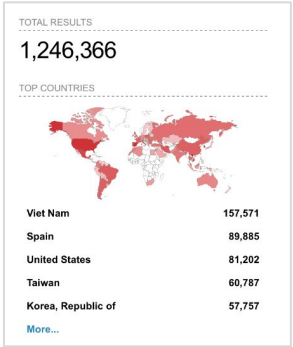
IP Camera Security
(Amcrest/Dahua)

Team 1
Abdulaziz AlMailam, Kwadwo Osafo, Karin Luna

The Problem

- IP Cameras widely popular, riddled with security issues
- Public access to private cameras notoriously common (Shodan search for “Dahua” reveals 1.2M results)
- Monopoly of camera industry (Dahua, Hikvision)
  - Vulnerabilities in one camera likely exist in various other cameras

Our goal is to raise awareness around the common cybersecurity issues within these cameras.



Our Plan

- Evaluate common vulnerabilities in a generic IP camera
  - In our example, we are using the Amcrest IP8M-2796EW-AI
- Attempt to recreate common attacks and known exploits
  - Brute force login attempt
  - DDOS attacks
  - Kaminsky DNS poisoning

Background Info - Device Factory-Default Settings

- Max Concurrent Connections: 10
- TCP Port: 37777
- UDP Port: 37778
- HTTP Port: 80
- RTSP Port: 554
- HTTPS: Off (user must upload a certificate to enable)
- Logging: Off
- Users (both have full administrative privileges):
    admin
    888888

Background Info - Device Factory-Default Settings

- Max Concurrent Connections: 10
- **TCP Port: 37777**
- UDP Port: 37778
- HTTP Port: 80
- RTSP Port: 554
- HTTPS: Off (user must upload a certificate to enable)
- **Logging: Off**
- **Users (both have full administrative privileges):**
  admin
  888888

Background Info - Device Factory-Default Settings

- **Logging: Off**
  - No evidence of an attack as no logs exist
  - Even if enabled, the logging system is often broken and can't properly open a file to write to (most cameras don't have local storage)
- **TCP Port: 37777** - Not rate-limited! More in a bit…
- **Users (both have full administrative privileges):**
  admin - default user, username/password can be changed
  888888 - hidden user intended for mouse-keyboard interaction, but actually also available remote login, only the password can be changed

Our Process - TCP Port (background)

- Authentication requests (and all other requests) to this port NOT rate limited
- Notable vulnerabilities:
  - **CVE-2013-6117:** authentication can be outright bypassed
  - **CVE-2017-6432:** Dahua DVR protocol is an unencrypted, binary protocol and is vulnerable to Man-in-Middle attacks
- TCP Port (37777) is a common target for many attacks
  - Dahua has since patched the known vulnerabilities, but more are being uncovered and they all tie back to port 37777

Our Process - TCP Port

- Authentication requests (and all other requests) to this port NOT rate limited
- **Still vulnerable to a brute force attack**
- Would be much harder if you also had to guess the username (we known that **admin** can be renamed)

  User **888888** already exists, just guess the password!

Our Process - TCP Port (cont.)

- CVE-2020-5736 - Null pointer reference:
  - Create a special packet with a null/zero byte in a specific location and send the packet over to TCP port 37777
  - Leads to a null pointer exception, causing the device to crash

- Camera expects the input credentials to authenticate to be valid, but they’re actually NULL and it crashes application due to the NullPointerException

Our Process - TCP Port (cont.)

- CVE-2020-5735 - Stack-based buffer overflow:
  - Create a special packet with a very large value in the *Protocol* field
  - Invoke the device's DDNS testing logic using command 0x62 and subcommand 0x04
  - Send the packet over to TCP port 37777
  - On the camera, memcpy() will write beyond the bounds of a stack buffer, causing the device to crash
- Can also be used to run shellcode but much harder since the payloads must be very, very small

Our Process - Javascript Web Authentication

- Javascript authentication script (/RPC2)
  - Gets user input, sanitizes it locally, then creates an MD5 hash and submits the login request
  - If valid, server creates a new session ID and returns it
- Just like with the TCP port attacks, you can try to guess Session IDs on until you succeed
  - If the camera is connected to a Digital Video Recorder (DVR), you are guaranteed that one user is always logged in
  - The default HTTP (80) rate limit is also fairly generous
- Because of limited server-side sanitization (mostly local), also vulnerable to injection attacks

Our Process - Timestamp Modification

- CVE-2022-30564 - Timestamp modification:
  - Create a special HTTP request to update the device time
  - API handling time updates doesn’t perform any authentication/verification and simply processes the request
- Can lead cameras to overwrite previous recordings
- Can mess up stored feeds due to a mismatch between recording times and actual timestamps viewable on recordings

Our Results / Interesting Insights

- Manufacturer’s of these cameras have caught up and now:
  - Use MD5 hashes of passwords
  - Allow per-user access control permissions
  - Allow firewall access control policies
  - Configurable port rate-limits (port 37777 is still NOT rate limited)
  - Flag usage of invalid credentials and inputs like session IDs (available only on specific products)

What We Learned From Our Results

- IP cameras are notoriously insecure
- Widespread nature of cameras makes it hard to update, hence vulnerabilities persist even after exposure
- Found vulnerabilities are fixed once reported
- Importance of cybersecurity awareness when using products
  - Rate-limiting
  - Sanitize all requests server-side before processing them
  - Authenticating all requests (re. timestamp modification)
  - Make all users subject to modification (888888)

Questions?