Building a Secure Future.™

Ottawa Hackday, October 3rd, 2019: RF Hacking (not Protecting Docker Containers!)

The RF Hacking Team:

- Lead: John Mehan
- Minions:

  Anurag Garg

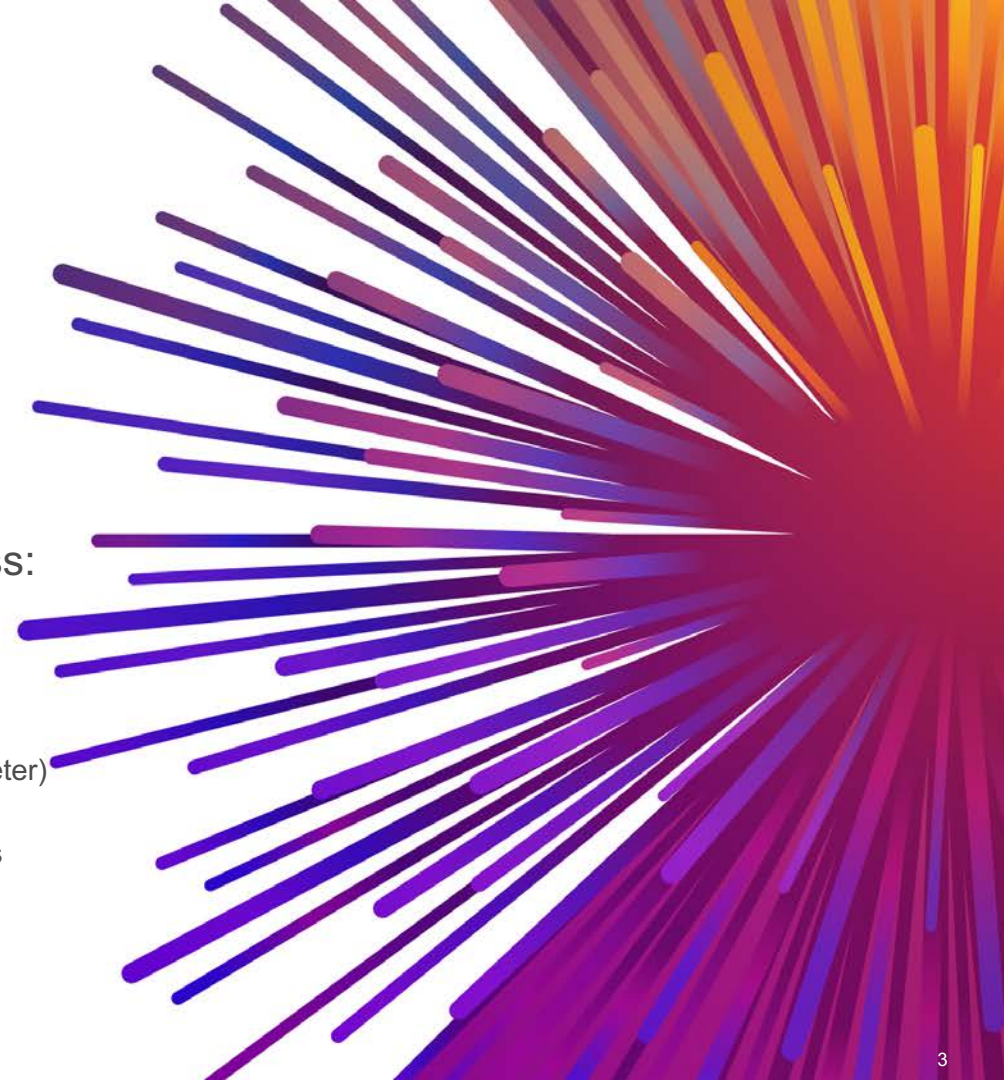
  JungAh Hong

  Mark Roberts

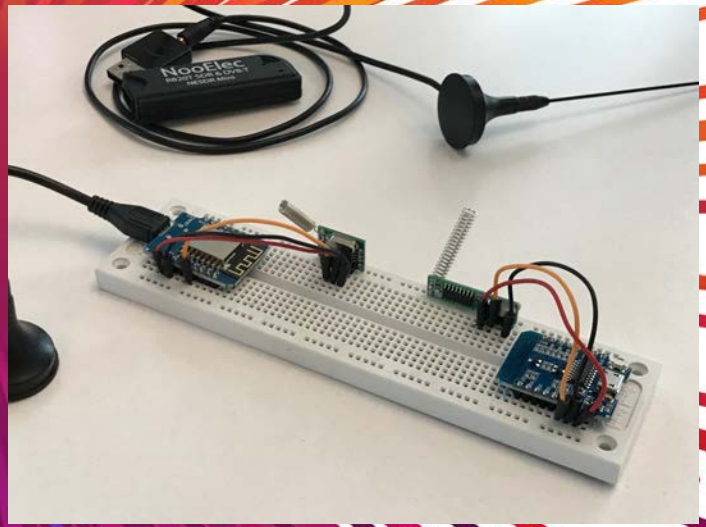
RF Hacking – Goals & Objectives

- Learn the fundamentals of SDR, and simple digital sensors and RF remotes
- We wanted to touch on the whole process:

  Detect, Analyze, Record, Transmit (Mimic/Replay), Modify/Hack
- Our high-level strategy/tactics:
  1. Start with well documented devices (Acurite thermometer)
  2. Explore unknown devices (Inkbird, Car fobs)
  3. Test our understanding by transmitting correct packets
  4. Contrive a simple hack that would be a viable prank

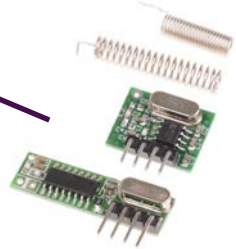


RF Hacking Hardware Tools



DVB/DAB USB SDR Receiver ($13.89)

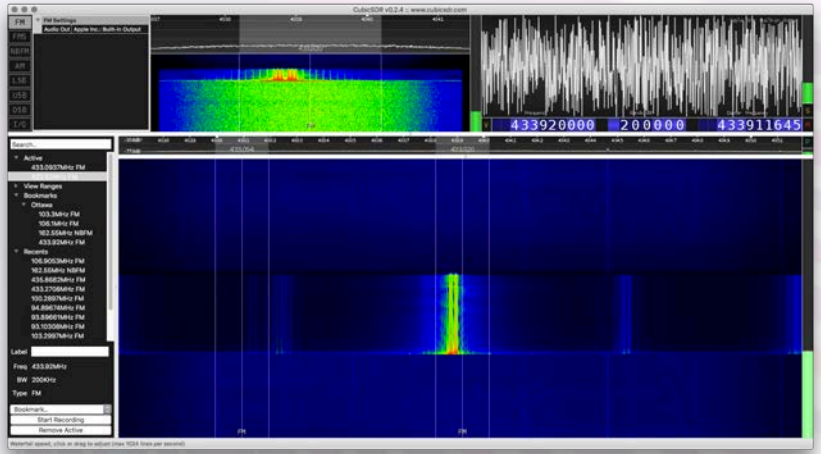
433 MHz ASK Receiver/Transmitter ($2.50)

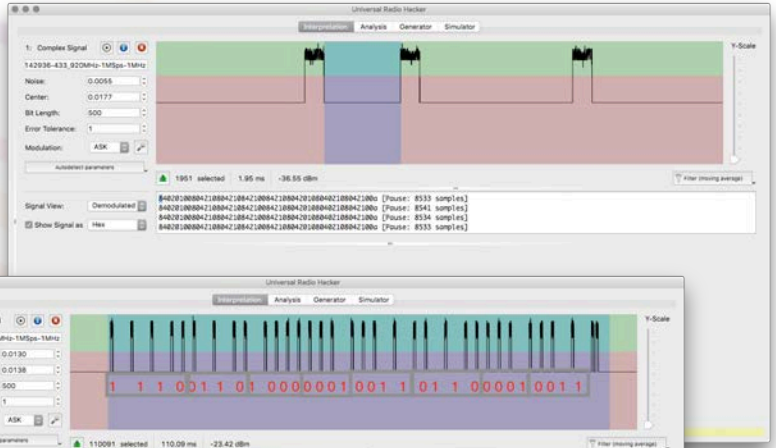


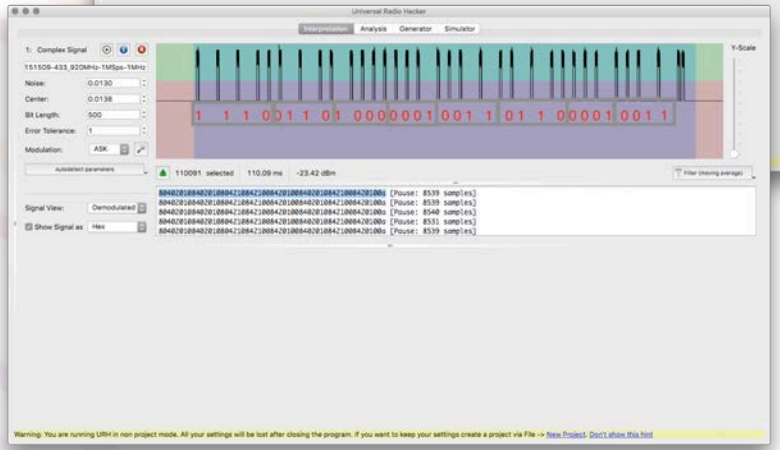
Arduino-like ESP8266 Micro ($2.81)

433MHz RF Devices to ’Test’

Software Tools







```
Detected OOK package    2019-10-03 11:49:46
Analyzing pulses...
Total count:  237,  width: 744.45 ms        (186112 S)
Pulse width distribution:
 [ 0] count:  237,  width:  508 us [496;516]    ( 127 S)
Gap width distribution:
 [ 0] count:  154,  width: 1932 us [1924;1944]  ( 483 S)
 [ 1] count:   70,  width: 3884 us [3876;3896]  ( 971 S)
 [ 2] count:    5,  width:  468 us [460;476]    ( 117 S)
 [ 3] count:    6,  width: 8520 us [8516;8532]  (2130 S)
 [ 4] count:    1,  width:  708 us [708;708]    ( 177 S)
Pulse period distribution:
 [ 0] count:  154,  width: 2440 us [2432;2456]  ( 610 S)
 [ 1] count:   70,  width: 4392 us [4380;4408]  (1098 S)
 [ 2] count:    5,  width:  972 us [972;980]    ( 243 S)
 [ 3] count:    6,  width: 9032 us [9028;9040]  (2258 S)
 [ 4] count:    1,  width: 1220 us [1220;1220]  ( 305 S)
Level estimates [high, low]:  15939,   1378
Attempting demodulation... short_width: 468, long_width: 708, reset_limit: 8536, sync_width: 0
Use a flex decoder with -X 'n=name,m=OOK_PPM,s=468,l=708,g=712,r=8536'
```

Eclectic Musings Of A Chaotic Mind

Learning a bit about LFSR Hash…

- It turned out that the Acurite thermometer used a rather sophisticated hash (for its class of device) as the checksum (Toeplitz LFSR it turned out)
- After seeing us flail around with some hacker and forum postings, JungAh grew frustrated and went away to write it from scratch using a description described in a paper
- Having the proper hash algorithm we could then take any rolling code from a received victim's transmission, create our own data and force the new values on the receiver… our prank was within reach!

```
#define MESSAGE_SIZE 32
#define OFFSET 4

uint8_t LSFR_sequence[MESSAGE_SIZE] = {0};

void calculateLSFR() {
int i;
uint8_t reg = 0x7C;
uint8_t temp_reg = 0;

for (i = 0; i < MESSAGE_SIZE; i++) {
temp_reg = reg & 0x01;
reg >>= 1;
reg |= (temp_reg << 7);

if (temp_reg) {
reg ^= 0x18;
}

LSFR_sequence[i] = reg;
//printf("%02x\n", LSFR_sequence[i]);
}
}

uint8_t combineLSFR(uint8_t len, uint8_t *data) {
uint8_t hash_reg = 0; // not 0x64
int byte_idx, bit_idx;
uint8_t byte, bit;
//printf("***COMBINE\n");

for (byte_idx = 0; byte_idx < len; byte_idx++) {
for (bit_idx = 7; bit_idx >= 0; bit_idx--) {
bit = (data[byte_idx] & (1 << bit_idx)) >> bit_idx;
if (bit) {
hash_reg ^= LSFR_sequence[byte_idx * 8 + (7 - bit_idx) + OFFSET];
//printf("[%d]: %02x\n", byte_idx * 8 + (7 - bit_idx), hash_reg);
}
bit = 0;
}
}

return hash_reg;
}

uint8_t Checksum(int length, uint8_t *buff) {
calculateLSFR();
return combineLSFR(length, buff);
}
```

Developing the Hash Code

Hash Code

- Generated by multiplying the message bits with the byte sequence generated by a linear feedback shift register (LFSR)
- The first part is to generate a sequence of bytes using the LSFR design
- The second part is to combine the LSFR sequence with message bits to form the final hash value

Generate Sequence

1. Rotate the register right one bit
2. If MSB == 1, XOR with 0x18
3. Perform these steps once for each bit in the message

Index	Start value	Rotate right	MSB == 1	XOR	Final value
1	0111 1100 (0x7C)	0011 1110	NO	-	0011 1110 (0x3e)
2	0011 1110 (0x3e)	0001 1111	NO	-	0001 1111 (0x1f)
3	0001 1111 (0x1f)	1000 1111	YES	xor 0x18	1001 0111 (0x97)
4	1001 0111 (0x97)	1100 1011	YES	xor 0x18	1101 0011 (0xd3)
5	1101 0011 (0xd3)	1110 1001	YES	xor 0x18	1111 0001 (0xf1)
...					
32	1000 0110 (0x86)	0100 0011	NO	-	0100 0011 (0x43)

```
3e 1f 97 d3 f1 e0 70 38 1c 0e 07 9b d5 f2 79 a4
52 29 8c 46 23 89 dc 6e 37 83 d9 f4 7a 3d 86 43
```

Combine with message bits

1. Initialize the hash register to 0
2. Sequence through the 24 message bits in the order they were received. (from MSB to LSB)
3. If a message bit is one, XOR the corresponding value (+4) from the LSFR sequence into the hash register.

Index	Msg bit	hash_reg	bit == 1	Calculation	New hash_reg
1	1	0x00	YES	0x00 xor 0xf1	0xf1
2	0	0xf1	NO	-	0xf1
3	1	0xf1	YES	0xf1 xor 0x70	0x81
4	0	0x81	NO	-	0x81
...					

Hacking RF – Conclusions

- Learned the mechanics of SDR radios and developed skills in recognizing signals and modulations
  - Familiarized ourselves with the most popular tools and techniques for RF Hacking
- Successfully recorded bursts and decoded packets
- Able to mimic signals in a replay attack
  - Learned some new Arduino skills like SPIFFS flash file system
  - Limited by our transmitters to ASK formats like OOK and Pulse Postion Modulation, etc.
- Could extract rolling code, modify data, create new checksum and…

Hacked John's Acurite to always read 69°C


irdeto
Building a Secure Future.™
THANK YOU!