ARLO: I'M WATCHING YOU

Rédigé par Thomas Jeunet - 08/03/2024 - dans Reverse-engineering

The consumer-focused Pwn2Own competition returned in Toronto in 2023 with the "SOHO smashup" category, but also added cameras under a new "Surveillance Systems" category. While we already had success with the Wyze Cam v3 and Synology BC500 in this category, other targets were also looked at. Therefore, this blog post aims at bootstrapping vulnerability research on Arlo cameras.

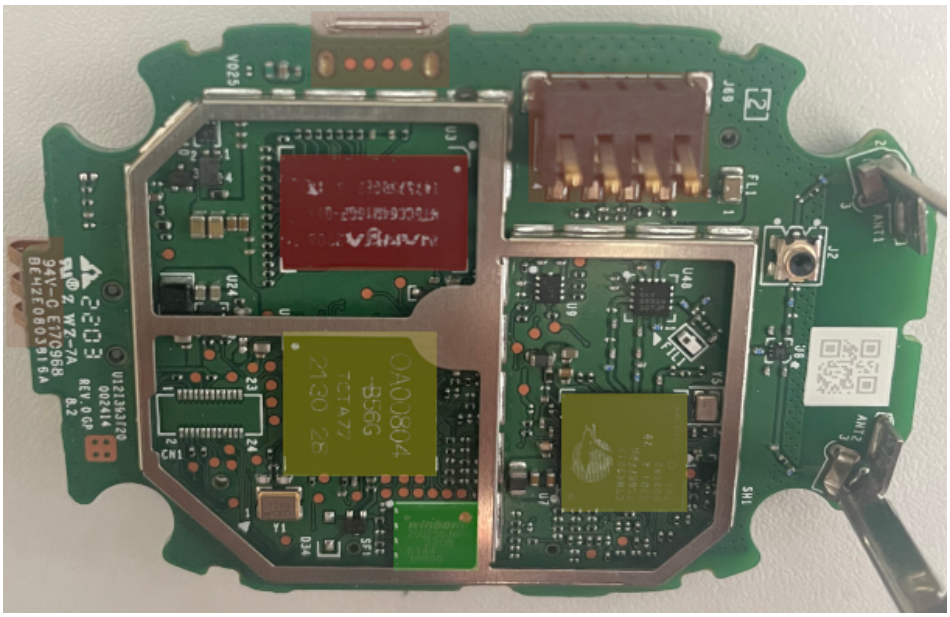
The camera comes with a USB charging cable and a battery. When installing the battery (following the user manual), a Micro-USB female port is found when removing the camera housing. However no device appears on the host when plugging a USB cable.

The device is configured using a QR code including SSID and password that the camera parses in order to connect to Wi-Fi networks. Once connected, all the traffic goes to Arlo cloud through TLS connections. Apart from NTP, only one request is performed without encryption: a HEAD HTTP request to `http://updates.arlo.com/arlo/fw/fw_deployed/production/updaterules/FB1001_UpdateRules.json`. This URI describes update binaries that can be downloaded (with paths relative to `http://updates.arlo.com/arlo/fw/fw_deployed/production/`), but the files are encrypted.

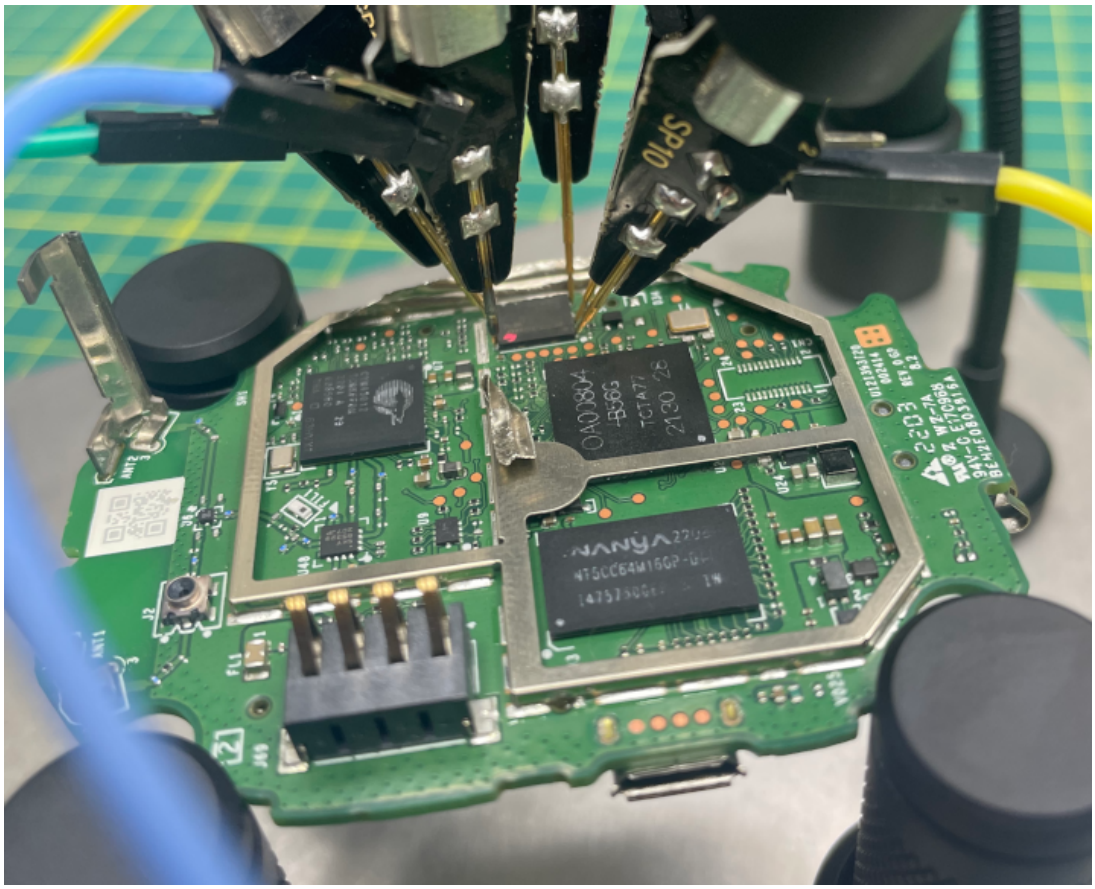
A mobile application (iOS and Android) manages the newly set up camera, but all traffic also goes to Arlo cloud and no direct connection between the phone and the camera could be identified.

HARDWARE EXPLORATION

Having no shell to interact with the system, neither clear text firmware to analyze, we have to resort to memory dumping to have anything to work with. This model internally consists of 2 PCBs, with shields hiding the chipsets. After removing them, it is possible to identify the chipsets manufacturers and models. The SoC is a Cypress (now Infineon) CYW43012, assisted with a OA00804-B56G video processor (the closest found documentation is for the OA00805-B56G-Z), using Nanya NT5CC64M16GP-DI RAM and Winbond W25Q256JW_DTR Serial Flash memory. The device is either powered by the power cable (connector on the left), battery (connector at the top right) or Micro-USB (top). The teardown process of a relatively similar model is also detail by iFixit.



Some pins of the Serial Flash memory are not easily accessible because of the shield, but it is easily bent or removed. Also, the `W25Q256JW_DTR` package is of WSON type, and the SOIC clip does not fit on the package, so PCBite are required to access the SPI pins.



Trying to dump the memory using 1.8V SPI, as stated in the datasheet, yield no response. However the dump succeeded using a 3.3V SPI adapter:

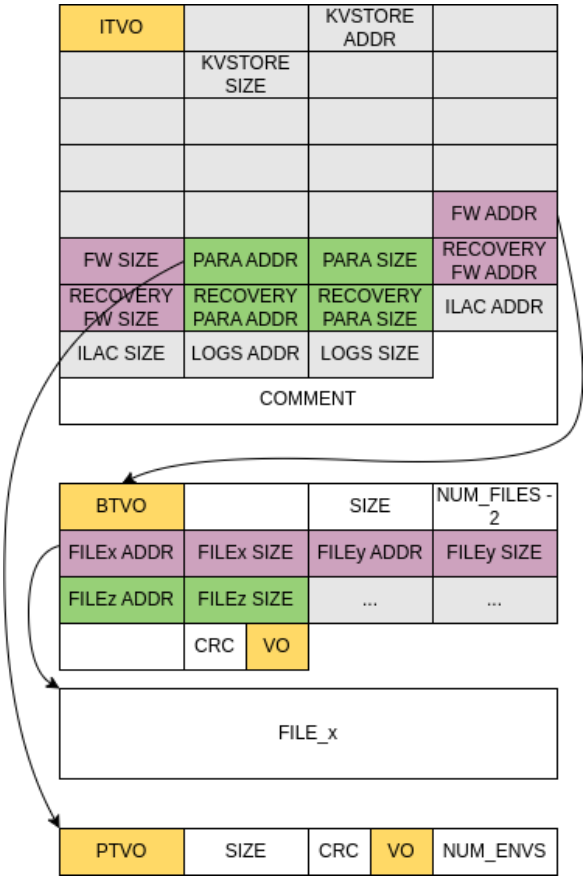
```
$ sudo flashrom -p buspirate_spi:dev=/dev/ttyUSB0,spispeed=2Mhz -c W25Q256JW_DTR -r dump.bin
[...]
Found Winbond flash chip "W25Q256JW_DTR" (32768 kB, SPI) on buspirate_spi.
Reading flash... done.
```

FIRMWARE ANATOMY

Unlike the update binary, the firmware is stored in clear text and strings in the file confirms this is code running on the SoC. However, the binary dump is not loaded as-is in memory, and we need to understand the binary format before analyzing code.

```
$ strings dump.bin
[...]
Easycam Liveview
OvSelectApp
Unable to write OvAppSelect
CamModePerm
CamModeTemp
sf_detect %s: %x(%x)
sflash found: %s
sf detect failed, readid: %x
error: g_sflash_fun_init.erase is NULL
error: g_sflash_fun_init.program is NULL
error: g_sflash_fun_init.read is NULL
error: data length is forbidden as 0 when read serial flash
EEROR:set qe bit error:%d
flash id not match, read:0x%x(0x%x)
ERROR:hardware haed magic num error,read:%x should be 0x4f565449
ERROR:para haeder magic num error,read:%x should be 0x4f565450
ERROR:fw haeder magic num error,read:%x should be 0x4f565442
para header error
para crc error
magic3 is broken, need do partitioin recover
[...]
```

The error strings indicate the magic values: `0x4f565449/"ITVO"` for hardware head, `0x4f565442/"BTVO"` for `fw header` and `0x4f565450/"PTVO"` for `para header` (the typos are corrected for readability).




```
$ hexdump -C dump.bin
00000000  49 54 56 4f 11 00 00 00  00 d0 ee 01 12 00 00 00  |ITV
O............|
00000010  00 40 19 10 00 28 00 00  00 00 00 00 00 00 0a 10  |.@...
(.........|
00000020  00 00 00 00 00 00 00 00  db 02 00 00 10 01 41 00
|..............A.|
00000030  03 00 00 00 00 44 00 00  00 00 00 00 00 00 0a 10  |.....
D.........|
00000040  00 00 00 00 00 00 00 00  03 02 00 00 00 10 00 00
|................|
00000050  00 00 f4 00 00 10 f4 00  00 20 03 00 00 30 f7 00  |.........
...0..|
00000060  00 00 f4 00 00 30 eb 01  00 20 03 00 00 50 ee 01  |.....0...
...P..|
00000070  00 78 00 00 00 00 ef 01  00 b0 04 00 6a 69 00 00  |.
x.........ji..|
00000080  61 72 6c 6f 5f 63 61 72  61 63 61 72 61 00 00 00  |arlo_carac
ara...|
00000090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
|................|
*
00001000  42 54 56 4f 00 00 00 00  a0 5e c1 00 26 00 00 00  |BTVO.....
^..&...|
[...]
00c16f00  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff
|................|
*
00f41000  50 54 56 4f 00 10 03 00  e1 7f 56 4f 06 00 00 00  |PTVO......
VO....|
00f41010  00 80 00 00 00 00 00 00  00 00 00 00 11 49 57 4f
|.............IWO|
00f41020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
|................|
*
00f42000  59 54 56 4f 94 0f 00 00  b5 00 26 4c f1 cd 56 4f  |YTVO......
&L..VO|
00f42010  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff
|................|
*
00f43000  77 69 66 69 5f 73 73 69  64 3d 74 65 73 74 61 70  |wifi_ssid=
testap|
00f43010  70 0a 77 69 66 69 5f 6b  65 79 3d 61 62 63 64 65  |p.wifi_key
=abcde|
```

The CRC implementation is CRC16 CCITT.

ITVO

The `ITVO` header mainly holds information about area on the Flash memory, with address and size as `uint32_t`:

1. `kvstore` apparently storing sensitive information, but this has not been investigated.
2. `fw` storing binaries executed on the CPU or peripheral CPUs, starting with a `BTVO` header.
3. `para` storing configurations for the device, starting with a `PTVO` header.
4. `cali` apparently storing encrypted information, using key from the `kvstore`, starting with a `CALI` header.
5. `logs` notably where logs are written when the device halts or reboots.

PTVO

The `PTVO` header describes configuration area. It starts with a `YTVO` header aligned on `0x1000` memory pages, holding notably its size and an ID. It is then followed by the configuration itself, also aligned on `0x1000` memory pages, and stored in text format as key/value pairs separated by `=` and followed by `\n`.

BTVO

The `BTVO` header describes the firmware binaries, loaded as-is in memory to be executed by the CPU or loaded in a second step by the peripheral requiring it. This device is based on an RTOS and the different modes of operation (setup, update, etc.) are not different tasks on the same OS but rather distinct full images loaded in memory when needed. These full images correspond to different "files" in the header and the index is used to address a specific mode of operation. Mode switching is performed by loading the file at its load address in memory and restarting. A shell is present in some files, so it is easy to determine the mapping between IDs and modes:

```
int __fastcall cmd_fw(int argc, char **argv) {
  int fw_id; // r0
```

```
  if ( argc <= 1 )
    goto ERROR;
  if ( !strcmp(argv[1], "uploading") ) {
    fw_id = 2;
RELOAD_FW:
    reload_fw(fw_id);
    return 0;
  }
  if ( !strcmp(argv[1], "liveview") ) {
    fw_id = 4;
    goto RELOAD_FW;
  }
  if ( !strcmp(argv[1], "arlogw") )
    goto RELOAD_ARLOGW;
  if ( !strcmp(argv[1], "calfw") )  {
    fw_id = 14;
    goto RELOAD_FW;
  }
  if ( !strcmp(argv[1], "upgrade") ) {
    fw_id = 9;
    goto RELOAD_FW;
  }
  if ( !strcmp(argv[1], "setup") ) {
    fw_id = 12;
    goto RELOAD_FW;
  }
  if ( !strcmp(argv[1], "setupqr") ) {
    fw_id = 15;
    goto RELOAD_FW;
  }
  if ( !strcmp(argv[1], "pjsip") ) {
RELOAD_ARLOGW:
    fw_id = 36;
    goto RELOAD_FW;
  }
ERROR:
  printf("usage: fw [ uploading | liveview | arlogw | calfw | setup | setupqr | upgrade ]\n");
  return 0;
}
```

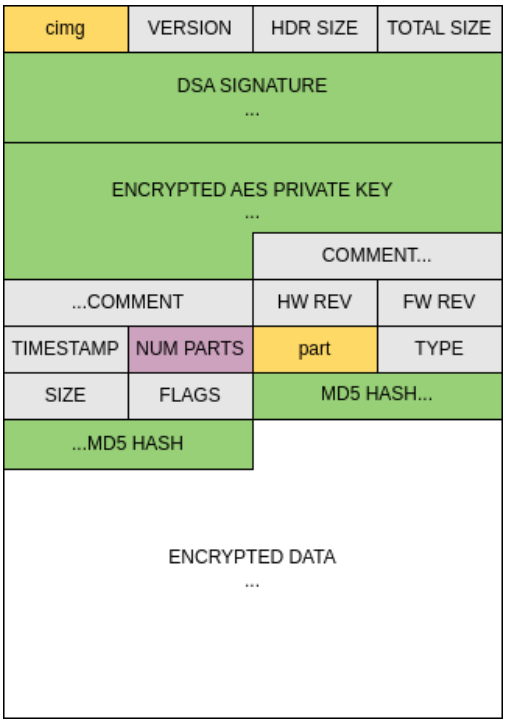
The `setupqr` mode handles device setup using the QR code to obtain Wi-Fi credentials. The `arlogw` mode appears to be the nominal mode when the device is properly setup and may have absorbed functionalities over time as it embeds `pjsip` COTS for audio/video streaming. Some modes use two files, in this case the first one is in charge of loading the second one, so mostly similar but slightly different file loading routines are present in most binaries.

The `upgrade` mode handles downloading, verifying and writing the update to the Flash memory. The analysis of this file was sufficient to understand the (encrypted) update format and process.

UPDATES

FETCHING AND PARSING

As stated previously, the update process first retrieves a JSON file. However, the URL previously identified does not match updates for this model. The proper update URL is `https://updates.arlo.com/arlo/fw/fw_deployed/production/updaterules/VMC4041P_UpdateRules.json` (`VMC4041P` is the model ID of the Arlo Pro4). This JSON document contains the binary path, its version and date, and the MD5 hash of the binary.



The downloaded binary follows this format:

1. Magic bytes to identify the format, `cimg`.
2. The format version, `1` in this case but `2` exists.
3. The header and total size of the binary, header size is `0x178` for version `1`.
4. The DSA signature of the package, (`s1` , `s2`) in ASN.1.
5. An encrypted AES key.
6. A comment string, empty in all observed updates.
7. A hardware and firmware revision, both `0` in all observed updates.
8. The binary timestamp in epoch format.
9. The number of following parts.

The part format is as follows:

1. Magic bytes, `part`.
2. A type, either `1` (`rootfs`), 2 (`generic`), `3` (`kernel`).
3. The size of the data.
4. Flags describing how the data is stored, either `0` (stored), `1` (compressed), `2` (encrypted) or `3` (compressed+encrypted).
5. A MD5 hash of the data. As this camera model does not check the hash validity, it is not clear what the input data exactly is.

The DSA signature is performed on the whole file with the signature removed.

DECRYPTION

The AES key is decrypted using an RSA private key that is embedded in the loaded file. The RSA private key is encrypted using a XOR key of 4 bytes, also in the loaded file.

```
do {
    rsa_priv_key[i] = ~xor_key[i & 3] ^ rsa_priv_key_obf[i];
    ++i;
} while ( i != 1193 );
bio = BIO_new_mem_buf(rsa_priv_key, 1193);
```

The clear text AES key is provided to `EVP_BytesToKey` along with a fixed salt from the loaded file in order to derive the actual key with the IV.

As OpenSSL is used in a version before 3.0, function identification is broadly possible using `ERR_put_error(int lib, int func, int reason, const char *file, int line)` : once this latter function is identified (the `file` parameter is helpful for this), it is possible to rename functions calling it using the `lib` and `func` parameters. The `openssl_err.py` script in the GitHub repository parses the `.h` file in the `openssl` include directory to generate a Python dict. The latter is then used in the `ossl_rename_ida.py` script that uses the Bip plugin to rename the functions.

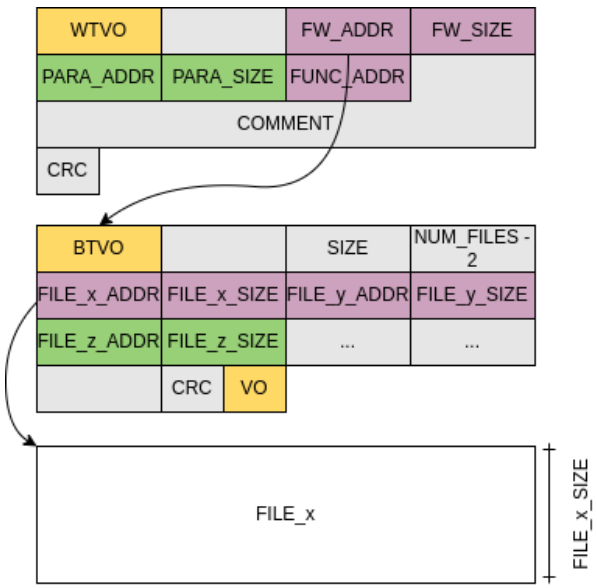
Regarding missing names, it is possible to use `ossl_assert` parameters as they include the file path and line number. For instance, `ossl_assert_int("/home/jenkins/agent/workspace/_VMC4041P_1.080.20.0_483_0f3935c/omnivision/make/../share/network/openssl/crypto/evp/digest.c 271, "ctx->digest->md_size <= EVP_MAX_MD_SIZE")` must match EVP_DigestFinal_ex.

Other function, such as `EVP_aes_256_cbc` and `EVP_sha1`, can be identified by using cross-references to NID definitions to identify the algorithm, for instance `#define NID_aes_256_cbc 427` and `#define NID_sha1 64`:

```
DDR_CODE:201244C0 aes_256_cbc     DCD 427, 16, 32, 16, 20482
DDR_CODE:201244D4                 DCD aes_init_key
DDR_CODE:201244D8                 DCD sub_200B3BF0
[...]
DDR_CODE:201245BC sha1            DCD 0x40, 0x41, 0x14, 0xC
DDR_CODE:201245CC                 DCD sub_200B3D58
DDR_CODE:201245D0                 DCD sub_200B3DB0
```

WTVO

Once decrypted, the content of a `part` is a `WTVO` header using the same logic as other headers:



The `WTVO` header holds information about the update content, similarly to what has been previously identified:

1. `fw` is actually a full `BTVO` area that is copied as-is on the Flash memory.
2. Eventually `para` would be a `PTVO` area copied as-is on the Flash memory but no updates were found to contain `para`.
3. A comment string.
4. A CRC to verify the header.

KEY REUSE

The URL for downloading updates is built using `snprintf(dst, 256, "%s/updaterules/%s_UpdateRules.json", update_url, base_model_id)` where `update_url` is one of:

- https://arloupdates.arlo.com/arlo/fw/fw_deployed/dev
- https://arloupdates.arlo.com/arlo/fw/fw_deployed/qa
- https://arloupdates.arlo.com/arlo/fw/fw_deployed/goldenft
- https://updates.arlo.com/arlo/fw/fw_deployed/staging
- https://updates.arlo.com/arlo/fw/fw_deployed/fieldtrial
- https://updates.arlo.com/arlo/fw/fw_deployed/production

The `base_model_id` is VMC4041P for this specific model. The list of possible `base_model_id` can be found on the Arlo support website.

URLs are accessible for every environments and all models, meaning we can download every currently available update binaries! And it seems the RSA private key is associated with the version 1 of the `cimg` format rather than specific for each model, hence we can decrypt many more firmware:

```
- Cameras
  - Arlo Wire-Free
    - NTGR_1.5.295_WiFi_upgrade.bin.enc                    cimg: version: 1 size: 0x18be78, date: 2021-08-31T21:07:27 / part: kern
el, encrypted / OVTW: (v1.5.295)
  - Arlo Pro
    - VMC4030_1.092.1.0_9_120d8b7.bin.enc                  cimg: version: 1 size: 0x3076c8, date: 2021-08-25T07:36:52 / part: kern
el, encrypted
  - Arlo Pro 2
    - VMC4030P_1.125.17.1_11_de8490d.bin.enc               cimg: version: 1 size: 0x387788, date: 2021-09-04T01:39:52 / part: kern
el, encrypted
  - Arlo Pro 3
```

```
    - http://updates.arlo.com/arlo/fw/fw_deployed/production/updaterules/VMC4040P_UpdateRules.json HTTP 404
  - Arlo Pro 4 (VMC4050P)
    - http://updates.arlo.com/arlo/fw/fw_deployed/production/updaterules/VMC4050P_UpdateRules.json HTTP 404
  - Arlo Pro 4 (VMC4041P)
    - VMC4041P_1.080.20.1_23_d50a19d.bin.enc                  cimg: version: 1 size: 0xc37248, date: 2023-08-08T00:27:03 / part: kern
el, encrypted
  - Arlo Pro 5S
    - http://updates.arlo.com/arlo/fw/fw_deployed/production/updaterules/VMC4060P_UpdateRules.json HTTP 404
  - Arlo Ultra / Ultra 2
    - VMC5040_1.070.52.1_35_1bdb65f.bin.enc                   cimg: version: 1 size: 0x656768, date: 2021-09-04T01:44:14 / part: kern
el, encrypted
  - Arlo Essential Outdoor 2nd Gen (VMC2050)
    - VMC2050-1.2.2_512_9955b13.prod-pd.enc                   cimg: version: 2 size: 0x6e31c8, date: 2023-10-24T01:10:44, (VMC2050) /
part: kernel, encrypted
    - VMC2050-1.2.2_512_9955b13.prod-pp.enc                   cimg: version: 2 size: 0x6e31c8, date: 2023-10-24T02:22:05, (VMC2050) /
part: kernel, encrypted
  - Arlo Essential Outdoor 2nd Gen (VMC3050)
    - VMC3050-1.2.2_512_9955b13.prod-pd.enc                   cimg: version: 2 size: 0x6e31c8, date: 2023-10-24T01:14:26, (VMC3050) /
part: kernel, encrypted
    - VMC3050-1.2.2_512_9955b13.prod-pp.enc                   cimg: version: 2 size: 0x6e31c8, date: 2023-10-24T02:21:55, (VMC3050) /
part: kernel, encrypted
  - Arlo Essential XL Outdoor 2nd Gen (VMC2052)
    - http://updates.arlo.com/arlo/fw/fw_deployed/production/updaterules/VMC2052_UpdateRules.json HTTP 404
  - Arlo Essential XL Outdoor 2nd Gen (VMC3052)
    - http://updates.arlo.com/arlo/fw/fw_deployed/production/updaterules/VMC3052_UpdateRules.json HTTP 404
  - Arlo Essential
    - VMC2020_1.090.32.1_55_07e3402.bin.enc                   cimg: version: 1 size: 0xda01e8, date: 2023-08-03T02:24:07 / part: kern
el, encrypted / BTVO: long
  - Arlo Essential Spotlight
    - VMC2030_1.090.32.1_55_07e3402.bin.enc                   cimg: version: 1 size: 0xda01e8, date: 2023-08-03T03:21:49 / part: kern
el, encrypted / BTVO: long
  - Arlo Essential XL Spotlight
    - http://updates.arlo.com/arlo/fw/fw_deployed/production/updaterules/VMC2032_UpdateRules.json HTTP 404
  - Arlo Q
    - VMC3040-1.7.4_5517.prod.upgrade                         cimg: version: 1 size: 0x2873f78, date: 2016-04-02T02:06:03 / part: roo
tfs, compressed+encrypted, zlib, unknown magic b'\x00\x90\x0f\xe1' / part: generic, encrypted, SquashFS
    - VMC3040-1.13.0.0_95_a58d08a_db3500e.prod.upgrade        cimg: version: 1 size: 0x2b320a8, date: 2023-03-14T06:47:10 / part: roo
tfs, compressed+encrypted, zlib, unknown magic b'\x00\x90\x0f\xe1' / part: generic, encrypted, SquashFS
  - Arlo Q+
    - http://updates.arlo.com/arlo/fw/fw_deployed/production/updaterules/VMC3040S_UpdateRules.json HTTP 404
  - Arlo Essential Indoor 2nd Gen (VMC2060)
    - VMC2060-1.2.2_515_b1ea04e.prod-pd.enc                   cimg: version: 2 size: 0x6a41c8, date: 2023-11-07T18:02:16, (VMC2060) /
part: kernel, encrypted
    - VMC2060-1.2.2_515_b1ea04e.prod-pp.enc                   cimg: version: 2 size: 0x6a41c8, date: 2023-11-07T18:33:54, (VMC2060) /
part: kernel, encrypted
  - Arlo Essential Indoor 2nd Gen (VMC3060)
    - VMC3060-1.2.2_515_b1ea04e.prod-pd.enc                   cimg: version: 2 size: 0x6a41c8, date: 2023-11-07T18:05:58, (VMC3060) /
part: kernel, encrypted
    - VMC3060-1.2.2_515_b1ea04e.prod-pp.enc                   cimg: version: 2 size: 0x6a41c8, date: 2023-11-07T18:34:01, (VMC3060) /
part: kernel, encrypted
  - Arlo Essential Indoor
    - VMC2040_OTA-1.19.0.0_1182_900c0a4_d2776d0.prod.enc      cimg: version: 1 size: 0x11ccc88, date: 2023-09-26T17:27:15 / part: ker
nel, encrypted, SquashFS
    - VMC2040_OTA-1.19.0.0_1182_900c0a4_d2776d0.prod.sig.enc  cimg: version: 1 size: 0x11cce88, date: 2023-09-26T17:27:15 / part: ker
nel, encrypted, SquashFS
  - Arlo Baby
    - ABC1000-1.14.0.0_124_a58d08a_b7792aa.prod.upgrade       cimg: version: 1 size: 0x34e2148, date: 2023-03-14T06:27:36 / part: roo
tfs, compressed+encrypted, zlib, unknown magic b'\xf6)\x00\xeb' / part: generic, encrypted, SquashFS
- Base_Station_SmartHub
  - Arlo Base Station (VMB3010)
    - VMB3010-1.21.0.2_1540_8a519c8.prod.chk.enc              cimg: version: 1 size: 0xcfc1d8, date: 2023-05-20T08:43:18 / part: kern
el, encrypted, NETGEARHDR0
  - Arlo Base Station (VMB3500)
    - VMB3500-1.21.0.2_1296_8a519c8.prod.upgrade              cimg: version: 1 size: 0xaed9e8, date: 2023-05-20T08:57:59 / part: root
fs, compressed+encrypted, zlib, unknown magic b"'\x05\x19V" / part: generic, encrypted, UBI
  - Arlo Pro Base Station (VMB4000)
    - VMB4000-1.21.1.0_4302_9611cf3.prod.chk.enc              cimg: version: 1 size: 0xfd01d8, date: 2023-06-16T08:26:17 / part: kern
el, encrypted, NETGEARHDR0
  - Arlo Pro Base Station (VMB4500)
    - VMB4500-1.21.1.0_4167_9611cf3.prod.upgrade              cimg: version: 1 size: 0xd0dbe8, date: 2023-06-16T08:05:18 / part: root
fs, compressed+encrypted, zlib, unknown magic b"'\x05\x19V" / part: generic, encrypted, UBI
  - Arlo Pro 3 SmartHub
    - VMB4540-1.21.1.0_1399_9611cf3.prod.upgrade              cimg: version: 1 size: 0xe60248, date: 2023-06-16T09:39:26 / part: root
fs, compressed+encrypted, zlib, unknown magic b"'\x05\x19V" / part: generic, encrypted, UBI
  - Arlo Ultra Smarthub
    - VMB5000-1.21.1.0_1431_9611cf3.prod.upgrade.enc          cimg: version: 1 size: 0x13ad7a8, date: 2023-06-16T12:38:58 / part: ker
nel, encrypted, pkgtb
- Doorbells
  - Arlo Video Doorbell 2nd Gen (AVD3001)
    - AVD3001-1.2.1_435_abb9732.prod-pd.enc                   cimg: version: 2 size: 0x65a1b8, date: 2023-10-18T23:54:56, (AVD3001) /
part: kernel, encrypted
    - AVD3001-1.2.1_435_abb9732.prod-pp.enc                   cimg: version: 2 size: 0x65a1b8, date: 2023-10-19T02:13:43, (AVD3001) /
```

```
part: kernel, encrypted
  - Arlo Video Doorbell 2nd Gen (AVD4001)
    - AVD4001-1.2.1_435_abb9732.prod-pd.enc                    cimg: version: 2 size: 0x6481b8, date: 2023-10-19T00:00:14, (AVD4001) /
part: kernel, encrypted
    - AVD4001-1.2.1_435_abb9732.prod-pp.enc                    cimg: version: 2 size: 0x6481b8, date: 2023-10-19T01:37:06, (AVD4001) /
part: kernel, encrypted
  - Arlo Video Doorbell - Wired
    - AVD1001_OTA-1.19.0.0_3_098d492_3567407.prod.enc          cimg: version: 1 size: 0x1081878, date: 2023-09-13T20:01:39 / part: ker
nel, encrypted, SquashFS
    - AVD1001_OTA-1.19.0.0_3_098d492_3567407.prod.sig.enc      cimg: version: 1 size: 0x1081a78, date: 2023-09-13T20:01:39 / part: ker
nel, encrypted, SquashFS
  - Arlo Video Doorbell Wire-Free
    - AVD2001_OTA1-1.8.0.0_4_4a16fed_758744c.prod.enc          cimg: version: 1 size: 0x1946188, date: 2023-10-27T16:00:51 / part: ker
nel, encrypted, SquashFS
    - AVD2001_OTA2-1.8.0.0_4_4a16fed_758744c.prod.enc          cimg: version: 1 size: 0x157c368, date: 2023-10-27T16:00:51 / part: ker
nel, encrypted, SquashFS
    - AVD2001_OTA2-1.8.0.0_4_4a16fed_758744c.prod.sig.enc      cimg: version: 1 size: 0x157c8f8, date: 2023-10-27T16:00:52 / part: ker
nel, encrypted, SquashFS
  - Arlo Audio Doorbell
    - AAD1001_1.2.0.0_320_401_DV1.bin                          unexpected container b'MMM\x00'
    - AAD1001_1.2.0.0_320_401_DV1.bin.enc                      cimg: version: 1 size: 0x7e248, date: 2021-08-18T00:16:46 / part: kerne
l, encrypted, MMM
    - AAD1001_1.2.0.0_320_401_DV2.bin                          unexpected container b'MMM\x00'
    - AAD1001_1.2.0.0_320_401_DV2.bin.enc                      cimg: version: 1 size: 0x7e5d8, date: 2021-08-18T00:18:13 / part: kerne
l, encrypted, MMM
  - Arlo Chime V2
    - AC2001_OTA-1.1.0.0_314_d36790b.prod.enc                  cimg: version: 1 size: 0xf63f8, date: 2023-05-03T18:09:53 / part: kerne
l, encrypted, MMM
    - AC2001_OTA-1.1.0.0_314_d36790b.prod.sig.enc              cimg: version: 1 size: 0xf63f8, date: 2023-05-03T18:09:53 / part: kerne
l, encrypted, MMM
  - Arlo Chime
    - AC1001_1.2.0.0_320_392_DV1.bin                           unexpected container b'MMM\x00'
    - AC1001_1.2.0.0_320_392_DV1.bin.enc                       cimg: version: 1 size: 0x7abe8, date: 2021-08-18T00:16:37 / part: kerne
l, encrypted, MMM
    - AC1001_1.2.0.0_320_392_DV2.bin                           unexpected container b'MMM\x00'
    - AC1001_1.2.0.0_320_392_DV2.bin.enc                       cimg: version: 1 size: 0x7af48, date: 2021-08-18T00:17:55 / part: kerne
l, encrypted, MMM
- Floodlights
  - Arlo Pro 3 Floodlight
    - FB1001_1.080.28.0_12_590aec8.bin.enc                     cimg: version: 1 size: 0xc3e388, date: 2023-06-15T04:00:28 / part: kern
el, encrypted
  - Arlo Security Light
    - http://updates.arlo.com/arlo/fw/fw_deployed/production/updaterules/ABB1000_UpdateRules.json HTTP 404
- Arlo_Home_Security_System
  - Arlo Keypad Hub
    - http://updates.arlo.com/arlo/fw/fw_deployed/production/binaries/LBB1001 HTTP 404
    - http://updates.arlo.com/arlo/fw/fw_deployed/production/binaries/LBB1001 HTTP 404
    - SH1001-1.10.8_c1a07e9.enc                                cimg: version: 2 size: 0xef3d28, date: 2023-10-27T08:09:31, (SH1001) /
part: kernel, encrypted
  - Arlo Siren
    - SLB1001-1.0.144_921dfbf.enc                              cimg: version: 2 size: 0x641c8, date: 2023-09-15T21:02:16, (SLB1001) /
part: kernel, encrypted
    - SLB1001-1.0.144_921dfbf.devsig.enc                       cimg: version: 2 size: 0x64278, date: 2023-09-15T21:02:16, (SLB1001) /
part: kernel, encrypted
    - SLB1001-1.0.144_921dfbf.sig.enc                          cimg: version: 2 size: 0x64278, date: 2023-09-15T21:02:16, (SLB1001) /
part: kernel, encrypted
  - Arlo All-in-One Sensor
    - MS1001-2.1.246_e8e8e84.enc                               cimg: version: 2 size: 0x66838, date: 2023-09-20T22:16:02, (MS1001) / p
art: kernel, encrypted
    - MS1001-2.1.246_e8e8e84.sig.enc                           cimg: version: 2 size: 0x668e8, date: 2023-09-20T22:16:02, (MS1001) / p
art: kernel, encrypted
  - Arlo Safe Button
    - FG_FW-0.0.26.2-apploader-signed.gbl                      unexpected container b'\xeb\x17\xa6\x03'
    - FG_FW-0.0.26.2-application-signed.gbl                    unexpected container b'\xeb\x17\xa6\x03'
```

Some of these updates are not an RTOS but rather a Linux kernel with a rootfs, either SquashFS or UBI. Also, the key and format for `cimg` version 2 is not the same, so it was not possible to decrypt such updates.

TOOLING

The tools developed during this study are available on our GitHub repository. When known, the output includes scripts to load the mode, such as `arlogw`, directly in IDA with proper load address and the related binary loaded.

CONCLUSION

You are now ready to bootstrap your own vulnerability research, as long as you are able to find out the RSA private key by yourself, as Arlo prefers this private key not to be released. Also keep in mind that the attack surface is small, unless you know how to perform Man-in-the-Middle attacks on TLS connections!