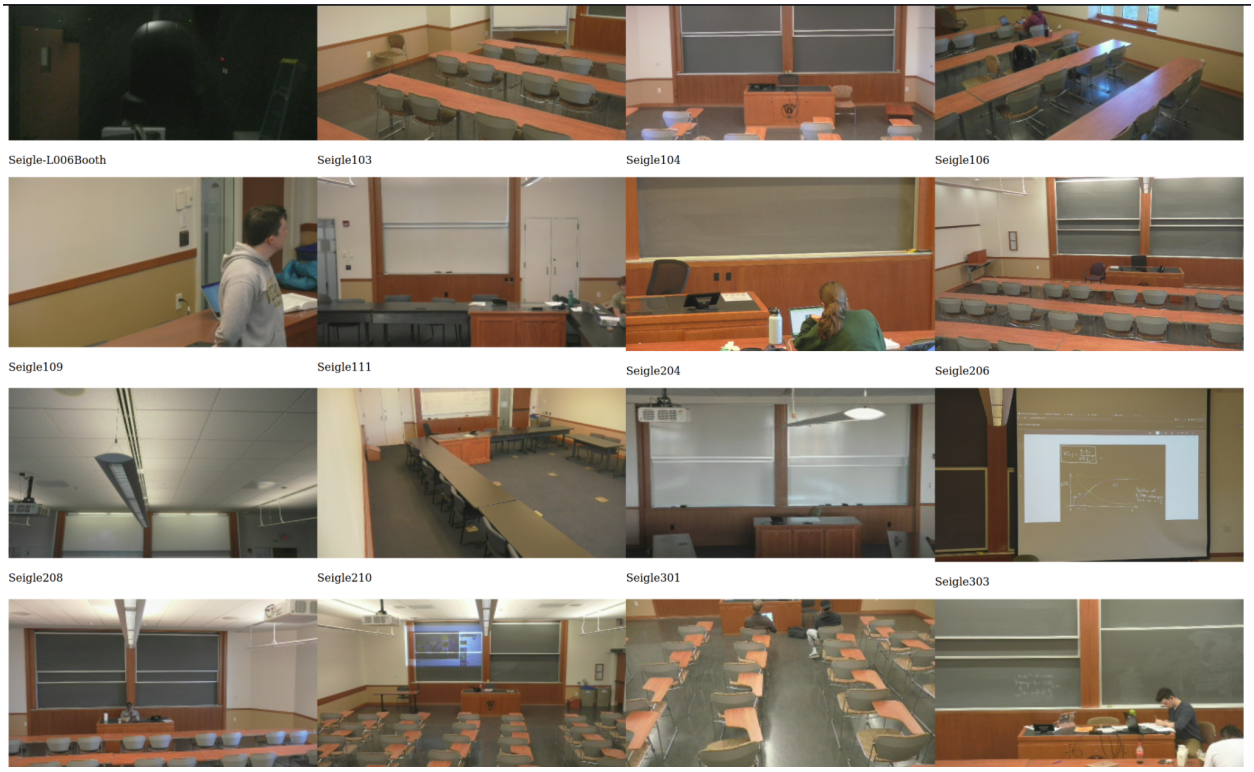
CSE 433 Final Project

Authors: Garrett Ramsey, Jasper Sands, Grayson Martin, Mac Wolf

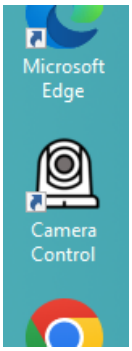
Overview

For our final project, we want to shed light on and responsibly report what we believe to be a concerning security vulnerability for the WashU community. We have found a large number of classroom conference cameras that have zero authentication requirements to connect to and manipulate. These cameras have surprising coverage, being able to move and zoom in on pretty much any part of the room with surprising accuracy and detail. Our intention with this report is to responsibly report this lapse in the school's security mechanisms. We will demonstrate the various different ways that unfettered access to these cameras can be exploited for malicious intent and give reasonable recommendations to patch this vulnerability.



Our Dashboard Webcam streaming every open conference camera on campus (page 1 of 9).

Discovery



Earlier this semester, Grayson was doing homework in an empty WashU classroom. He got bored and decided to take a stroll around the room, coming up to the teacher's desk where he found the computer at the front of the room opened up to the camera controls of the room. He found that he could move around the camera, zoom in, turn off, or perform numerous other functions. Noticing that there was an IP address as the URL, he typed it into his own computer and realized it was accessible to anyone on the WashU network.

Weeks 1-2: Device Investigation and Network Scan

We first wanted to determine more about the cameras themselves. We inspected the initial camera and determined that it was a Panasonic AW-HE38, an HD camera with both wired and IP connectivity. We found the manual online, and from there we were able to determine more details about the camera specs themselves, as well as that they could all be password protected. Given that this single camera chose not to enable that feature, we were curious if the other cameras around campus were similarly unsecured. To test our theory, we wrote a bash script that scans the entire network. It would send a ping to each IP address it found, and if a response was received, it would send a request that is unique to the particular model of camera, requesting its name. This so happened to be the building name and room number where the camera was located, and we compiled the resulting names and IP addresses into a list, which can be found in our repository linked at the bottom of this report. To further expand our search and see if there were cameras of other models, we modified our bash script to search by IP bytes. We began by searching the lowest byte and found a number of random websites and links among some of the cameras we had already discovered. We then expanded our search to the lowest 2 bytes, and found a handful of new cameras in different classrooms. We also briefly attempted to scan 3 bytes by making our search multithreaded, but even then it would have taken multiple days, so we left it at our 2 byte scan. At the end, we found 126 cameras, however with more time we likely could have found more cameras or other sensitive information.

Weeks 3-4: Camera Feeds and OpenCV

The cameras provide an MJPEG (Motion JPEG) stream via HTTP. In response to a standard HTTP GET request at the /mjpeg endpoint, the camera sends a never-ending response of complete JPEG images separated by a "content boundary". Each image is a frame within the live stream video. Although a slightly outdated and highly inefficient streaming protocol, MJPEG is very easy to interact with programmatically.

By simply requesting a few images from the stream URL and terminating the TCP connection early (a MJPEG stream never ends, so they are always terminated "early") once we've acquired as many frames as we desire, they can be saved to create a video recording, or passed through

any number of computer vision or image processing algorithms. We used OpenCV's implementation of MOG2 motion detection to automatically process the feeds of all cameras in near real-time.

Attack Vectors

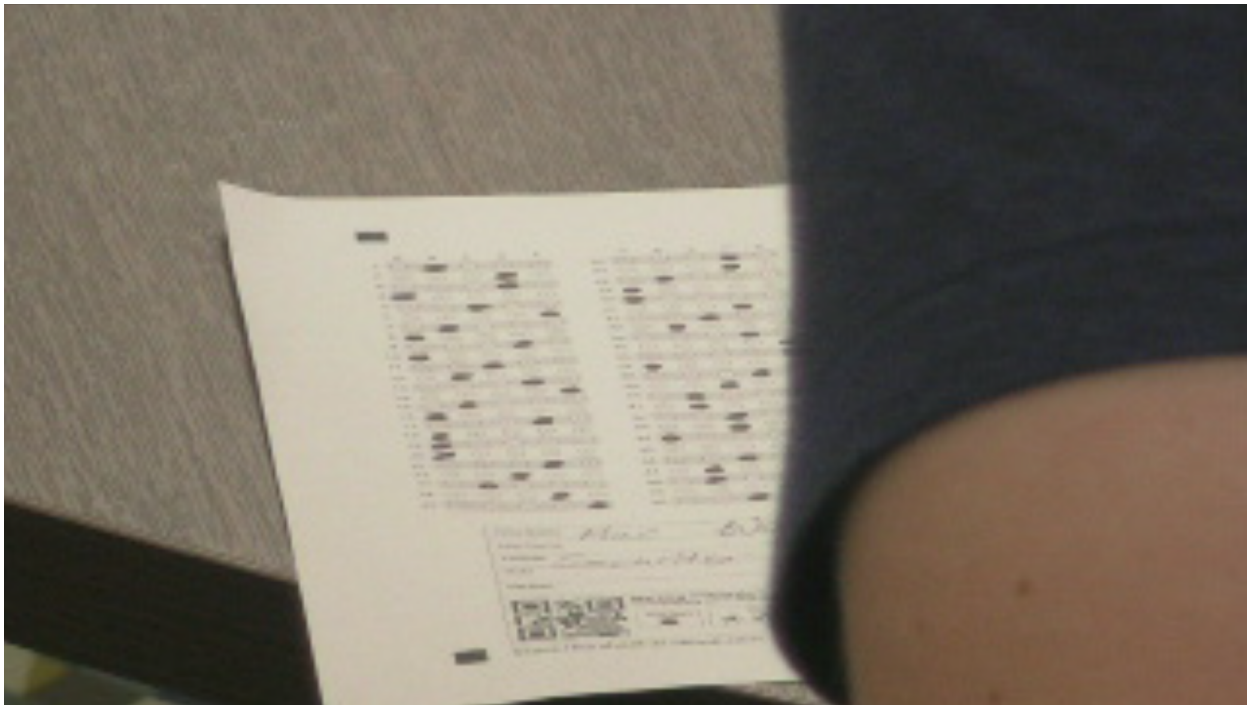
Academic Integrity

Gaining access to a camera in a classroom presents multiple issues regarding academic integrity. Two examples are listed below:

1) Falsifying Attendance
   Many classes have codes or short quizzes (such as PollEV) presented at the beginning of lecture that must be scanned or completed in order to receive attendance credit. If one is able to view the screen without actually being present in the room, it's easy to falsify your attendance.

Screenshot from a Classroom Camera displaying a working Attendance QR Code

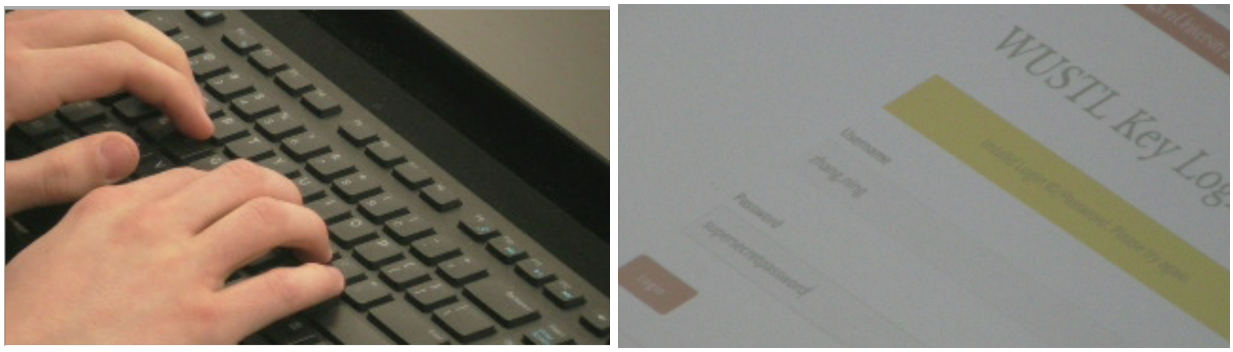
2) Cheating on Exams
   Using the impressive zoom of the cameras, it is quite feasible to move and zoom in on a student's exam while they are taking it, and analyze the image to figure out what their answers are. We staged a fake exam to demonstrate how easy it is to zoom in on someone's paper remotely (See image below).



Screenshot of a classroom camera zoomed in on a student's (fake) exam

Side Channel Attack

It is common for students and professors to log into accounts in classrooms. The high-resolution zooming capabilities of the conference room cameras enables an adversary to record keystrokes from keyboards or view plaintext information on screens. This attack can also be replicated to view a student's laptop, as the zooming capabilities of 22x according to the manual, allow for an adversary to view the screen and keyboard of most laptops in a room.



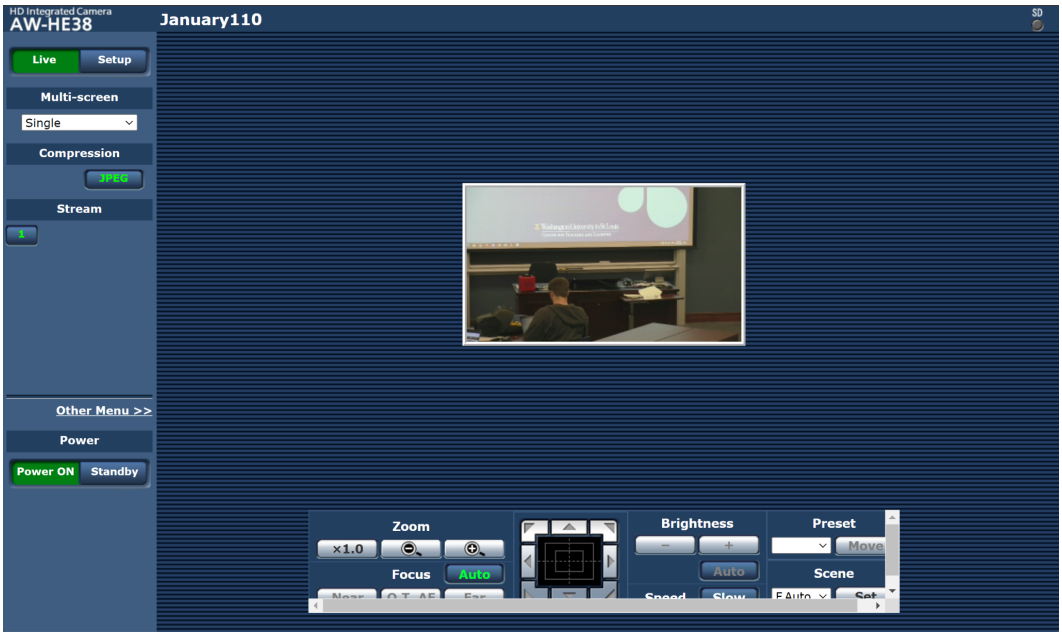
Screenshot from a Class Camera Showing Zoomed in, Clear Views of Both the Classroom Computers Keyboard and Screen. (This is us demonstrating not a real user)

Stalking

Information concerning the occupancy of classrooms across campus can be easily obtained by downloading a few frames from the video streams of each of the conference cameras available and running an OpenCV background subtraction algorithm to detect motion. See the provided code for a proof-of-concept implementation of this technology. This methodology could be further refined using specific facial detection and recognition algorithms such as buffalo or antelopev2 to track specific individuals and their movement across campus.

Denial of Service Attacks

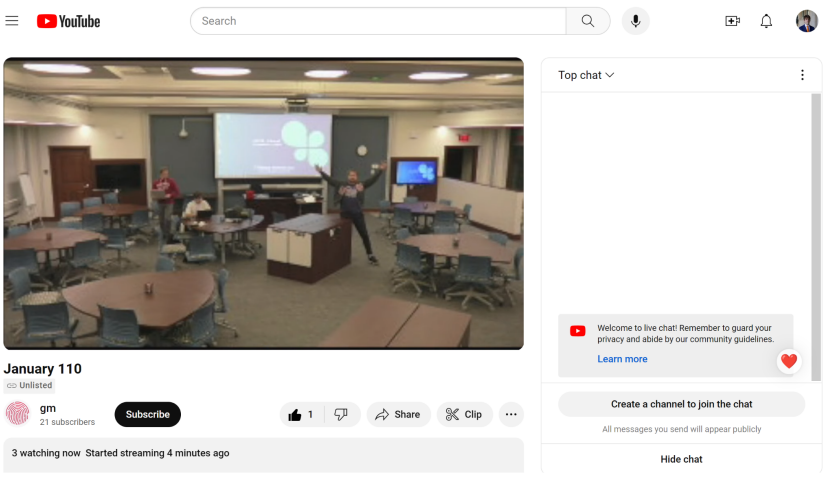
Another genuine concern is the disruption of course operations, such as Zoom recordings. As mentioned prior, the user has full control over any camera they connect to, including movement in all directions, zoom and power on/off functionality. One could easily ruin an entire Zoom recording for a lecture by mispositioning the camera or even shutting it off, and it would likely go unnoticed.



A Screenshot of the Camera Regularly used for 433S Zoom Calls with the Various Controls that Could be Used to Disrupt Operations.

Public Access

While cameras do require you to be logged into WashU wifi, there's nothing stopping a threat actor who is already logged in to forward the stream to the general public. Utilizing OBS, it's incredibly easy to pipe the camera source to a YouTube livestream. While the cameras themselves are only accessible to someone on the WashU network, this attack allows for any outsider to view into one of our classrooms. An adversary could easily set up a Raspberry Pi or something similar to run a stream indefinitely, creating a serious invasion of privacy for students.



Recommendation

Responsible Disclosure

The details of this issue, as well as a list of affected devices, was provided to the WashU IT department in a responsible disclosure. They have not expressed interest in resolving the issue through changes in configuration or device deployment.

Password Protection

All of the cameras discovered come with the capability to be password protected. WashU IT could easily utilize this functionality and add an additional layer of protection, deterring most adversaries, as there likely would be more valuable information to spend time trying to break the password to.

Isolated VLAN

To prevent unauthorized use, the conference cameras in question could be isolated on a VLAN. The presentation desktop computers or booth operator computers in each classroom which need to access the camera control page can be added to a new VLAN as well with access to the camera VLAN. The camera VLAN can also block all outbound internet traffic, potentially allowing automatic updates, to increase the security of these IOT devices.

Source Code

All of our source code can be found at the repository here:
https://wustl.box.com/s/j1zti6li42eas3z60dnzwoelo8u4yszs

References

- https://na.panasonic.com/us/audio-video-solutions/broadcast-cinema-pro-video/professional-ptz-cameras/aw-he38h-hd-professional-ptz-camera
- https://gocv.io/getting-started/
- https://obsproject.com/forum/threads/guide-how-to-stream-to-youtube.4333/