err)  
    }  
  
    getNtpPolicyListRequest := intersightClient.NtpApi.GetNtpPolicyList(intersightAuthCtx)  
  
    ntpResult, httpResult, err := getNtpPolicyListRequest.Execute()  
    if err != nil {  
        log.Fatalf("Error making Intersight API call: %v", err)  
    }  
  
    log.Printf("HTTP Status: %v %v", httpResult.StatusCode, httpResult.Status)  
  
    for _, ntpPolicy := range ntpResult.NtpPolicyList.Results {  
        fmt.Printf("NTP Policy: Name=%v Enabled=%v NtpServers=%v\n", *ntpPolicy.Name, *ntpPolicy.Enabled,  
            ntpPolicy.NtpServers)  
    }  
}
```

main.go (partial code listing)

and run it again ...

```
> go build && ./clus-intersight-go-example  
2022/05/03 12:35:08 Error making Intersight API call: undefined response type
```

It didn't work but we don't really know why ...

Let's enable debugging in the Intersight API client

```
func setupIntersightClient(keyID, keyFile string) (*intersight.APIClient, context.Context, error) {
    config := intersight.NewConfiguration()

    config.Debug = true

    client := intersight.NewAPIClient(config)

    // Set up the authentication configuration struct
    authConfig := intersight.HttpSignatureAuth{
        KeyId: keyID,
        PrivateKeyPath: keyFile,

        SigningScheme: intersight.HttpSigningSchemeRsaSha256,
        SignedHeaders: []string{
            intersight.HttpSignatureParameterRequestTarget, // The special (request-target) parameter
            "Host", // The Host request header specifies the domain name of the server
            "Date", // The date and time at which the message was originated.
            "Digest", // A cryptographic digest of the request body.
        },
        SigningAlgorithm: intersight.HttpSigningAlgorithmRsaPKCS1v15,
    }

    ...
}
```

main.go (partial code listing)

Now build and run with debug enabled

```
> go build && ./clus-intersight-go-example
```

```
2022/05/03 12:35:46
```

```
GET /api/v1/ntp/Policies HTTP/1.1
```

```
Host: intersight.com
```

```
User-Agent: OpenAPI-Generator/1.0.9.5808/go
```

```
Accept: application/json
```

```
Authorization: Signature keyId="keyID",algorithm="rsa-sha256",headers="(request-target) host date digest",signature="RIMXlls1WJA+MxCnah28kPAI0ZRMADbSPJFkkH3uU/wzzpyDK0TQ8BEx0B8md2hkCr0Ks9+wA
```

```
Date: Tue, 03 May 2022 02:35:46 GMT
```

```
Digest: SHA-256=47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=
```

```
Accept-Encoding: gzip
```

```
2022/05/03 12:35:47
```

```
HTTP/2.0 401 Unauthorized
```

```
Content-Length: 200
```

```
Cache-Control: no-cache, no-store, must-revalidate
```

```
Content-Security-Policy: base-uri 'self' https://cdn.intersight.com/; child-src 'self' https://www.intersight.com/ https://cdn.intersight.com/ https://cloudsso.cisco.com https://www.cisco.co
```

```
Content-Type: text/plain; charset=utf-8
```

```
Date: Tue, 03 May 2022 02:35:56 GMT
```

```
Pragma: no-cache
```

```
Server: nginx
```

```
Set-Cookie: AWSALB=RJB6Ks0NxmKLwbFKS11LgE+M60FirkLAiPhFGkm3Q0S47n10i9R11ShP8Roa80B5Hc7CnXYMdXE5I5hFtvoXgcjc9/Rup67L3Mdp1XaewARieH07oc2qCJpgOiek; Expires=Tue, 10 May 2022 02:35:56 GMT; Path=/
```

```
Set-Cookie: AWSALBCORS=RJB6Ks0NxmKLwbFKS11LgE+M60FirkLAiPhFGkm3Q0S47n10i9R11ShP8Roa80B5Hc7CnXYMdXE5I5hFtvoXgcjc9/Rup67L3Mdp1XaewARieH07oc2qCJpgOiek; Expires=Tue, 10 May 2022 02:35:56 GMT; Pat
```

```
Strict-Transport-Security: max-age=63072000; includeSubDomains
```

```
Via: 1.1 098fddbcdf00e65b8479d1d17b41d28a.cloudfront.net (CloudFront)
```

```
X-Amz-Cf-Id: fjfADLjZ0ve3x1Tu0SbtUHTMpXGESVUFjyhXP8qbk5JreTNYXDITwg==
```

```
X-Amz-Cf-Pop: SYD1-C1
```

```
X-Cache: Error from cloudfront
```

```
X-Content-Type-Options: nosniff
```

```
X-Content-Type-Options: nosniff
```

```
X-Frame-Options: SAMEORIGIN
```

```
X-Starship-Traceid: NB955325fb8a597a7cc801d679899c2cb5
```

```
X-Xss-Protection: 1; mode=block;
```

```
{"code":"InvalidRequest","message":"Cannot process the request. The keyId is malformed in this API-KEY request","messageId":"iam_apikey_missing_keyid_field_malformed","messageParams":{},"trac
```

```
2022/05/03 12:35:47 Error making Intersight API call: undefined response type
```

Now build and run with debug enabled

```
> go build && ./clus-intersight-go-example
2022/05/03 12:35:46
GET /api/v1/ntp/Policies HTTP/1.1
Host: intersight.com
User-Agent: OpenAPI-Generator/1.0.9.5808/go
Accept: application/json
Authorization: Signature keyId="keyID",algorithm="rsa-sha256",headers="(request-target) host date digest",signature="RIMXlls1WJA+MxCnah28kPAI0ZRMADbSPJFkkH3uU/wzzpyDK0TQ8BEx0B8md2hkCr0Ks9+wAj
Date: Tue, 03 May 2022 02:35:46 GMT
Digest: SHA-256=47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=
Accept-Encoding: gzip

2022/05/03 12:35:47
HTTP/2.0 401 Unauthorized
Content-Length: 200
Cache-Control: no-cache, no-store, must-revalidate
Content-Security-Policy: base-uri 'self' https://cdn.intersight.com/; child-src 'self' https://www.intersight.com/ https://cdn.intersight.com/ https://cloudsso.cisco.com https://www.cisco.co
Content-Type: text/plain; charset=utf-8
Date: Tue, 03 May 2022 02:35:56 GMT
Pragma: no-cache
Server: nginx
Set-Cookie: AWSALB=RJB6Ks0NxmKLwbFKS11LgE+M60FirkLAiPhFGkm3Q0S47n10i9R11ShP8RoA80B5Hc7CnXYMdXE5I5hFtvoXgcjc9/Rup67L3Mdp1XaewARieH07oc2qCJpgOiek; Expires=Tue, 10 May 2022 02:35:56 GMT; Path=/
Set-Cookie: AWSALBCORS=RJB6Ks0NxmKLwbFKS11LgE+M60FirkLAiPhFGkm3Q0S47n10i9R11ShP8RoA80B5Hc7CnXYMdXE5I5hFtvoXgcjc9/Rup67L3Mdp1XaewARieH07oc2qCJpgOiek; Expires=Tue, 10 May 2022 02:35:56 GMT; Pat
Strict-Transport-Security: max-age=63072000; includeSubDomains
Via: 1.1 098fddbdcdf00e65b8479d1d17b41d28a.cloudfront.net (CloudFront)
X-Amz-Cf-Id: fjjfADLjZ0ve3x1Tu0SbtUHTMpXGESVUFjyhXP8qbk5JreTNYXdITwg==
X-Amz-Cf-Pop: SYD1-C1
X-Cache: Error from cloudfront
X-Content-Type-Options: nosniff
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
X-Starship-Traceid: NB955325fb8a597a7cc801d679899c2cb5
X-Xss-Protection: 1; mode=block;

{"code":"InvalidRequest","message":"Cannot process the request. The keyId is malformed in this API-KEY request","messageId":"iam_apikey_missing_keyid_field_malformed","messageParams":{},"trac

2022/05/03 12:35:47 Error making Intersight API call: undefined response type
```

Now build and run with debug enabled

```
> go build && ./clus-intersight-go-example
2022/05/03 12:35:46
GET /api/v1/ntp/Policies HTTP/1.1
Host: intersight.com
User-Agent: OpenAPI-Generator/1.0.9.5808/go
Accept: application/json
Authorization: Signature keyId="keyID",algorithm="rsa-sha256",headers="(request-target) host date digest",signature="RIMX1ls1WJA+MxCnah28kPAI0ZRMADbSPJFkkH3uU/wzzpyDK0TQ8BEx0B8md2hkCr0Ks9+wAj
Date: Tue, 03 May 2022 02:35:46 GMT
Digest: SHA-256=47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=
Accept-Encoding: gzip

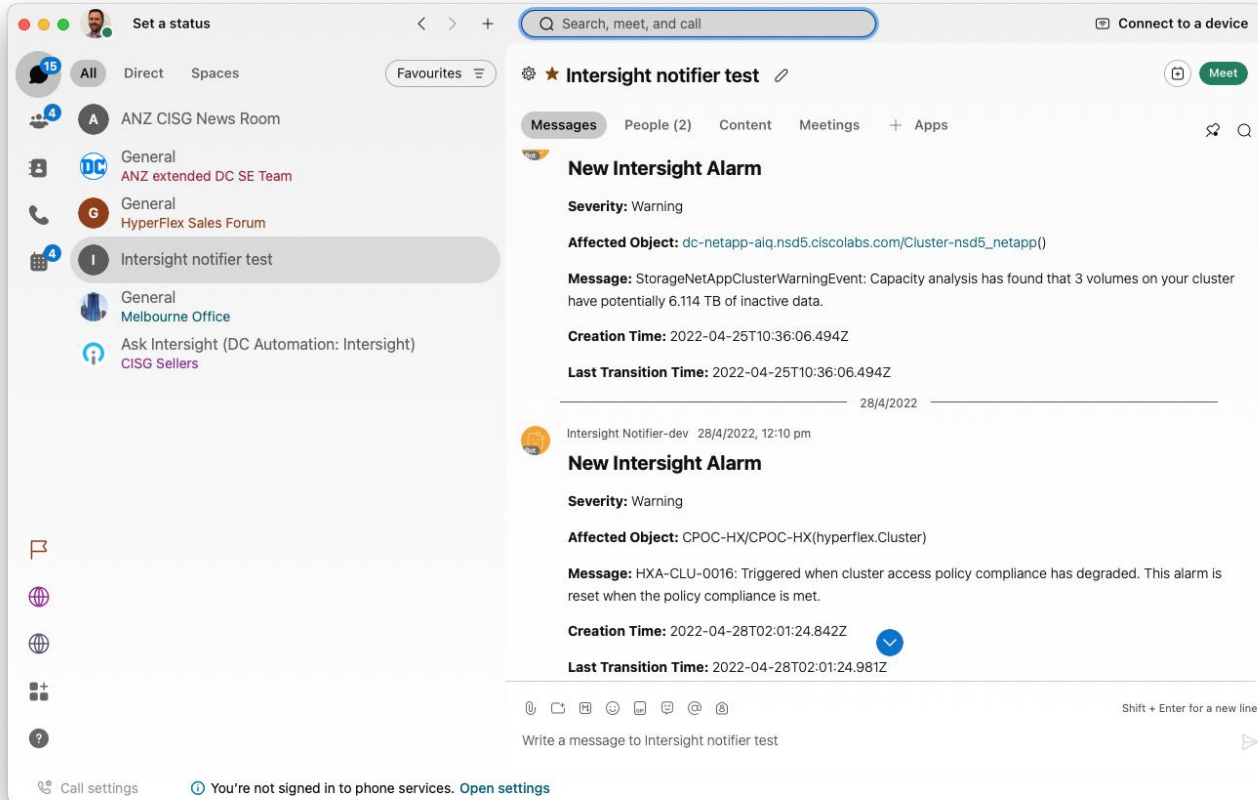
2022/05/03 12:35:47
HTTP/2.0 401 Unauthorized
Content-Length: 200
Cache-Control: no-cache, no-store, must-revalidate
Content-Security-Policy: base-uri 'self' https://cdn.intersight.com/; child-src 'self' https://www.intersight.com/ https://cdn.intersight.com/ https://cloudsso.cisco.com https://www.cisco.co
Content-Type: text/plain; charset=utf-8
Date: Tue, 03 May 2022 02:35:56 GMT
Pragma: no-cache
Server: nginx
Set-Cookie: AWSALB=RJB6Ks0NxmKLwbFKS11LgE+M60FirKLAiPhFGkm3Q0S47n10i9R11ShP8RoA80B5Hc7CnXYMdXE5I5hFtvoXgcjc9/Rup67L3Mdp1XaewARIEH07oc2qCJpg0iek; Expires=Tue, 10 May 2022 02:35:56 GMT; Path=/
Set-Cookie: AWSALBCORS=RJB6Ks0NxmKLwbFKS11LgE+M60FirKLAiPhFGkm3Q0S47n10i9R11ShP8RoA80B5Hc7CnXYMdXE5I5hFtvoXgcjc9/Rup67L3Mdp1XaewARIEH07oc2qCJpg0iek; Expires=Tue, 10 May 2022 02:35:56 GMT; Pat
Strict-Transport-Security: max-age=63072000; includeSubDomains
Via: 1.1 098fddbdcf00e65b8479d1d17b41d28a.cloudfront.net (CloudFront)
X-Amz-Cf-Id: fjjfADLjZ0ve3xlTu0SbtUhtMpXGE5VUFJyhXP8qbk5JreTNYXDitWg==
X-Amz-Cf-Pop: SYD1-C1
X-Cache: Error from cloudfront
X-Content-Type-Options: nosniff
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
X-Starship-Traceid: NB955325fb8a597a7cc801d679899c2cb5
X-Xss-Protection: 1; mode=block;

{"code":"InvalidRequest","message":"Cannot process the request. The keyId is malformed in this API-KEY request","messageId":"iam_ap
2022/05/03 12:35:47 Error making Intersight API call: undefined response type
```

Use Cases and Examples



WebEx Intersight Alarm Notifier



WebEx Intersight Alarm Notifier

```
func (i *IntersightPoller) pollIntersightAlarms(subscription *storage.Subscription, ctx context.Context) {  
    req := i.client.CondApi.GetCondAlarmList(ctx)  
    req = req.OrderBy("CreationTime desc")  
  
    // Only get alarms since the time of the last one. If we don't know the time of the last one, just get the last 2.  
    if subscription.AlarmLastCreationTime != "" {  
        req = req.Filter(fmt.Sprintf("(Severity in ('Critical','Warning','Info')) and (Acknowledge eq 'None') and (CreationTime gt %s)", subscription.AlarmLastCreationTime))  
    } else {  
        req = req.Filter("(Severity in ('Critical','Warning','Info')) and (Acknowledge eq 'None')").Top(2)  
    }  
  
    res, httpRes, err := req.Execute()  
    if err != nil {  
        log.Errorf("Failed polling faults: %v, HTTP Response: %v: %v", err, httpRes.StatusCode, httpRes.Status)  
        return  
    }  
  
    alarms := res.CondAlarmList.Results  
    for j := len(alarms) - 1; j >= 0; j-- {  
        alarm := alarms[j]  
        log.Infof("Alarm retrieved: %s", alarm.GetDescription())  
  
        if isAlarmNameIgnored(alarm.GetName(), subscription.IgnoredAlarmNames) {  
            log.Info("Ignoring alarm due to filter")  
            continue  
        }  
  
        i.webexBot.SendMessageToRoomID(subscription.WebexRoomID, alarm)  
    }  
}
```

WebEx Intersight Alarm Notifier

```
func (i *IntersightPoller) pollIntersightAlarms(subscription *storage.Subscription, ctx context.Context) {
    req := i.client.CondApi.GetCondAlarmList(ctx)
    req = req.OrderBy("CreationTime desc")

    // Only get alarms since the time of the last one. If we don't know the time of the last one, just get the last 2.
    if subscription.AlarmLastCreationTime != "" {
        req = req.Filter(fmt.Sprintf("(Severity in ('Critical','Warning','Info')) and (Acknowledge eq 'None') and (CreationTime gt %s)", subscription.AlarmLastCreationTime))
    } else {
        req = req.Filter("(Severity in ('Critical','Warning','Info')) and (Acknowledge eq 'None')").Top(2)
    }

    res, httpRes, err := req.Execute()
    if err != nil {
        log.Errorf("Failed polling faults: %v, HTTP Response: %v: %v", err, httpRes.StatusCode, httpRes.Status)
        return
    }

    alarms := res.CondAlarmList.Results
    for j := len(alarms) - 1; j >= 0; j-- {
        alarm := alarms[j]
        log.Infof("Alarm retrieved: %s", alarm.GetDescription())

        if isAlarmNameIgnored(alarm.GetName(), subscription.IgnoredAlarmNames) {
            log.Info("Ignoring alarm due to filter")
            continue
        }

        i.webexBot.SendMessageToRoomID(subscription.WebexRoomID, alarm)
    }
}
```


isctl – Intersight CLI

- *isctl* command structure is generated automatically from the API structure (<https://github.com/cgascoig/isctl>)
- `isctl <get|create|update|delete> GROUP CLASS [flags]`

```
> # Get all NTP Policies
> isctl get ntp policy

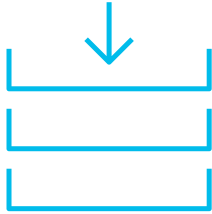
> # Get all rack servers
> isctl get compute rackunit

> # Create new NTP Policy named "NtpTest" in Org "default" with NtpServers "1.1.1.1" and "2.2.2.2"
> isctl create ntp policy --Name NtpTest --Organization default --NtpServers 1.1.1.1,2.2.2.2

> # Update existing NTP Policy named "NtpTest" in Org "default" with NtpServers "1.1.1.1"
> isctl update ntp policy name NtpTest --NtpServers 1.1.1.1

> # Delete existing NTP Policy named "NtpTest"
> isctl delete ntp policy name NtpTest
```

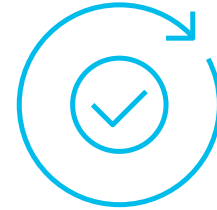
Other Examples



List NTP Policies



Alarm Streamer



Workflow Runner

<https://github.com/cgascoig/intersight-go-examples>

Conclusion



Key Takeaways

- Go is well suited to infrastructure automation tasks
- The Intersight Go SDK is generated automatically
- It is easy to get started with the Intersight Go SDK
- We can't wait to see what you build!

Technical Session Surveys

- Attendees who fill out a minimum of four session surveys and the overall event survey will get Cisco Live branded socks!
- Attendees will also earn 100 points in the Cisco Live Game for every survey completed.
- These points help you get on the leaderboard and increase your chances of winning daily and grand prizes.



Cisco Learning and Certifications

From technology training and team development to Cisco certifications and learning plans, let us help you empower your business and career. www.cisco.com/go/certs

Pay for Learning with Cisco Learning Credits

(CLCs) are prepaid training vouchers redeemed directly with Cisco.



Learn

Cisco U.

IT learning hub that guides teams and learners toward their goals

Cisco Digital Learning

Subscription-based product, technology, and certification training

Cisco Modeling Labs

Network simulation platform for design, testing, and troubleshooting

Cisco Learning Network

Resource community portal for certifications and learning



Train

Cisco Training Bootcamps

Intensive team & individual automation and technology training programs

Cisco Learning Partner Program

Authorized training partners supporting Cisco technology and career certifications

Cisco Instructor-led and Virtual Instructor-led training

Accelerated curriculum of product, technology, and certification courses



Certify

Cisco Certifications and Specialist Certifications

Award-winning certification program empowers students and IT Professionals to advance their technical careers

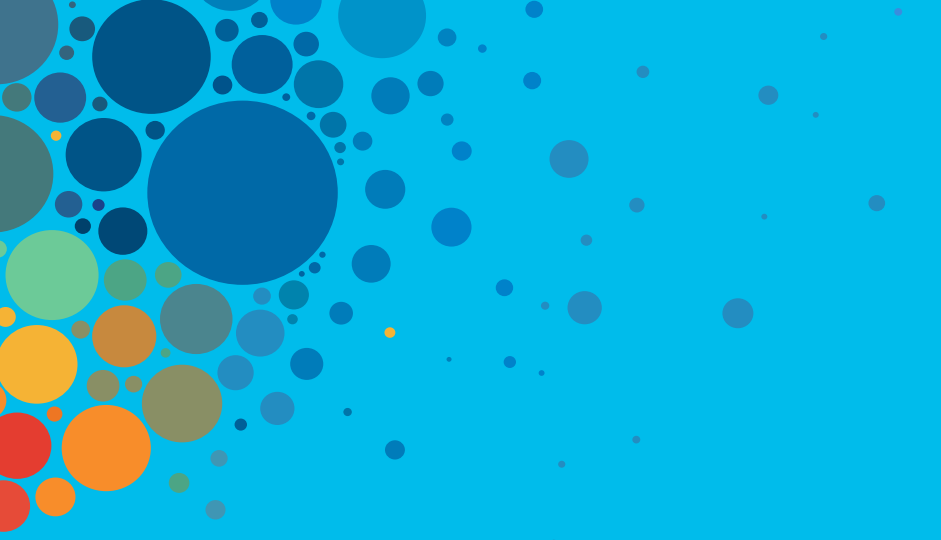
Cisco Guided Study Groups

180-day certification prep program with learning and support

Cisco Continuing Education Program

Recertification training options for Cisco certified individuals

Here at the event? Visit us at **The Learning and Certifications lounge at the World of Solutions**



Continue your education

- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Engineer meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand



The bridge to possible

Thank you

CISCO *Live!*



#CiscoLive