

Smarter Security Cameras: Filtering Camera Data from Meraki's MVSense Using On-Person Devices ~Andrew Afonso~

~The Background:

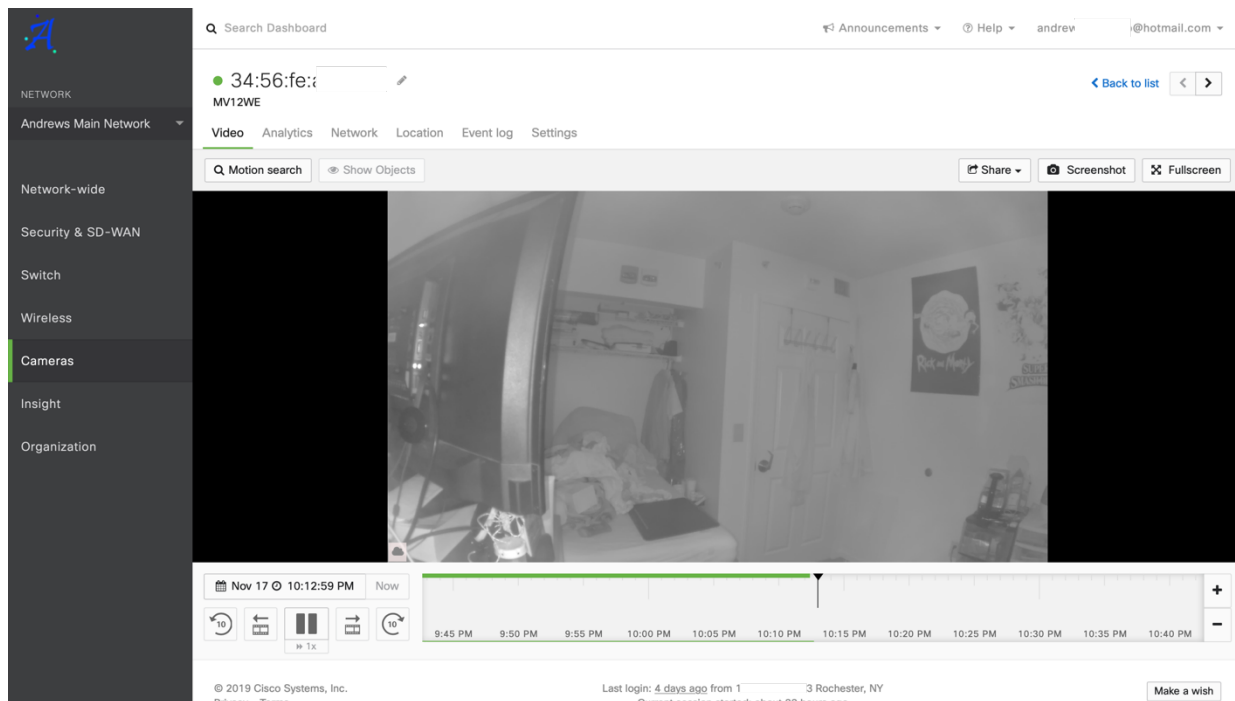
As someone who thoroughly believes in the power of automation, especially in everyday life, my thoughts often drift to something along the lines of “How can I make the computer handle that?” for a variety of tasks. One day, thinking about my MV12 (WE), my thoughts shifted to Smart cameras overall.

Often the “smart” functionality in a security camera simply entails some form of modified motion detection. Maybe differentiation between objects and people, maybe scheduled alerts, but nothing that really feels smart.

When it comes down to it, these cameras are only using their video information to perform filtering, or maybe some information supplied from the user. I can have Siri turn on my lights when I get home, change the temperature when I get home, so why doesn't my security camera know when I'm home, and only alert me of things when I am away? Thankfully, mine can, pretty easily, thanks to the Meraki ecosystem (The M in MV).

~Meraki

Meraki makes cloud managed networking products. Their most recent product line is the MV cloud connected camera. Starting with MVx2 series, the cameras are now “smart” including computing components allowing for video analytics at the endpoint. Within the past year a new product, MVSense, was introduced which provides API's to interact with the cameras and their data in real-time. This allows us to bring the people detection functionality of the camera into a Python script.



Smarter Security Cameras: Filtering Camera Data from Meraki's MVSense Using On-Person Devices ~Andrew Afonso~

Meraki's first and main product is their line of AP's, but they make routers (firewalls) as well as switches. All of these devices have API's that allow insight into the details of your entire network. This allows us to be able to pick a few network devices, such as a phone or laptop, as devices we know will generally leave the house with us, and have our script look for those devices to determine if we are likely home or not.

Additionally, some Meraki AP's have integrated Bluetooth radios as well, allowing for BLE (Bluetooth low energy) device detection. You might ask why we would need to use a BLE device as an on-person device if we can just check the network clients. In networks with hundreds, or thousands of clients, unthreaded parsing of that list takes time, and sometimes you forget your phone or laptop at home, or simply don't want to take them with you. With BLE tags being available for as low as \$10 (trackr), you can simply put one in your wallet, purse, backpack, or on your keys, pretty much on any object that always leaves the house with you. Then you can remove the scripts network client search, and just check for the BLE tag.

The screenshot shows the Meraki dashboard interface. On the left is a dark sidebar with navigation links: NETWORK, Andrews Main Network (selected), Network-wide, Security & SD-WAN, Switch, Wireless (highlighted), Cameras, Insight, and Organization. The main content area is titled 'Bluetooth clients for the last two hours'. It includes a search bar and a table with 14 Bluetooth clients. The table columns are: Status, Description, Last seen, Last seen by, Manufacturer, Connectivity, and Tags. The clients listed include various Apple devices (iPhone, iPad, MacBook) and Bose devices (SoundLink Mini, SoundLink Revolve, etc.). The connectivity bar for each client shows a green bar indicating signal strength. At the bottom of the dashboard, there is a footer with copyright information (© 2019 Cisco Systems, Inc.), login details (Last login: 4 days ago from Rochester, NY), session information (Current session started: about 23 hours ago), and a link to the privacy policy (Data for this organization is hosted in North America).

Status	Description	Last seen	Last seen by	Manufacturer	Connectivity	Tags
<input type="checkbox"/>	f0:18:98:...	Nov 17 23:02	e0:cb:bc:...	Apple	<div></div>	
<input type="checkbox"/>	dc:a9:04:...	Nov 17 23:09	e0:cb:bc:...	Apple	<div></div>	
<input type="checkbox"/>	a4:83:e7:...	Nov 17 23:11	e0:cb:bc:...	Apple	<div></div>	
<input type="checkbox"/>	Tile	Nov 17 22:30	e0:cb:bc:...		<div></div>	
<input type="checkbox"/>	Tile	Nov 17 22:39	e0:cb:bc:...		<div></div>	
<input type="checkbox"/>	LE-reserved_T	Nov 17 23:11	e0:cb:bc:...	Bose	<div></div>	
<input type="checkbox"/>	LE-reserved_E	Nov 17 22:52	e0:cb:bc:...	Bose	<div></div>	
<input type="checkbox"/>	LE-Sonar	Nov 17 23:11	e0:cb:bc:...	Bose	<div></div>	
<input type="checkbox"/>	LE-ShadyT's BSP	Nov 17 23:11	e0:cb:bc:...	Bose	<div></div>	
<input type="checkbox"/>	LE-Nightshade	Nov 17 23:00	e0:cb:bc:...	Bose	<div></div>	
<input type="checkbox"/>	LE-Deep Space Fine	Nov 17 22:52	e0:cb:bc:...	Bose	<div></div>	
<input type="checkbox"/>	Ember Ceramic Mug	Nov 17 23:11	e0:cb:bc:...		<div></div>	
<input type="checkbox"/>	BLE_367E	Nov 17 22:22	e0:cb:bc:...		<div></div>	
<input type="checkbox"/>	14:99:e2:...	Nov 17 23:11	e0:cb:bc:...	Apple	<div></div>	

~The Idea

Taking all this into account, I had a simple idea. *A camera only needs to notify you of motion detected if you aren't home.* Normally this isn't something a security camera would be able to determine, outside of potentially using facial recognition 🤖 to do so. But using Meraki's API's, we can determine if someone is home or not very easily by looking to see if the devices that person carries with them each time they leave the house are present on the network.

Smarter Security Cameras: Filtering Camera Data from Meraki's MVSense Using On-Person Devices ~Andrew Afonso~

~Finalized Concept

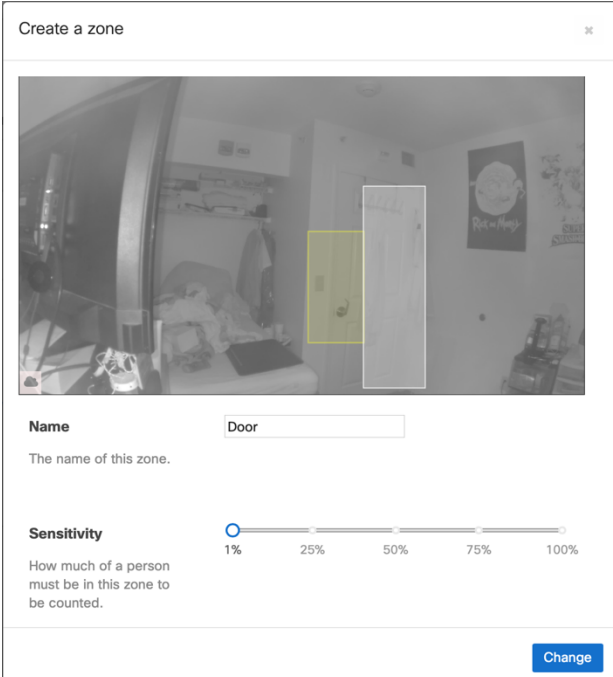
"If a person is detected by the camera, check if any trusted devices are on the network, or detected nearby (Bluetooth). If none are, send a priority alert. If the devices are on the network or nearby, it is unlikely to be a notifiable event."

~Implementation:

Implementation is easy enough, this is not an insane idea, but at the same time, like all things Meraki it's a tad wonky.

There are no API calls to check if there is 'motion' for a camera. To create automations based on motion alerts, you would need a Webhook server to be a recipient of the motion alerts. This project focuses on using only the MVSense and Dashboard API's, so adding functionality that requires a Webhook server was not ideal.

What you **are** able to query via the MVSense API is a count of the 'people' currently detected in the 'zones' on the camera. Zones are a tool that let you specify areas of the cameras FOV (field of view) to monitor in different ways. Particularly useful in retail applications, where you want to know where people are walking and spending their time, the feature is more of a 'hack' for us to check for motion.



Create a zone

Name

The name of this zone.

Sensitivity

How much of a person must be in this zone to be counted.

1% 25% 50% 75% 100%

Change

On my Dashboard, I added a new zone to my camera over the area that I want to monitor for movement, my door, and set the sensitivity to 1%. This might sound counter intuitive, and it is. The setting is:

"How much of a person must be in this zone to be counted"

Which means 1% is actually the most sensitive. This means that pretty much any movement or motion will be counted as a 'person', and will cause the count of people returned to the API call to increase. Bingo!

A 'zone 0' already exists for each camera, and is assigned to the cameras entire FOV. But without knowing the sensitivity of zone 0, it is a good idea to add in the super sensitive zone(s) for areas of interest.

Smarter Security Cameras: Filtering Camera Data from Meraki's MVSense Using On-Person Devices ~Andrew Afonso~

~Pseudocode

1. Setup / Read in config.
2. While 1==1:
 - a. Check if number of 'people' detected in any zone is > 0.
If false, keep looping.
If true:
 - i. Get list of MAC's devices on the network that have had activity within the last half hour.
 - ii. Get list of MAC's of BLE devices seen by the network within the past minute.
 - iii. If any of the MAC's returned match one of the on-person device MAC's, do nothing. (Because you are home)
If none of the devices match:
 1. Do something.
Currently the script writes to a .log file, but this could be customized to send email alerts, to post to a server somewhere, to do a number of things. The important part is knowing if motion is worrisome or not.

~Setup

To set this up yourself you will just need to do a few things:

1. Enable Meraki API access, store your api key somewhere safe.
https://documentation.meraki.com/zGeneral_Administration/Other_Topics/The_Cisco_Meraki_Dashboard_API
2. Enable MVSense for the target camera. If your camera supports MVSense, you will have 10 free licenses.
3. Run monitorMV.py

That's it!

~Future Improvements

- Thread client detail parsing to improve speed.
- Implement Webhooks to allow processing of motion events.
- Implement email notifications.
- Actual GUI