Prioritize: Inferring Slack of Network Traffic

Peter Vondras, Tufts University

December 14, 2016

1 Abstract

Given that data, once passed to the network, travels to destination with no thought of when it will be used, this project builds on the framework laid out in SANS, [1] focusing on potential benefits, risks and feasibility of inferring the acceptable slack of network traffic.

This project introduces Prioritize, a network traffic analyzer that infers the importance of a Nest Indoor Security Camera's communication packets. It is hoped that this information might be generalized and expanded upon to create a wider toolset that could be leveraged for more efficient processing of a highly stressed network.

2 Introduction

Webcams create a considerable amount of continuous upload traffic, yet it is unlikely that the information that is being sent is of immediate importance at all times. To this end, this paper assumes that if a webcam is watching an unmoving scene, referred to as inactive, the network traffic has more slack than if the scene is changing, referred to as active. The problem main problem then is inferring when the camera is inactive.

By analyzing the network traffic of the Nest Cam during inactivity, activity, and the transition between, it became clear that:

1. The throughput of the camera while inactive is measurably lower than when it is active.

2. There are predictable bursts of data when the camera is inactive which repeat over a 4000ms interval.

3. There is a measurable increase in bursts when the camera is active.

Leveraging this, Prioritize reliably recognizes a transition from inactive to active or vice versa in an uncongested network. What is more, Prioritize will also pick up these transitions during an extended TCP upload which is hard to impossible to do by inspecting packets by eye. However, if the network is overwhelmed by UDP packets, Prioritize will downgrade both active and inactive cameras to an inactive classification.

The rest of this paper is organized as follows. Section 3 depicts the experiments that were run, how they were setup and their hard results. Section 4 describes how Prioritize was designed and works along with its performance. Section 5 and 6 discuss the meaning of the results and touch on what is left to be done in the future and section 7 concludes the paper's findings.

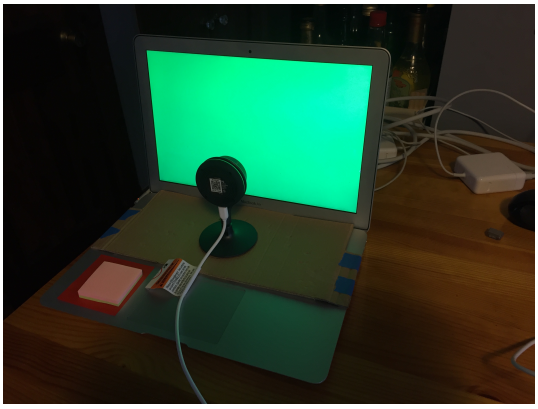

Figure 1: Camera Setup

3 Experiment

3.1 General Setup

* In order to have repeatable results the camera was aimed at a macbook air 13in Mid 2011 laptop screen on high brightness. A rudimentary alignment jig was used to aid repeatable setup. See Figure 1. A thick shroud was draped over the setup which ensured that no shadows fell across the screen during testing. The image was then monitored via a streaming connection provided by nest.com. Using this setup, a fullscreen image or video encompasses the entire image.

There are several ways to capture the camera's traffic, one such way is to create a wireless hotspot with another computer and then monitor that computers ethernet traffic. Data was collected using tcpdump using the following command which will filter out all traffic that is not the camera uploading data to the cloud: "sudo tcpdump -n -i en7 dst 52.201.218.19 and dst port 443"

*This experiment and the Prioritize source code was created for dual purpose, the exploration of inferring slack of network traffic and as an example of how even the use of well designed and secure IoT products exposes the user to risk and possible privacy invasion. Therefore, sections three and four are replicated for the use of both perspectives.

3.2 Test Cases

The following experiments were executed:

1. Inactive Tests without local network congestion: Traffic was captured of the camera monitoring a green, blue and red image. Each run was one minute in duration and was preceded of at least a minute of the same image to ensure that no change was recorded. There were ten runs completed of each of the three image types.

2. Active Tests without local network congestion: A video of strobing single color screens was downloaded and played in front of the video camera for one minute.[2] This experiment was performed ten times.

3. Transition between Inactive and Active Tests without local network congestion: The camera is aimed at the strobe video which is paused on a single color image. Every minute, the video is resumed or if it is running, paused. This is done for a total of twenty minutes, yielding twenty transition from Inactive to Active and twenty from Active to Inactive.

4. Transition between Inactive and Active Tests with extended TCP upload congestion: The transition test above was repeated for four minutes with the addition of a TCP upload being run on the local network to a host out of the network. Iperf was used to simulate this. The measurement was started at least 10 seconds after the upload traffic was started to allow the router buffer to fill up.

5. Transition between Inactive and Active Tests with extended heavy UDP upload congestion: The transition test above was repeated for four minutes with the

Average Throughput over all tests

Run Num	blue	green	red	strobe
1	0.173	0.180	0.187	0.455
2	0.175	0.181	0.184	0.460
3	0.175	0.182	0.183	0.456
4	0.180	0.182	0.182	0.455
5	0.179	0.183	0.186	0.453
6	0.177	0.179	0.185	0.444
7	0.180	0.183	0.183	0.456
8	0.177	0.183	0.185	0.445
9	0.177	0.180	0.182	0.456
10	0.180	0.181	0.186	0.459
AVG	0.177	0.181	0.184	0.454
STD	0.002	0.001	0.002	0.005
STILL AVG	0.181			
STILL STD	0.003			

Figure 2: Average throughput of experiments

addition of 20Mbps UDP upload being run on the local network to a host out of the network. Iperf was used to simulate this. A speed test measured the upload bandwidth at 9.85Mbps. The measurement was started at least 10 seconds after the upload traffic was started to allow the router buffer to fill up.

3.3 Results

The average throughput (Mbps) of all runs of the Inactive and Active tests are summarized in Figure 2. It is clear from these tests that over a one minute period, there is a statistically significant difference between an Inactive and active camera in terms of throughput.

Additionally, looking at the packets with Wireshark, we can make some observations regarding delivery time and size. The first observation is that packets seem to be sent at two different speeds, either bursts where the interval between packets are less than 5ms or sporadically, where the interval is on the order of 20-60ms, which will be referred to as slow. For the bursts, every packet, except perhaps the last one, is the max size allowable (1514 bytes). This allows us to infer that each of these bursts represents some logically connected pieces of data. If you count every burst as a single packet, there are on average 27 chunks of data sent every second. As the camera reports to supply 30 fps, it is likely that each chunk is the requisite data to update the image.

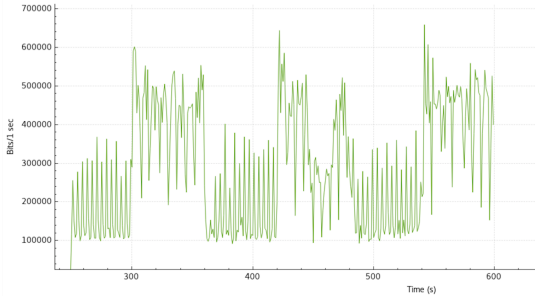

Figure 3: Average throughput of experiments

Another observation is that a large burst starts every 4000ms. Between this time and the next cycle, the throughput drops and although their might be a burst, it is less likely and will inevitably be smaller.

Observing the active traffic does not yield much information other than the fact that the 4000ms cycle is replaced by frequent and often bursts or varying length. Even with low traffic on the local network, there were sporadic low traffic segments during an active period, lasting up to 750ms.

The transition experiment without local network congestion looked much like alternating between the inactive and active one minutes tests, but both the the TCP and UDP traffic tests obscured the patterns. The number of bursts during inactive segments was highly increased so as to mask its presence. That being said, the throughput was still an accurate measure of activity for the TCP test. Although the throughput is generally much higher during Activity, there are significant

periods low throughput during a period of Activity. This is shown best by the 20 second period in the second active segment in Figure 3 in which the throughput looks much like an Inactive period.

The UDP test seemed to throttle the camera's throughput permanently, likely due to TCP back-off from heavy packet loss.

4 Description of Prioritize

4.1 Design

Prioritize has three modes, Startup, Inactive and Active. Figure 4 shows a state diagram with the general transition logic between the three states. To navigate between the three modes, Proactive uses six variables:

1. Throughput Window (TW): Proactive continually measures the throughput of the camera in Mbps over a period of time. That period of time is called the Throughput Window. It does this by adding every new packet to a Byte_Sum and keeping a Tail pointer to the oldest packet in the Byte_Sum.

2. Throughput Threshold (TT): This is a value in Mbps that is used in the logic to trigger a mode change. Proactive uses 0.23Mbps globally for the nest camera, which was determined through experimental measurements and empirical testing to be both stable and responsive.

3. Burst Count (BC): As observed in the testing, data comes in either slow intervals (20-60ms) or in bursts of max sized packets within 5ms of each other. Proactive keeps track of the number of these bursts over a specified window. Single max sized packets are also considered bursts as they are rare during inactive mode.

4. Burst Window (BW): The period of time back that the system keeps track of the Burst Count. The Burst Window is set globally to 1500ms using the same methods as the TW.

5. Inactive Time: The amount of time that camera has been below the TT.

6. Active Time: The amount of time that the camera have been active.

Proactive starts in Startup mode and averages the throughput over the TW. 3800ms is a likely time for the TW because it is the longest window that you are guaranteed not to have two large bursts during an Inactive cycle. The throughput deviation over this period is higher than over the entire minute because it is over a much smaller time but also because, in order to ensure that there is never two large bursts in the TW, there are short periods when there is no large burst at all which pulls the throughput down dramatically. If after the initial TW, the throughput is less than the TT, Proactive moves to Inactive mode, otherwise, it will categorize the camera as Active.

When Inactive mode is triggered, if it is not already, the TW is set to 3800ms. Additionally, Proactive will attempt to backfill the packets into the throughput measurement if the TW was increased. It will stop backfilling if it gets to a packet of max size. This backfill feature helps dramatically with flickers back and forth at the tail end of Action mode. Active mode will be triggered if the throughput crosses the TT.

Upon entering Active mode, the system will change the TW to 1200ms after waiting at least that amount of time by monitoring the Active Time. Reducing the TW lowers the response time for recognizing the end of

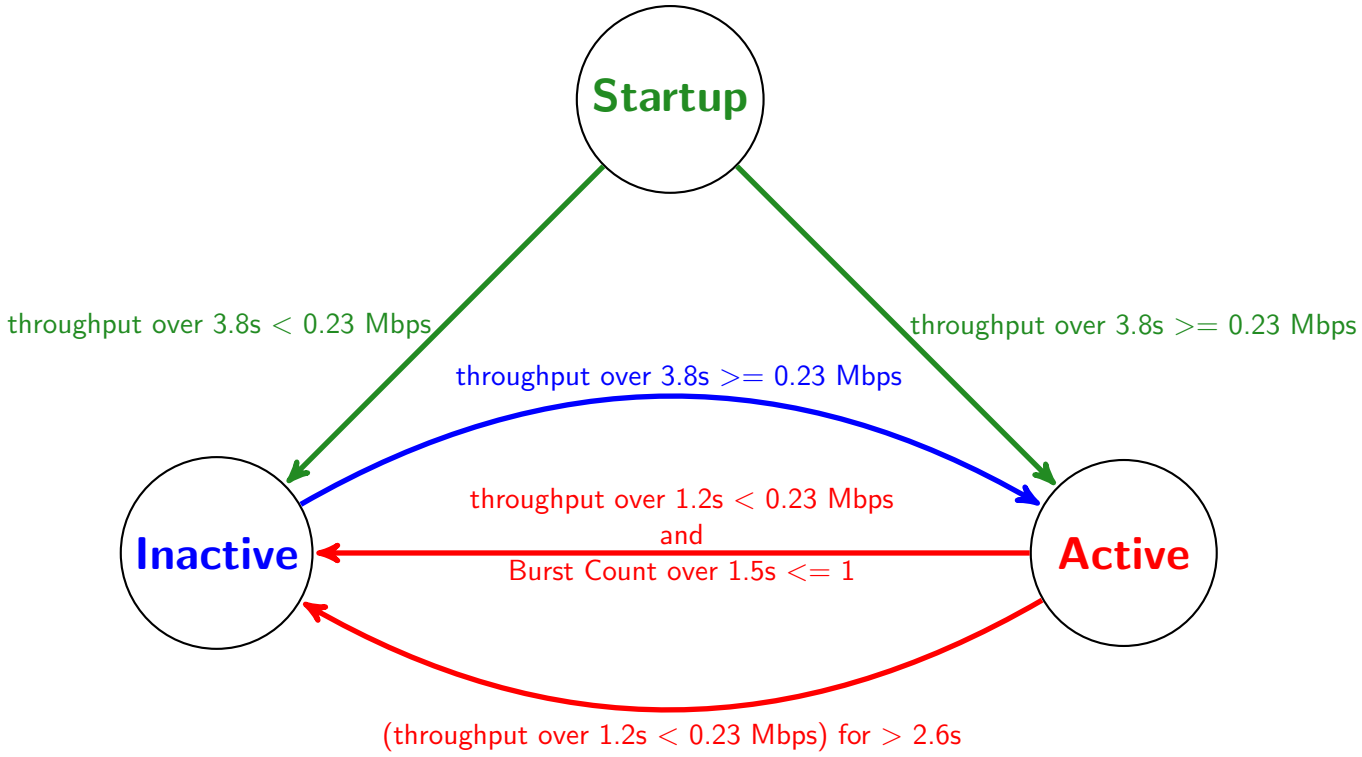

Figure 4: State Diagram

	Time to classify traffic w/o Interference (s)	
Run Num	Recognise Active	Recognise Inactive
1	0.523	1.501
2	0.896	2.117
3	0.437	1.270
4	0.430	1.535
5	1.418	1.528
6	1.036	1.438
7	1.554	1.517
8	1.300	1.467
9	0.676	1.554
10	1.314	
AVG	0.958	1.548
STD	0.427	0.230

Figure 5: Time elapsed for Proactive to change modes compared to when the camera actually does change traffic modes

an Active period. There are two paths for Proactive to reclassify the camera as Inactive:

1. The camera crosses below the TT while the Burst Count is also less than or equal to one. This will allow the single large burst in an inactive mode to not interfere with the end of an Active period. The Burst Count maximum also provides smoothing during an Active period such that random drops in throughput do not inadvertently trip a false transition change. These false changes were present even when the TW was not decreased.

2. The Inactive Time is greater than 2600ms.

4.2 Performance

Proactive was able to detect activity on all data collected in this experiment with low lo-

Packets to classify traffic w/o Interference		
Run Num	Recognise Active	Recognise Inactive
1	22	40
2	50	52
3	29	34
4	27	43
5	49	46
6	36	38
7	66	41
8	46	40
9	30	41
10	47	
AVG	40.2	41.7
STD	13.6	5.1

Figure 6: Packets between Proactive changing modes and the camera actually changing traffic modes

cal network congestion. On average, Action was detected 958ms (See Figures 5 and 6) after it actually occurred. Inactivity was also detected reliably at an average of 1548ms after the camera returned to normal. The algorithm used the Burst Count and TT method to return to Inactive mode every time. No jitter between modes or false classifications were observed.

Under heavy TCP traffic, Proactive also was able to reliably classify the camera and detect mode changes. It was not possible to figure out the exact packet that the camera actually changed modes as it was in the uncongested tests. In fact, Proactive was able to downgrade the mode to Inactive about three seconds before I could by inspecting packets. In all cases, the system used the Inactive Time to transition back to Inactive mode. No jitter between modes or false classifications were observed.

The heavy UDP traffic forced the nest camera to drop its throughput below the TT regardless of what the camera was seeing. This is a limitation of the current system. It is possible to see this as acceptable as under extreme network distress, it is likely that deprioritizing camera streams would be prudent.

Despite Proactive's ability to detect camera activity, it would be ideal to speed up the classification time, especially on the side of recognizing action. This is because, the measurements show that there is an entire second, when potentially critical information is being delayed. The time for classification of Inactive is acceptable as it is conservative of the camera's network needs.

5 Discussion

This tool is envisioned to be most helpful on a network with many cameras, where the total bandwidth used is significant enough that it might be advantageous to be able to wait for a convenient moment to burst a larger segment of data. It is worth noting however that once data has been analyzed "organized" for efficiency, that it will not have the same characteristics as before. To gain similar benefits higher up in a network architecture, it would be necessary to mark packets at this step.

Another interesting issue is security. This is a first step toward inferring intent based of what was opaque activity. This tool could be used to monitor whether a specific camera was Active and therefore could inform someone with malicious intent when someone either was home, in a specific room, or has left the house. What is most bothering about this is that the tool does not need access to the unencrypted data that the camera is sending and therefore encryption is not a solution. One way to obscure this tools ability to track the camera would be to change the destination IP and port from time to time. Another, albeit wasteful way would be to send a consistent amount of traffic or random decoy bursts. Finally, using a VPN might also be a helpful precaution.

One question that will bear more thought is what the effects of advancing methodologies

of intent inference will be on privacy? Perhaps an argument could be made that an advanced local network might make these inferences and then homogenize the network traffic to make similar inferences harder to make farther upstream.

6 Future Work

Moving forward, it is imagined that more adaptive systems could be developed that would tune themselves to a new camera type. It might be possible to feed a streaming video to the tool in order for the tool to automatically classify network traffic as Active or Inactive during a setup stage. Additionally, the repeated Inactive burst pattern was not leveraged as heavily as it might have been which points to faster detection being possible.

Additional improvements include, ability to sniff networks directly and modify or delay packets and the ability to monitor multiple cameras at the same time, but the end goal would be to use this information to increase the productivity of a network.

7 Conclusion

By analyzing the network traffic of a Nest Indoor Security Camera, a reliable algorithm has been implemented to classify the importance of the traffic being sent. Proactive was shown to be able to detect the mode of a camera in seconds in both uncongested and also heavy TCP upload scenarios. The hope is that this tool might be used in the future in to better utilize upload bandwidth prioritization when many cameras are present.

8 Acknowledgements

Special thanks are given to Fahad Dogar for his guidance around experiment design and next steps. Additionally, Ming Chow is appreciated for opening my eyes to the way information might be used with malicious intent and the importance of understanding what we reveal when using the internet.

References

[1] Fahad Dogar, Sana Farrukh and Mamoon Raja *Slack-Aware Networking for Emerging Machine-Centric Applications* 2016 Draft.

[2] 12 Color Strobe Light Effect *https://www.youtube.com/watch?v=FTYY9pwlxjE*