**Hey Network, Talk Data to Me**

With the introduction of Cisco DNA Center, Cisco made its network infrastructure easily programmable. However, it can be cumbersome to collect the relevant data using APIs, or click through a bunch of dashboards, when all you need is just a high-level overview of the network's health or the status of a specific device. Wouldn't it be nice if you could just chat with your network and ask it how it is feeling today or what switches are currently offline? Let’s have a walkthrough on the proof-of-concept related to an easy-to-extend framework written in Python that leverages the Cisco DNA Center APIs as well as Cisco Mindmeld, to provide a natural language interface to your network using Webex as the chat backend.

**From DevOps to NetDevOps and ChatOps**

DevOps is a hybrid between Development and Operations. The goal of DevOps is to shorten the development life cycle and provide continuous delivery with a higher software quality. The methodology in short is to “Build – Test – Deploy – Monitor”. The idea behind DevOps is that you have less complexity to manage, faster resolution of problems, and continuous delivery of software. To achieve the above-mentioned features, these are the requirements for DevOps:

* clear process of deployment
* data-driven decision-making environment
* automation

**How do you collaborate on your DevOps process?**

This is where ChatOps comes into the picture. It’s a collaboration model that triggers workflows from a chat interface. You will be able to communicate while maintaining your infrastructure. Apart from that you will have a central space for automated feedback. You can maintain (and log!) a timeline of communication, thereby leading to actions. In short, ChatOps help to keep everyone up-to-date.

**Why should you (as Network Engineers) care?**

Instead of making huge changes, if you could bring continuous, small changes, then it would be much more manageable and monitorable, which could help a lot in deployment. To summarize, NetDevOps brings the culture, technical methods, strategies, and best practices of DevOps to Networking.

**This leads to the following questions.**

* How do you trigger your network changes?
* How do you check your network?

This is where Chatbots come into play. A chatbot is a piece of software that conducts a conversation that (often) tries to mimic human behavior. Conversational AI is the process of creating personalized customer experiences.

A 10,000-foot Overview of what is that you are asking what is the status of your network and for that you need to get an answer as shown below.

To achieve the above feature let’s have the following architecture:

In this scenario you will be using Webex as the chat interface and DNA-C to get data from the network infrastructure.

**Let’s get Started with Notifier Bots**

A Notifier Bot will constantly check your infrastructure and gather the data. The Notifier Bot will alert you in case something is wrong. Please note its only observational.

Following is the JSON format on how data will be gathered from DNAC.

If anything goes wrong, you can monitor via DNAC dashboard. Same is the case with Meraki dashboard and in the case of ACI also. Instead of having three different dashboards open for monitoring why can’t you have the entire infrastructure team members in a single Webex room.

In the above scenario, if anything goes wrong in the network the “Notifier Bot” could trigger an alert to the WebEx room which can be handled by experts present in that room. Now the question is, can you also react without leaving Webex Teams?

To communicate with the bot, you can use “Command” Bots. You could implement some reactions such as checking the network, switch status, etc., based on the command types as shown in the example below.

Example

/delete project #12  
/get site\_health  
/get client\_details  
FF:FF:FF:FF:FF:FF

This is a way of writing commands and Implement reactions based on the strings. In the below example you can ask the command bot to list the devices present.

If you are only interested in switches, then try something like what you see below. To implement this command, you may need only around 6 lines of code, where the API is used for handling the filtering of the devices.

You can also grab details of any device , by using the following command:

**Command Bots – Pros & Cons**

Pros: It’s easy to write and once implemented its very powerful.

Cons: It’s highly dependent on the underlying infrastructure and you need to understand what command will help to trigger the task that you want to accomplish. An unintuitive Interface where the string passing could be error prone because of the requirement of manual programming.

Humans don’t say “delete #12”. Humans do say “delete the device with id number 12”.

**Building Natural Language interfaces is hard!**

If you ask a bot – “Call an Uber”. The reply may be there is no one with the name “Uber” in your contact list. Similarly, if you are asking the bot to show apartments in Barca the answer can be “I don’t know any city called Barca”. To overcome these kind of chat issues where the bot could understand what humans are looking for, Cisco implemented MindMeld.

It’s an end-to-end, state-of-the-art Natural Language Understanding, which can be deployed on-premises or on-cloud. It’s a Python-based framework and open source.

Cisco MindMeld Platform Architecture

You can ask the bot similar type of questions, as shown below, and get the same answer:

To get it working as shown above you need to feed it a ton of data, and you need to have training samples to make the program understand for example what a MAC address looks like. DNAC has a bunch of networks where you can generate the training samples. You can also use the Swagger which is available for DNAC and autogenerate more training data that are machine readable.

By using buttons and cards you can make web forms something similar as shown below.

This will help to update the device details if requested and displayed in the format of a form that can be easily updated from the Webex interface itself.

If you check this repo “[DNACenterInventoryWebexIntegration](https://developer.cisco.com/codeexchange/github/repo/gve-sw/DNACenterInventoryWebexIntegration)” you can find the code on how to update the Network inventory using DNA-C with the help of Chatbot. The workflow of the project is as shown below.

**Credit**

Cisco live presentation Hey network talk data to me - DEVNET-1330

**Reference**

* <https://github.com/cisco-code-projects/DNAC-Webex-Chatbot>
* <https://www.youtube.com/watch?v=S594va_thMY>