Assessing the security of “simple” IoT devices

Author: Dirk-Jan Out, CEO Brightsight

Abstract

The Internet of Things (IoT) is becoming more widespread every day, and with this spread, the number of security incidents involving IoT is rising. Industry and regulators worldwide are looking at security testing/evaluation as a possible solution for the security problems introduced by IoT. Many initiatives take a silo approach and are looking at security testing/evaluation of various product types (e.g. IP-camera’s, connected refrigerators, or automotive product).

This paper examines the assessment of security of relatively simple IoT endpoint devices and shows that assessing these endpoint devices as a whole (the aforementioned product types) is problematical and un-economical. In this paper we advocate a two-step approach consisting of a thorough security evaluation of so-called platforms, followed by a high-level and fast security test of the final product. This approach is more aligned with the economic interests and technical possibilities of the developers of IoT products, and is therefore much more suitable.

1. Security of “Simple” IoT devices

In this white paper, we consider the security of “simple” IoT devices, e.g. an IoT doorbell.

The question that we answer is ‘*How can the security of “simple” IoT devices be assured in a cost-effective way?*’

We define “simple” IoT devices as devices consisting of a single IC with software running on it, which may be mounted on a circuit board with additional components (but all security functionality is contained in the IC plus software) and is connected to a network. We also consider larger devices, such as connected refrigerators, to be “simple” if all security-relevant functionality of that device is concentrated into a single IC.

Assuring the security of “simple” IoT devices turns out to be not so simple when the devices are examined in more detail, as a “simple” device typically contains:

- An IC, that may have security functionality such as encryption/decryption, memory management, random number generation
- An operating system as platform for programming
- Authentication functionality to ensure that not everyone can access the device
- Access control to ensure that different persons can have different levels of access
- Storage for keys, passwords, settings and data
- Functionality for provisioning and/or updating the device
- Network functions for integration with other IoT devices
- And finally, an application that performs the actual functions of the device

Each of these parts is complex in itself and may contain entry points for attacks on the device, leading to subversion of the entire device.

Lesson #1: “Simple” IoT devices are security-wise often quite complex

Some IoT devices, such as a connected doorbell, seem wholly unrelated to security. They protect no assets, and it seems that no attacker will be interested in hacking them. Other IoT devices do have assets to protect and may well be of interest to attackers, such as IP cameras that may protect privacy and connected door locks that may prevent unauthorised access.

However, due to their connectivity, IoT devices often also have to protect various assets that do not directly “belong” to them such as:

- Access to the network that they are connected to (and other devices on that network)
- Easy access to data and services on cloud servers that “trust” the IoT device
- Passwords that the user may also use for other products and services
- Privacy-sensitive information of the owner of the product
- The ability to generate network traffic, that can be used for denial-of-service attacks on other systems

In 2016 a French website suffered the (at that time) largest distributed denial of service attack ever, as a botnet containing 145.000 hacked IP-cameras and connected digital video recorders started to send huge volumes of traffic (adding up to more than 1 Terabit per second) to the website, making it totally unavailable.
Source: arstechnica.com/information-technology/2016/09/botnet-of-145k-cameras-reportedly-deliver-internets-biggest-ddos-ever/

Should the IoT device be successfully hacked, these indirect assets can also become compromised.

A North-American casino installed a high-tech fish tank as an attraction for its customers. This fish tank used advanced sensors that automatically regulated temperature, salinity and feeding schedules. A hacker gained access to the fish tank and managed to copy 10GB of data from the casino network to which the fish tank was connected.
Source: DarkTrace Global Threat Report, 2017, www.darktrace.com

Lesson #2: IoT devices (even those who have no assets of their own), can provide hackers access to important assets.

2. Assuring the Security of IoT Devices

The big problem with IoT devices and security is that the security is **invisible** to end-users.

Typical questions such as “is this IoT device secure”, “what does the developer mean with secure” or “what are the responsibilities of the end-user towards security” go unanswered. An end user can therefore not determine whether a device is secure.

Many developers of IoT devices have little experience with security, and find development cost and time-to-market much more important. Even developers that do understand security and find it important cannot always be certain of the security of their devices: they often rely upon hard- and software they obtained from other developers, who also do not give ironclad guarantees.

If both end-users and developers cannot determine whether an IoT device is secure (or even what it means for an IoT device to be secure), an authorative third party specialised in security is required.

Such a party can assess the security of the device by security testing or security evaluation.

Security testing typically consists of:

- Examining the existing security functionality and trying out various attacks. E.g. a device has a password, and one would examine “is the user forced to change it when he first uses it?”, “is there a minimum length and minimum complexity of the password”, “what happens when you enter an empty password”, “what happens when you enter a password of 10.000 characters”, “what happens when you enter a password with various control characters” etc.
- Running automated vulnerability scanners (e.g. Nessus) to find out if there are known vulnerabilities in the device.
- Running fuzzers (programs that generate random input, such as ISIC) to see if the device will misbehave.
- Performing measurements on the device and using signal analysis techniques to extract secrets from the device.

Security testing is cheap, relatively quick, requires relatively little skill, but it will find only the most obvious flaws. This may not be the level of trust that needs to achieved, as hackers can spend way more time on the device and may well uncover flaws that were not detected by the security testing.

In other words, security testing is aimed at showing that either an IoT device is vulnerable, or the tester ran out of time trying to hack it.

Lesson #3: Security testing of “simple” IoT devices is far from complete and therefore less meaningful

Security evaluation typically consists of some combination of:

- Discussing the design together with the developer so as to understand both the design itself, and the underlying security architecture and security assumptions
- Examining the source code to find whether programming mistakes are made that can lead to vulnerabilities

The combination of these two steps leads to a solid argumentation why the IoT device is secure, or where that cannot be argued solidly: targeted ideas for possible vulnerabilities in the IoT device. The IoT device is then tested to determine whether these vulnerabilities are actually present and can be exploited.

Additional activities that can be performed as part of the security evaluation are:

- Examination of the developer’s testing to verify that the security of the device works as expected in case of normal behaviour and just beyond its limits
- Examination of the developer’s quality processes to determine that the development process is likely to result in secure devices
- Examination of the entire lifecycle of the device (sourcing of components, production, deployment, use, disposal) to ensure that no vulnerabilities are introduced there
- Examination of the manuals to determine that they provide users with sufficiently clear information to correctly install and operate the device

Security evaluation will uncover many more security vulnerabilities than security testing, but is also more time-consuming and therefore more expensive. This may be prohibitive for many IoT devices.

In addition, many developers of IoT devices will have bought significant components of their device, such as the IC, the OS, libraries etc. and have no clue on how these components are designed and tested. They do not have the source code and have no knowledge on the quality processes of their suppliers and cannot provide this information. In this case security evaluation becomes impossible.

Lesson #4: Security evaluation of “simple” IoT devices is time-consuming and in many cases impossible.

So we seem to be stuck between incomplete on one hand and time-consuming/impossible on the other. Multiply this problem with 10.000 or more different IoT devices and the problem becomes intractable.

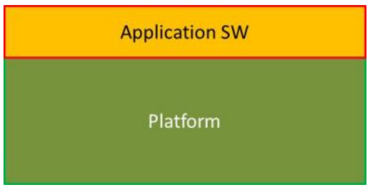
Combining lessons #3 and #4 leads to:

Lesson #5: It is not practical to perform security testing or security evaluation on (tens of) thousands of different types of “simple” IoT devices.

This seems pretty damning for the assessment of security on “simple” IoT devices. Now what?

3. Splitting IoT devices

The solution that we propose is to split the IoT device in two layers, a generic platform and a specific application, and subsequently perform a security evaluation on the platform and security testing on the application plus the platform.



The platform will consist of an IC, an operating system and perhaps some libraries. Ideally, this platform will be developed by a developer[1] who specialises in creating platforms for many application developers, and has a good understanding of security.

The platform should provide generic security services such as Access Control, Authentication, Random Number Generation, Secure Boot, Secure Communication, Secure Storage, Secure Update, Platform/Application(s) Separation, and Trusted Time with clearly defined and well-documented interfaces. Platforms do not have to deliver all security services, they can provide some subset: application developers can then choose between platforms that provide the services they need and that are determined to be secure.

The application consists of software running on the platform, and relies on the security services provided by the platform, by calling the defined interfaces. The IoT device can therefore be developed by a developer with only limited understanding of security (e.g. to send this information

[1] This is the simple case where one developer produces the entire platform. Section 4 addresses involvement of multiple developers.

to the web site, use the secure_transmission_to_website() function). The application developer can now concentrate his development efforts on the specific functionality to be delivered by the application instead of providing security.

Case 1 A developer of personal medical devices wanted to be able to read out these devices by smartphone, and for this reason, wanted to add Bluetooth LE connectivity to his device. The developer purchased a Bluetooth LE compatible platform to run his application. Because the medical developer did not want to rely on the security claims of the platform developer, the medical developer ran a number of security tests and found the platform wanting. Several iterations followed where the medical device developer pointed out bugs and vulnerabilities and the platform developer patched these.
In essence, the medical device developer assisted the platform developer with security: an unusual reversal of roles, which cost the medical developer significant effort, and more importantly: time-to-market. Had a certified platform been available, this could have been avoided.

Case 2 ExpressLogic (rtos.com) is having the security of its X-Ware IoT Platform Secure Cloud evaluated by Brightsight using Common Criteria EAL4+, which is a thorough security certification involving design review, source code review, site visits, vulnerability analysis and penetration testing. The emphasis of the certification lies on secure communication between the platform and cloud servers (over TLS over IPv4 and IPv6). Once the platform has been certified, applications that run on the platform and use TLS to securely communicate with cloud providers[2], can be easily and quickly certified as well: in that certification one only has to check whether they use the TLS correctly, and not use other communication channels.

These two cases show that the platform/application split approach works, but only if the application developer can fully rely on the platform. If he can (because the platform has been successfully evaluated), the split approach between platform and application has the following advantages:

1. *Reduction of overall evaluation effort*: There are fifty or so platform developers against tens of thousands of application developers. This means that deep understanding of security only needs to be present in fifty companies rather than tens of thousands.
2. *Increase in reach of the results*: A single platform can be used for many products (sometimes hundreds). The efforts spent on the platform evaluation are therefore magnified and help secure many end products.
3. *Re-use of evaluation results*: As platforms are re-used by many application developers, the costs of the deep security evaluation are divided over many application developers, rather than borne by a single application developer.
4. *Quick security testing of the application*: The majority of the security of a total solution will be provided by the platform. In this case, a security evaluation of the total solution can be done by using an evaluated platform, a simple check whether the application uses the platform correctly, and a quick test of the application. This can be done cheaply and quickly.

[2] Such as Alibaba, Amazon, Baidu, Google, IBM, Microsoft, Tencent and Xively

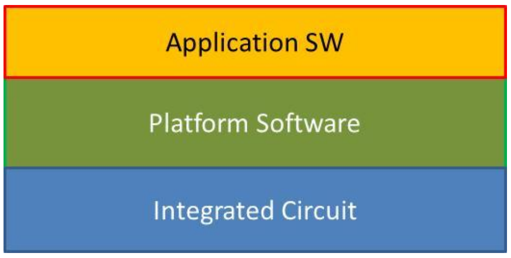
Lesson #6: It is possible to provide meaningful and affordable security evaluation on dozens of platforms.

Lesson #7: It is possible to provide meaningful and affordable security testing on an application on top of an evaluated platform.

Lesson #8: The combination of security evaluation on platforms and security testing on an application on top of that platform is a practical, meaningful and cost-effective way to assess the security of IoT devices.

4. Iterating the split

The solution proposed in the previous section can made even more practical and cost-effective. In many cases, a platform developer produces only the software part of a platform and he buys the hardware part from one or more hardware developers. In this case, a similar argument holds: one can split the platform in a hardware and software part.



Subsequently, one can define generic security services to be provided by the hardware to the software, such as Secure Boot, Cryptography and Random Number Generation and evaluate these. The software can then use these services and combine them with its own functionality to provide its own services to the application, with similar advantages:

1. *Reduction of overall evaluation effort*: There are 10 or so IC developers against 50 or so platform developers, so specialised HW security knowledge need only be present in even fewer companies
2. *Spreading of costs:* ICs are used by many platform developers. This means that the costs of an IC can be divided over multiple platform developers, rather than be borne by a single one.

And one can take this split even further by including libraries from yet another developer into the platform software. If such a library (e.g. a very fast and secure implementation of an IP-stack) is itself evaluated, it can be re-used in the platform software of many other developers.

5. Summary

In this white paper, we have analysed how the security of “simple” IoT devices can be assured in a cost-effective way. We have shown that even these “simple” devices are often quite complex from a security perspective. We have also shown that even seemingly innocent devices like connected doorbells may provide access to more valuable assets outside the direct influence of end-users, and are therefore at risk of being attacked.

We have argued that security testing/evaluation can help in providing assurance to IoT devices but that security testing of a complete IoT device is far from complete and therefore less meaningful, while security evaluation of complete IoT devices is time-consuming and often impossible, and that one therefore cannot do meaningful and affordable security testing or evaluation on the vast horde of IoT devices that are in the market now and in the near future.

We have then presented a practical solution: if one splits a device into application and platform, once can apply security evaluation to the platform and security testing on the application. The combination of the two leads to a combined result that is affordable and meaningful.

6. Further work

This paper leaves a number of practical issues unanswered, and these will be addressed in Part 2 of this paper:

1. **Security evaluation:** How does one actually perform the security evaluation of platforms? What are the requirements?

2. **Security testing:** How does one actually perform the security testing of applications, given that the platform it resides upon has been security evaluated?

3. **Composition:** When you have an application that has been tested on one platform, is it possible to easily re-test this on another platform? Similarly, if one has platform software evaluated on one IC, is it possible to easily re-evaluate this on another IC?

About the author:
Dirk-Jan Out is the CEO of Brightsight BV, the number one security lab in the world. Brightsight has over 35 years of experience in evaluation security products and is a Common Criteria and EMVCo hardware and software security lab. With our over 160 security evaluators, based in our headquarters in Delft, The Netherlands and local offices in Barcelona and Beijing, we serve our global customers.