Short Paper: A Dangerous 'Pyrotechnic Composition': Fireworks, Embedded Wireless and Insecurity-by-Design

Andrei Costin
EURECOM
Sophia Antipolis
France
andrei.costin@eurecom.fr

Aurélien Francillon
EURECOM
Sophia Antipolis
France
aurelien.francillon@eurecom.fr

ABSTRACT

Fireworks are used around the world to salute popular events such as festivals, weddings, and public or private celebrations. Besides their entertaining effects fireworks are essentially colored explosives which are sometimes directly used as weapons. Modern fireworks systems heavily rely on *wireless pyrotechnic firing systems*. Those *embedded cyber-physical systems (ECPS)* are able to remotely control pyrotechnic composition ignition. The failure to properly secure these computer sub-systems may have disastrous, if not deadly, consequences.

We describe our experience in discovering and exploiting a wireless firing system in a short amount of time without any prior knowledge of such systems. In summary, we demonstrate our methodology starting from analysis of firmware, the discovery of vulnerabilities and finally by demonstrating a real world attack. The most recent version of the firmware of this device is not vulnerable anymore to those security issues. Unfortunately, there are more than 20 vendors of similar devices that may remain vulnerable to similar attacks, in particular some of them do not have a firmware update mechanism. This suggests more that a more strict certification, with requirements for wireless security, as a realistic long term solution to the problem.

Categories and Subject Descriptors

C.3 [**SPECIAL - PURPOSE AND APPLICATION - BASED SYSTEMS**]: Real-time and embedded systems; C.2.1 [**Network Architecture and Design**]: Wireless communication; D.4.6 [**Security and Protection**]: Invasive software

Keywords

Embedded; Wireless; Firing Systems; Security; Vulnerabilities; Exploitation

WiSec'14, July 23–25, 2014, Oxford, UK.

ACM 978-1-4503-2972-9/14/07 ...$15.00.
http://dx.doi.org/10.1145/2627393.2627401 .

1. INTRODUCTION

Fireworks are essentially explosives used for entertainment purposes. A *fireworks event*, also called a *pyrotechnic show* or *fireworks show*, is a display of the effects produced by *fireworks devices*. Fireworks devices are designed to produce effects such as noise, light, smoke, floating materials (e.g., confetti). The fireworks event and fireworks devices are controlled by *fireworks firing systems*. Firing systems, besides fireworks, often serve other primary industries as well. This includes special effects and military training or simulation.

Despite the fact that fireworks are intended for celebrations, their usage is often associated with high risks of destruction, injuries, and even death. Many recent news and research studies show the dangers of fireworks [3, 24]. Sometimes fireworks are even used as real weapons in street clashes [12]. Fireworks accidents are often caused by equipment mishandling, not following safety rules or low quality of the fireworks devices. Another aggravating factor is that fireworks are generally intended to be displayed in densely crowded and public areas. All these accidents still happen despite the strict control of the distribution of fireworks and the need for a professional license to handle such devices.

Classically *fireworks firing systems* consist of mechanical or electrical switches and electric wiring (often called shooting wire). This type of setup is simple, efficient and relatively safe [5]. However, it dramatically limits the effects, complexity and capabilities of the fireworks systems and events. Advances in software, embedded and wireless technologies allows fireworks systems to take full benefit of them. A modern (wireless) firing system is at the same time a complete *embedded cyber-physical system (ECPS)* and an instance of *wireless sensor/actuator network (WSAN)*. Since fireworks firing systems are increasingly relying on wireless, embedded and software technologies, they are exposed to the very same risks as any other ECPS, WSAN or computer system.

Based on recent research, both critical and embedded systems of all types acquired a bad security reputation. For example, airplanes can be spoofed on new radar systems [15], a car control can be taken over [14, 22] and can be compromised to failure [21], an implanted insulin pump can be completely compromised [25] or an array of PLCs in a nuclear facility can be rendered nonfunctional [18, 23].

In this paper we approach the study of firing system risks from the perspective of computer, embedded and wireless security. We describe our experience in discovering and exploiting a wireless firing system in a short amount of time

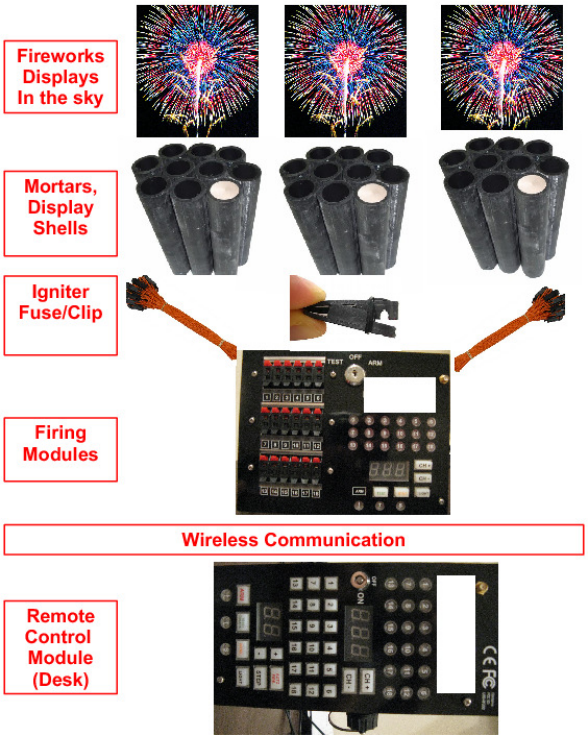

Figure 1: Generic diagram and components of a wireless firing system.

without any prior knowledge of such systems. In summary, we demonstrate our methodology starting from analysis of firmware to the discovery of vulnerabilities. Our static analysis helped our decision to acquire such a system which we analyzed in-depth. This allowed us to confirm the presence of exploitable vulnerabilities on the actual hardware. Finally, we stress on the need of hardware and software security and safety compliance enforcement for pyrotechnic firing systems.

2. OVERVIEW OF FIREWORKS SYSTEMS

Figure 1 presents a generic diagram of a fireworks firing system. A fireworks firing systems is composed of:

- *Remote control modules* (also sometimes known as *main control*) control the entire show, which includes sequencing cues and sending *fire* commands. They connect to firing modules by wired or wireless connections. In simple scenarios a single remote control module is paired with all firing modules, while in more complex shows there are several remote control modules, each one paired with a show-specific subset of firing modules. All remote control modules act independent of each other. Those devices rely on a microcontroller embedding its own firmware.

- *Firing modules* receive *fire* commands from remote control modules and activate minimum ignition current for the igniter clips. Firing modules are based on micro-controllers and have their own firmware.

- *Wired connections* are described here for completeness, however, these do not apply to our case study where remote control and firing modules are all wireless. Classic *fireworks firing systems* consist of electric wiring between remote control and firing modules [5]. Simple connection cables having End-Of-Line (EOL) resistors are used to securely terminate wire loops. EOL resistors allow the remote control to monitor the field wiring for open or short circuit conditions, hence detecting wiring problems and tampering.

- *Wireless transceivers* are enabling the wireless connections between the remote control modules and the firing modules. Those connections are often performed using 433.92 MHz modules (often capable of using *rolling codes* [2]), or 2.4GHz ZigBee compatible (IEEE 802.15.4) modules which support AES by standard. Those modules rely on microcontrollers that have their own firmware. The devices we study in section 3 are only communicating with *wireless* transceivers between the remote control modules and the firing modules, those actually support AES and several modes of operation, but do not use it.

- *Igniter clips* connect firing modules to the pyrotechnic devices housed inside mortars and ignite the fire once firing module activate the minimum necessary current.

- *Mortars* house the pyrotechnic devices; they also ensure safe launch and firing of the pyrotechnic device into the sky.

- *Pyrotechnic devices* are the actual pyrotechnic compositions which produce visual and sound effects in the sky once *fire* command is activated.

2.1 Regulation, Compliance and Certification

Many critical systems, including wireless firing systems, advertise as *"Simple, Reliable, Wireless"* or *"Proven, Secure, Reliable"*. However, such systems must first address regulation, compliance and certifications in order to be able and operate in certain geographical regions or conditions.

On the one hand, devices with fire-hazard risk, such as pyrotechnics and explosives, must conform to fire protection regulations of the country of manufacturing and/or operation. For USA, it is the National Fire Protection Association (NFPA). Specifically, NFPA-79 *"provides safeguards for industrial machinery to protect operators, equipment, facilities, and work-in-progress from fire and electrical hazards"* [9]. This standard applies to *"the electrical/electronic equipment, apparatus, or systems of industrial machines operating from a nominal voltage of 600 volts or less"*. The safety feature provided by this standard is the requirement of a key-switched operation before any potentially dangerous action can start.

This certification however does not apply to the hardware designs or the firmware implementations which control NFPA-certified *industrial machinery*.

On the other hand, all wireless or radio-frequency (RF) modules must comply with national radio-frequency licensing and allocation plans. This includes Federal Communications Commission (FCC), CE Marking (*Conformité Européenne*) and Industry Canada (IC) certification. The system we analyze contains a California Eastern Labs (CEL) IEEE 802.15.4 2.4GHz RF transceiver, which is CE and FCC

certified. However, those certifications do not apply to the security of the communication channels or network protocols or of the firmware, but only to the transceiver.

We argue that, given the risk of the devices controlled by such equipment, a certification, based on a security evaluation of the architecture, firmware and communications should be mandatory. We show in this paper that this is not the case.

As a counter-example we consider the avionics field. Avionics encompass virtually the entire spectrum of hardware and software involved in the aviation field where safety and high-risk are considered. All avionics devices must pass strict compliance testing for both hardware (DO-254) and software (DO-178B) [20]. Despite those certifications there are recent examples of wireless avionics protocols shown to be deployed without security [15].

3. EXPERIMENTS AND RESULTS

3.1 Summary

In [16] we performed a large-scale firmware analysis by crawling the Internet for firmware images, reaching 172K firmware candidates. After unpacking firmware images, we run simple static analysis, correlation and reporting tools on each firmware image, which lead us to discover 38 previously unknown vulnerabilities. In this process, by pure chance, we discovered the firmware images of a wireless firing system. We deliberately omit the name of the vendor and the system for safety and ethical reasons.

Analysis of firmware images for that system has shown us components (strings, binary code, configurations) which appeared insecure [1]. The findings were convincing enough that we acquired the devices for a detailed analysis. Another factor to motivate the acquisition is that according to the vendor, this system is used by *"over 1000 customers in over 60 countries"*.Hence, these systems appear to be particularly popular among fireworks display companies and can be exploited on large geographical areas and can impact a wide range of public events.

3.2 Firmware Acquisition and Static Analysis

Among many others, our crawlers collected from the Internet several firmware images, in *Intel Hexadecimal Object Files (iHex)* format, dedicated to the the wireless firing system. After unpacking, we use several heuristics, including *keyword matching*. *keyword matching* searches for special keywords such as `backdoor, telnet, UART, shell` which often allows to find multiple vulnerabilities. The firmware images were matching the string `Shell>`. Based on this we isolated those firmware images and proceeded to analyze them further with automated and manual approaches.

We identified several security issues with the firmware images we analyzed. First, plain iHex format doesn't provide any encryption or authentication hence the functionality is openly accessible for study by the attacker and likely open to malicious firmware modifications. In addition to this, iHex format provides mechanisms that can be use by attackers to insert code or data into memory regions that might have not been designed to be accessible.

[1] This analysis was performed on the stable firmware as of Nov 2013, meanwhile a new firmware addressing most of the security issues we discovered was made stable, and is now deployed.

3.3 Hardware Acquisition and Analysis

The static analysis findings were convincing enough that we acquired the actual wireless firing system to analyze it further. Indeed, static analysis is known to be faster and to scale better than dynamic analysis as it does not require access to the physical devices. However, one important research challenge remains to confirm the results of static analysis. The analyzed firmware images were designed to run on specific embedded devices, without the actual hardware, it is very hard to confirm the discovered vulnerabilities. Indeed, findings of the static analysis study may be not exploitable in a live system, e.g., because the vulnerable code is not executed, or is activated by a configuration option. Even though this could be discovered by emulation, it is a tedious process in itself as it can be error prone and a generic emulator would need to be customized to emulate this particular platform.

These systems usually come bundled with firing modules and remote control modules. The exact number and placement of each module depends on the setup and choreography of each fireworks show. Both of these modules contain *CEL MeshConnect* 2.4GHz ZigBee (IEEE 802.15.4) transceivers [10]. Both modules are equipped with key-operated switches, as required by NFPA-79 chapter 9.2.

Both modules provide a servicing serial port (UART) which provides access to a built-in menu which displays the `Shell>` prompt we discovered earlier. This allows to testing, repair and debug the remote control module. This UART port is only accessible by physically removing the plastic chassis of the modules and as such, it can be abused only with physical attacks. When properly secured, this port could be used for example to restore or update the AES-128 encryption keys of the wireless ZigBee transceivers. In addition to the above, the USB *SNAP Stick SS200* [13] provides reprogramming and sniffing functions over 2.4GHz ZigBee (IEEE 802.15.4), and is tailored in particular for SNAP chipsets and software.

Remote Control Module

A detailed view of main components of the remote control module can be seen on Figure 2. After remote control module's disassembly, we confirmed that it uses a *ColdFire MCF52254* processor from Freescale [8]. This is consistent with the result of *Motorola m68k family* provided by our architecture detection tool in Section 3.2. It also uses a *SST25VF032B* flash chip by Microchip [7].

The remote control module exposes a USB port. This port has two main functions. One function is to upload a fireworks show orchestration script. This orchestrator script is a CSV file which instructs the main processor of the remote control module to which firing module and when to send firing cue signals in order to achieve the planned visual, sound or smoke effects. Another function is to upgrade the firmware of the main (not wireless) micro-controller unit (MCU) of the device. This is done via an `.ihex` file, as described in Section 3.2.

3.4 Wireless Analysis

This systems, as many others from other vendors, contains a 2.4GHz ZigBee (IEEE 802.15.4) CEL MeshConnect transceiver. The discovery, configuration query and setup, pairing and firmware upgrade of these units is done through

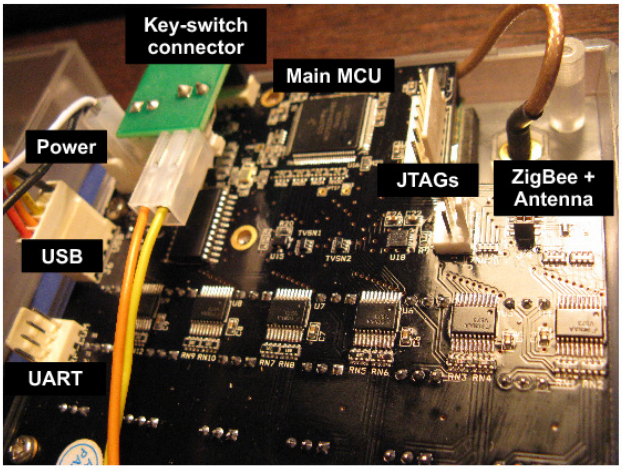

Figure 2: Remote control module's hardware.

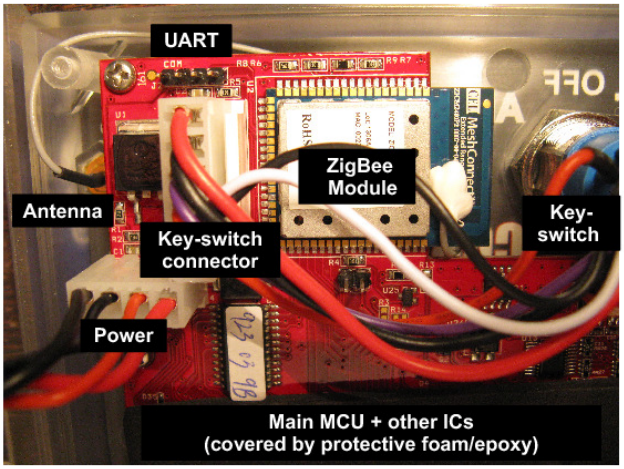

Figure 3: Firing module's hardware.

Synapse Portal [2] software. We installed *Synapse Portal* and then ran the discovery and configuration query. The wireless chipsets on remote control, firing and firmware reprogramming modules have AES-128 capable firmware installed. However, the encryption is not enabled, no encryption key is present and AES-128 seems to be unused. In addition to this, the system's documentation didn't seem to support AES-128 secured configuration steps. Surprisingly, even though those devices are standard compliant and as such have AES-128 capabilities, neither authentication nor encryption of the messages are used. This is most likely due to the difficulty to properly setup key management and distribution, and that could be perceived more as a risk of operational failure during a fireworks show, rather than a securing mechanism.

Further analysis revealed that it is possible to upload Python application scripts to remote wireless chipsets. These scripts are executed in a Python interpreter within the wireless chipset's MCU [10]. The provided interpreter framework is a subset of Python. Before being uploaded to target nodes, *Synapse Portal* compiles these Python scripts into binary form and stores them as SNAPpy files (with extension `.spy`) files [11]. The binary form is targeted for the specific MCU which drives each wireless chipsets. These scripts expose entry-points (functions) that can be remotely called (via RPC) by other wireless nodes. These scripts can interact with the MCU of the wireless chipsets or with GPIO-ports of the wireless chipsets. Usually those GPIO-ports are connected to the main MCU of the remote control or of the firing module. This allows interaction with the main MCUs as well as with IO peripherals such as buttons, displays and igniter clips.

The typical use of script entry-points is as follows. The remote control modules process the CSV orchestration scripts. When it decides a *fire* command is required, it sends a ZigBee packet containing a higher-level message to call a specific entry-point on a specific remote module.

The usual procedure of *normal firing* is as follows. The firing modules are paired with a particular remote control module. Subsequently, firing modules will accept the *arm*, *disarm* and *fire* commands only from the paired remote control module. The pairing is enforced by checking remote control's 802.15.4 *short address* (similar to a MAC address filtering). The physical key on all firing modules are turned into *arm* position. The staff departs to the safe regulatory distance to fire the cues. The key on all remote control modules are turned to *on*. The staff confirms everything is safe and ready, and then presses the *arm* button on the remote control, which in turn wirelessly sends a *digital arm* command to firing modules. The firing modules enter a *confirmed arm*, ready for subsequent *fire* command. The staff starts the show by sending, either manually or scripted, *fire* commands to corresponding firing module's cues.

3.4.1 Wireless Attacks

The lack of encryption and mutual unit authentication, opens the system to multiple attacks, in particular sniffing, spoofing and replaying.

We describe a simple attack, yet which we consider as the most dangerous for the fireworks show staff members. The attacker would perform the following sequence of operations in a continuous manner. Eavesdrop the packets (broadcasts, multicasts, node-to-node), from those learn the 802.15.4 addresses of each remote control and firing modules, and learn their corresponding pairing. For each learned pair, the attacker spoofs the remote control's 802.15.4 addresses, spoofs the *digital arm* command to the pair's firing module, and immediately send *fire* command for all cues once *digital arm* confirmation comes from the firing module. The consequence of this attack is that as soon as the show operator will turn the physical key of a given firing module to *arm* position, it will immediately receive the sequence of *digital arm* and *fire* for all cues. This will fire all the pyrotechnic loads and in the worst case will not allow enough time for the staff to depart to the safe distance. Thus it will defeat the physical key safety and function separation. We *successfully implemented* this attack using components described in Section 3.4.2 and *tested* this attack in practice on the systems we acquired.

Alternatively, an attacker could easily replace default Python functions responsible for firing cues, with arbitrary malicious Python functions. For example, each malicious firing cue function could fire all cues at once instead of firing only it's own cue, thus potentially producing a massive chain explosion. Or it could not fire cues at all or fire them at random, rendering the fireworks show below expectations. Last but not least, an attacker can remotely set random encryp-

[2] `http://www.synapse-wireless.com/snap-components-free-developers-IDE-tools/portal`

tion keys on remote nodes. This would result in a denial-of-service for the legitimate user, since her legitimate devices would not be able to communicate with exploited devices anymore. This can definitely ruin a holiday celebration or produce disadvantages to competitors in professional fireworks competitions.

3.4.2 Wireless Attack Implementation

SNAP Stick SS200 [13]. It is mainly a firmware programmer for the remote control and firing modules and is based on well-known ATmega128RFA1 chipset from Atmel. Conveniently, using *SNAP Portal*'s utilities, and a special proprietary firmware for it, made available by Synapse as *ATmega128RFA1_Sniffer*, it can be turned into a SNAP-specific 802.15.4 sniffer, where it sniffs and decodes 802.15.4 packets based on Synapse's higher level protocol semantics (e.g., multicasts, broadcasts, peer or multicast RPC calls). We used it to sniff and record the packets between remote control and firing modules during their normal operations. Finally, we also used it to validate our packet injection and replay attacks. If this sniffer received them, then the remote control and firing modules would see our rogue packets. Otherwise we had to fix our injector (regardless the fact that our lower level raw packet sniffer could see them), and then test again sniffed packets and actual devices' behavior.

Goodfet [1]. It is an embedded bus adapter for various microcontrollers and radios, additionally proving great open-source support for advanced attacks. It conveniently provides firmware for TelosB devices to allow sniffing among other functionalities. We tested our attack with this Goodfet firmware running on TelosB.

KillerBee [6]. It is a framework and tools for exploiting ZigBee and 802.15.4 networks. It conveniently provides a pre-compiled Goodfet firmware for extra attack functionality. We tested our attack with this Goodfet firmware running on TelosB.

Crossbow's TelosB. The sniffer based on SS200 is useful for SNAP protocols and visualization, but it filters out and strips down the packets, hence is largely limiting. We required a lower level raw packet sniffer. We also required an inexpensive and open-source supported approach. TelosB hardware and Goodfet firmware was a perfect fit, so we used them as an additional, much more verbose and raw, sniffer. After learning the SS200 higher level packets for critical commands we correlated them with raw packets recorded by TelosB-Goodfet. Alternatively, a Zigduino [3]could have been used for this task.

Econotag [4]. Econotag is a inexpensive and convenient open-source platform for 802.15.4 networks. We assembled sequences of packets instructing to arm and fire sent from the remote control module to the firing module. Finally, we coded an infinite loop of these sequences in a custom firmware. Once plugged, the Econotag successfully performs the attack on a firing module once it's key is turned to *physical arm* position. A Zigduino could have been used for this task as well.

[3]`http://www.logos-electro.com/zigduino/`

Implementation notes. We implemented a simple attack, however it is obvious and trivial to extend the implementation to automatically and continuously sniff new firing modules, and subsequently spoof remote control sequences.

3.5 Solutions

Below we summarize a set of recommendations that can dramatically increase the security of the hardware, firmware and wireless communication of the analyzed wireless firing system. With increased security, a safer operation of the entire system can be achieved:

- Provide "factory reset button" to a "factory safe" image and state – this can help reset the wireless chipsets to no encryption state when wireless crypto key (e.g., AES-128) is forgotten.
- In "basic mode" – a clear-text and insecure mode, allow only testing functionality (e.g., identification, communication, continuity).
- In "secure mode" – a mutual authenticated and encrypted mode, allow additional functionality such as *fire* command to igniter clips and firmware upgrade of the both main and wireless MCUs.
- Implement "secure scan" techniques [19] – to allow debugging, testing and restoring of the main MCU and board.
- Remote-code attestation – ensuring, via static or dynamic root of trust, that safety critical code is not tampered with; this could be achieved via minor hardware and firmware modifications, for example as presented in SMART [17].
- Formal verification – this can dramatically increase security and safety of firmware, hardware and communication protocols.
- Compliance standards and testing – strict compliance testing for both hardware and software, similar to DO-254 and DO-178B respectively.

4. FUTURE WORK

On the one hand, we aim at implementing an attack of wireless remote firmware upgrade of the main MCU via the 2.4GHz ZigBee (IEEE 802.15.4) chipsets. This is opposite to the current procedure, where the firmware upgrade is initiated from a USB stick connected locally to the device under attack. Since we have the actual devices under our full control, we also aim at using a dynamic analysis platform for firmware security testing, such as Avatar [26]. In Avatar, instructions are executed in an emulator, while the IO accesses to the embedded system's peripherals are forwarded to the real device. An additional aim is to find vulnerabilities in the CSV parser of the remote control to achieve a USB plug-and-exploit proof of concept.

On the other hand, we aim at finding solutions to help this particular category of devices. Solutions not specific to wireless firing systems, include secure firmware upgrades, encrypted and authorized wireless communication channels, secure restore and debug chains. Finally, wireless firing systems specific solutions include secure latency control and secure positioning.

5. CONCLUSIONS

We presented vulnerability discovery and exploitation of wireless firing systems in a short amount of time without prior knowledge of such systems. We started with an automated large-scale framework for firmware crawling and analysis [16], and employed simple heuristics (e.g., keyword matching) and very simple static analysis. We were able to quickly and automatically isolate firmwares of critically important remote firing systems and identify several potential vulnerabilities through both automatic and manual static analysis. These vulnerabilities include unauthenticated firmware upgrade, unauthenticated wireless communications, sniffing and spoofing wireless communications, arbitrary code injection and functionality trigger, temporary denial-of-service. We successfully implemented and tested an unsophisticated attack with potentially devastating consequences.

We conclude that, given the risk presented by their usage, the security of wireless firing systems should be taken very seriously. We also conclude that such systems must be more rigorously certified and regulated. We stress on the necessity and urgency to introduce software and hardware compliance verification similar to DO-178B and DO-254 respectively. We strongly believe these small improvement steps, along with solutions in Section 3.5, can definitely help increase the security and safety of such wireless embedded systems.

Last but not least, we discussed the issues with the vendor. A firmware update that is now deployed is addressing most of the security issues. Unfortunately, there are more than 20 vendors of wireless firing systems that may remain vulnerable to similar attacks, in particular some of them do not have a firmware update mechanism.

Acknowledgments

We thank the anonymous reviewers, as well as Scott Smith, for their comments and suggestions for improving this paper. In particular we thank our shepherd, Jens Schmitt, for his valuable time and inputs guiding this paper for publication.

The research leading to these results was partially funded by the European Union Seventh Framework Programme (contract Nr 257007).

Disclaimer

We remind that performing such attacks outside of a lab is an illegal activity prohibited by law. Eurecom cannot be held responsible in case of an abuse of such techniques.

6. REFERENCES

[1] `https://github.com/travisgoodspeed/goodfet`.

[2] Atmel AppNote AVR411: Secure Rolling Code Algorithm for Wireless Link.

[3] California Fireworks Display Goes Horribly Wrong: Dozens injured by catastrophic misfire during Simi Valley Fourth of July. ABCNews, 5th July 2013.

[4] Econotag. `http://redwire.myshopify.com/`.

[5] Fireworks Electric (Wired) Firing Systems. `http://www.skylighter.com/fireworks/how-to/setup-electric-firing-systems.asp`.

[6] KillerBee; Framework and tools for exploiting ZigBee and IEEE 802.15.4 networks. `http://code.google.com/p/killerbee/`.

[7] Microchip SST25VF032B Flash Chip Datasheet.

[8] Motorola ColdFire MCF52254 Processor Datasheet.

[9] NFPA 79: Electrical Standard for Industrial Machinery. `http://www.nfpa.org/79`.

[10] Synapse Module Comparison Chart. `http://content.solarbotics.com/products/documentation/synapse_comparison_table.pdf`.

[11] Synapse SNAP Network Operating System – Reference Manual, v2.4, 2012.

[12] Ukraine protests: Kiev fireworks 'rain on police'. `http://www.bbc.com/news/world-europe-25820899`.

[13] USB Snap Stick SS200. `https://www.synapse-wireless.com/snap-components/usb-mesh-snap-stick`.

[14] S. Checkoway, D. McCoy, D. Anderson, B. Kantor, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno. Comprehensive Experimental Analysis of Automotive Attack Surfaces. In *Proceedings of the USENIX Security Symposium*, San Francisco, CA, August 2011.

[15] A. Costin and A. Francillon. Ghost in the Air (Traffic): On insecurity of ADS-B protocol and practical attacks on ADS-B devices. *Black Hat USA*, July 2012.

[16] A. Costin, J. Zaddach, A. Francillon, and D. Balzarotti. A Large Scale Analysis of the Security of Embedded Firmwares. In *To appear at USENIX Security Symposium*, August 2014.

[17] K. El Defrawy, A. Francillon, D. Perito, and G. Tsudik. Smart: Secure and minimal architecture for (establishing a dynamic) root of trust. In *Proceedings of the Network & Distributed System Security Symposium, San Diego, CA*, 2012.

[18] N. Falliere, L. O. Murchu, and E. Chien. W32.Stuxnet Dossier. *White paper, Symantec Corp., Security Response*, 2011.

[19] D. Hely, F. Bancel, M.-L. Flottes, and B. Rouzeyre. Secure scan techniques: a comparison. In *On-Line Testing Symposium, 2006. IOLTS 2006. 12th IEEE International*, pages 6–pp. IEEE, 2006.

[20] V. Hilderman and T. Baghi. *Avionics certification: a complete guide to DO-178 (software), DO-254 (hardware)*. 2007.

[21] J. Hirsch and K. Bensinger. Toyota settles acceleration lawsuit after $3-million verdict. Los Angeles Times, October 25, 2013.

[22] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. Experimental security analysis of a modern automobile. *2010 IEEE Symposium on Security and Privacy*.

[23] R. Langner. Stuxnet: Dissecting a cyberwarfare weapon. *Security & Privacy, IEEE*, 9(3):49–51, 2011.

[24] V. Puri, S. Mahendru, R. Rana, and M. Deshpande. Firework injuries: a ten-year study. *Journal of Plastic, Reconstructive & Aesthetic Surgery*, 62(9), 2009.

[25] J. Radcliffe. Hacking Medical Devices for Fun and Insulin: Breaking the Human SCADA System, 2011.

[26] J. Zaddach, L. Bruno, A. Francillon, and D. Balzarotti. Avatar: A Framework to Support Dynamic Security Analysis of Embedded Systems' Firmwares. In *Network and Distributed System Security Symposium*, NDSS 14, February 2014.