Eberhard von Faber, Walter Sedlacek

# Using Game Theory to Improve IT Security in the Internet of Things

## The Idea of a Durability Date or: What happens if nobody cares?

Game theory is a branch of mathematics and economics. It aims predicting rational and hence actually observed human decisions and to understand the reasons for an alternative to be preferred to another. Decisions of market participants or players considerably influence the achieved level of information security since the latter is primarily determined by the facts if somebody cares about security and if market participants force others to do so. In the Internet of Things many is at sixes and sevens. Vulnerabilities in cameras and other devices are exploited to turn them into weapons. Can Game Theory be used to cause a change for the better?

### Einleitung

Computer systems and other electronic, IT-equipped devices or systems frequently possess vulnerabilities (security gaps), which are able to be exploited by attackers. A common method for eliminating such vulnerabilities entails the updating of software, typically defined as patching. Especially in case of objects of everyday use which contain software and have a connection to the Internet, **a danger exists that is not known whether the software updating is planned in the first place and secondly, actually carried out. Thirdly making things worse**, device and software manufacturers use software from third parties for which they do not feel responsible. This article exclusively deals with such a problem and outlines a possible solution for it. Additionally, it will be described how the updating of software in different computer systems is ensured. This application primarily takes place in industrially operated data centers. The solution hereby presented is to be understood as an idea and as food for thought. The advantages and disadvantages involved will be a subject of discussion. **Scientific work on Game Theory was granted the Nobel prize for economics eight times. This emphasizes the importance of this sometimes underestimated discipline which is exemplary applied to IT security in this paper.**

**Prof. Dr. Eberhard von Faber**

T-Systems, Chief Security Advisor, IT Division; working areas: security archtitecture, developer of ESARIS, secure IT production, secure IT outsourcing, integration into processes and ITIL, standardization, cloud, IAM
E.von-Faber@t-systems.com

**Walter Sedlacek**

Mag., MSc MBA PMP; T-Systems; different roles in intern. management and project management; management of the intern. Roll-out of ESARIS; head of the regional data center in Singapure
Walter.Sedlacek@t-systems.com

### 1. Background and Problem Description

#### 1.1 Subject

Today's modern Internet unites two characteristics, resp. trends: the decentral, distributed use and compilation of information on one hand; and the central provision of IT-services on the other hand. Both areas are not separate from each other. On the contrary, central applications increasingly use and process data generated by decentrally distributed components and devices; and they also make data available for those components and devices. Sensors, which are distributed throughout the Internet, compile information, which is centrally processed. Actuators receive their control command from central IT-applications. For several years now, the number of electronic, IT-equipped devices, which act as sensors and actuators and also process information them-

selves, has tremendously risen. As such devices are built in everyday items and in industrial plants or such, one also coins this term the Internet of Things (IoT).

We define the decentrally distributed components and devices in a simplified manner as being devices in the Internet of Things (IoT), resp. abbreviated to IoT-devices. (That this is a simplification, because, e.g., PCs in private hands are actually not IoT-devices, is irrelevant for the discussion that follows.) Central components and devices represent the opposite.

Nevertheless, the difference between these types of devices is not really in where they are. It is of much more significance if they are clearly in the possession of and in the care of an IT-service provider, because they, e.g., are installed in their data center, or if the possession or the care is not so clearly regulated, resp., recognizable. Why is the difference so important? One may be of the assumption that an IT-service provider sees the updating of the software of his systems as his job, insofar the IT-service provider is directly or indirectly affected if possible vulnerabilities are not eliminated. (This point will be elaborated on in Chapter 2.) On the other hand, there are, however, IoT-devices that do not make it clear to the user or operator if the software updating is supported and carried out. (A possible solution is outlined in Chapter 3.) First of all, the problem should be thoroughly explained; and through the use of examples thus be elucidated.

### 1.2 Problem Description Based on Examples

Let us observe a few examples of such IoT-devices to better understand this problem. *First:* to directly attack IT-services, resp. servers or, e.g., to bring malware into circulation, attackers build up so-called bot networks. They consist of a multitude of captured computer systems, which are remotely steered, without the authorization of the owner; and they are misused by the operator of the botnet. For a long time, vulnerable PCs have been seized and made a part of such a bot network. For some time now, a new level of quality has emerged through the Internet of Things. It was reported in 2016 that bot networks such as "Mirai" convert inexpensive Internet-cameras in the hundreds of thousands for the purpose of being able to misuse them. This was made possible due to vulnerabilities in the software of these cameras.

*Second:* It is not so far ago that an unknown IT-security specialist with the pseudonym „Kenzo2017" issued a warning that certain routers (They are no IoT-devices in the true sense of the word ), which households and companies are connected to, are susceptible to being remotely steered and to being exploited for attacks. The manufacturer issued a software update; but nothing happened. *Third:* Industrial plants such as wind power plants, hydro-electric plants or other machines operated by small enterprises transfer measuring data and diagnostic data over the Internet to central applications in data centers and receive control commands on the same path. For the exchange of this data, standard components are installed in the industrial plants. The owners and the operators of industrial plants, as well as those of video recorders and TVs, are often totally unaware that these IoT-devices have to be run and maintained according to security guidelines. They are part of a function of the plant, which serves an industry-specific and business-specific purpose, which the owner and the operator of the industrial plant are to feel primarily responsible for. The (known) manufacturer installs a part of a different (perhaps even unknown) manufacturer, for the maintenance of which, relating to IT-security, in the end, nobody feels responsible for or cares about.

*All three examples* have something in common – the purchasers, owners, and operators of the IoT-devices are often unclear about the significance of IT-security. Why? Purchasers, owners, and operators are often not informed in a way that enables them to immediately recognize different security levels and to be sufficiently aware of possible implications. There are no logos and no labels that show which devices and systems differ in terms of better or worse IT-security. The specialists themselves also have difficulty making this difference.

*In the second example,* it also depends if the already provided patches (software-updating) have really been applied. The misuse of IoT-devices, e.g., for bot networks can only then be adequately made more difficult when the software updating is carried out extensively and practically all affected devices are updated.

*The third example* shows that there must be consequences if, e.g., hidden IoT-devices are not updated. Only in this case would the manufacturer of the plant have to point this out to the owner and the operator. And only then can it be expected that the owner and the operator would require to make the updating of the software a component of the contract and thus enforce it.

In this article, the outlined solution for a specific category of IoT-devices is oriented towards these three observations. However, it should be noted that for the elimination of the vulnerabilities, the updating of the software in no cure-all: it is no guarantee for the sufficient protection of systems, but an important prerequisite for this, because IT-systems, as a rule, are not perfectly secured in the sense of being free of vulnerabilities.

## 2. Approach to Central Computing

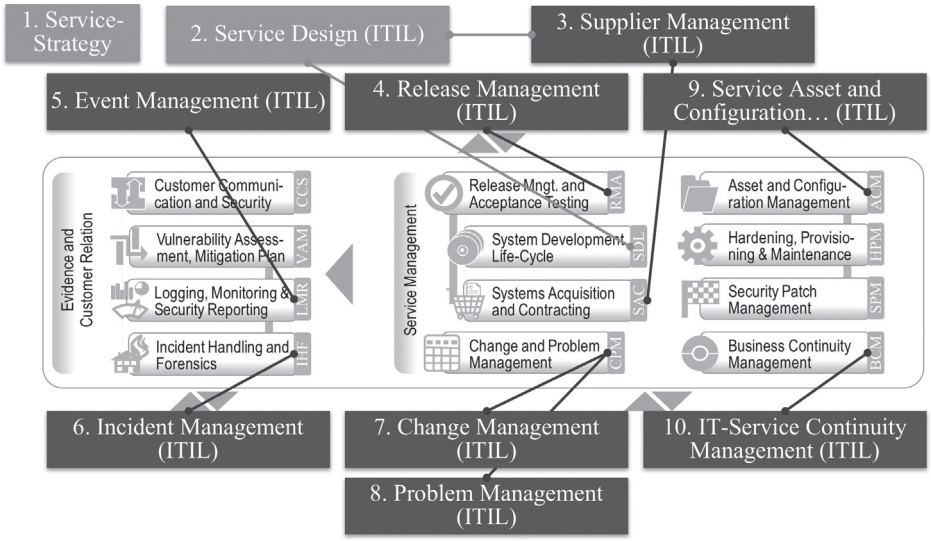
Before a solution is developed for IoT-devices, a look should first be taken at "conventional" IT-systems to examine how such challenges are addressed there. More precisely, it will be explained how this is to be done in an industrialized IT-production, as the latter can be assumed to have the highest level of maturity. Readers not interested in this extra voyage can skip this chapter.

In this regard, we will first define what is to be understood by an industrialized IT-production. The surface area of a data center approaches the size of a football field and houses some 2,800 racks with a total of nearly 40,000 physical servers (computer systems). Climate control, power supply and the like are not included in this figure. Such a number of systems calls for a large-scale data center. The IT-systems are run by an IT-service provider, who makes his IT-services available to his customers. Particularly when the customer is a large enterprise, the corresponding IT will clearly be very complex. Large-scale enterprises have very specific business processes. The support of such business practices through modern IT calls for certain requirements and solutions, which raise the complexity of the IT and TC of the IT-service provider. To assure quality and to keep costs under control, the provision of the IT-service is process-oriented and highly organized in a shared-task manner. Similar to automobiles of today no longer manufactured by a team of specialists, but by people trained only to carry out certain, simple tasks along the assembly line, the IT of this day and age is also industrially produced.

ESARIS and the *ESARIS Security Taxonomy* [9] enable that the IT-security be able to have command of such an IT-produc-

tion which is process-oriented and characterized by a high degree of division of labor. In the process, familiar measures are integrated. Hence, it has less to do with implementing measures such as access protection, encryption, monitoring, etc. (For this purpose, there are ample other sources such as [6] and [7]). On the contrary, a challenge is faced in possessing a method which ensures that hundreds and thousands of such security measures be defined, communicated and appropriately properly applied in an industrialized production environment with thousands and sometimes ten thousands of highly specialized employees in several countries around the globe [9].

**Figure 1 | Section from the ESARIS Security Taxonomy (12 of 31 areas)**




Because the IT-production environment is organized as a process-oriented manner, the ESARIS Security Taxonomy focuses approximately half of its activities on the development, implementation, and the operation of IT-services, including their care, resp. maintenance and further development. As the IT-production environment takes avail of a great number of technologies, the other half of the Taxonomy concentrates on the typical technology areas which also supports the division of labor within the IT-service provider and with all its partners and suppliers. At this point, only the activities in the development, implementation, and the operation of IT-services are of interest since this is all about the enforcement of IT-security in the broadest sense. Its characterization is mainly derived from established procedures, as defined in ITIL.[1] In ITIL, however, like also in ISO/IEC 20000 and other similar standards, IT-security on the one hand and the specifications of an industrialized IT-production on the other hand are not or only insufficiently taken into account. These are the reasons for the augmenting of existing best practices by the association *Zero Outage Industry Standard* on the basis of ESARIS. [11]

Figure 1 shows a section from the *ESARIS Security Taxonomy* with references to processes (from ITIL) and how they are established in an industrialized IT-production. It is notable that four areas of the *ESARIS Security Taxonomy* have no equivalent in ITIL. This underscores why a direct taking on of the ITIL-processes would not only have been too confusing, but also insufficient. Three of these newly included areas touch upon the subject of this article directly: the *Vulnerability Management and Mitigation Planning (VAM)*, the *Patch Management (SPM)* and the area of *Hardening, Provisioning and Maintenance (HPM)*. The *Vulnerability Management* is so essential for the subject of "security" that an additional area (VAM) had to be created. A further area that was added entails *Patch Management (SPM)*. Surprisingly, one only finds this key activity with difficulty in ISO/IEC 20000, resp., ITIL. SPM, however, is so important that it has to be precisely elaborated on in order that vulnerabilities be systematically eliminated and security gaps be sealed. The *Hardening, Provisioning and Maintenance (HPM)* comprises significant, practical guidelines for the realization of guidelines in a general life-cycle as described in ITIL-processes.

In the shaping and in the elaboration of the ITIL-processes through ESARIS, a complete picture of a life-cycle unfolds which thoroughly models and takes into consideration IT-security. The process and the connections are only outlined briefly. For further details, please refer to the cited literature.

What is particularly relevant for the subject of this article involves the following procedures (see Figure 1). The Life-Cycle Management provides the regular updating of the software (Patch). The basis for this are defined processes and procedural methods, which include the definition of patch-classes *(Security Patch Management (SPM))*. The patch-process starts with activities such as the collection, the quality assessment, the packaging and the scheduling for the implementation, followed by compatibility tests. This implementation is managed by *Change Management (Change and Problem Management (CPM))*. This process ensures the contingency planning and preparation of a roll-back, for risk mitigation and review and approval. At the end, the patches are implemented and the change is reported as accomplished. The patch-status is maintained and known within the Asset Management *(Asset and Configuration Management (ACM))*.

Several security patches follow this path because the transition between function and security is often fluid. However, there is a reason for the existence of the mentioned second area of Vulnerability Management (Vulnerability Management and Mitigation Planning (VAM)). For example, scanners are operated, which examine the systems in a targeted manner as to whether vulnerabilities exist. There are several, sometimes different sources that inform on the vulnerabilities; and penetration tests and forensic examinations provide further indications of vulnerabilities no matter of type.

The Vulnerability Management only performs an analysis and assessment and provides proposals for a further course of action (termed "mitigation planning"). What comes after? If there

1 IT Infrastructure Library; ITIL is a compilation of proven procedures for the implementation of an IT-Service-Management (ITSM) that meanwhile represents a de facto standard. The corresponding international standard for the ITSM is ISO/IEC 20000 [8].

is evidence of an imminent danger, the issue is forwarded to Incident-Management (IHF), even when the cause and the measure for remediation are already known because, for example, a patch already exists. The implementation is carried out again through the Change Process,[2] with the Patch Management Process being the final executive arm. If there is no imminent threat, the question is posed as to whether the solution for the elimination of the vulnerability is known. If a software update is available, this is of course the case. However, the Vulnerability Management also observes other cases. In the first case, all information is forwarded to Patch Management; in the second case, the solution may not be known at all, so that the case is transferred to Problem Management.

In this way, a large-scale industrialized IT-production is ensured that vulnerabilities are systematically detected and are eliminated by the updating of software. The IT-service provider can only then provide this service in the required quality when it is a component of his IT-service. If the customer, e.g., lets his own applications be executed on the infrastructure of the IT-service provider, the customer can then himself be responsible for the updating of the application-software, while the IT-service provider takes care of the updates of the infrastructure components only. On the contrary, it is, however, also frequent that the IT-service provider, possibly by the order of a third party, assumes the updating of the devices and systems, which are not in his possession and also not even installed in a data center. This also applies to IoT-devices, which are offered as so-called "Managed Services". IT-service providers indeed take avail of this vehicle to continuously eliminate vulnerabilities by means of updating the software.[3]

Business customers can infer from their contracts with the IT-service provider as to whether the updating is a part of the offered and contractually agreed on services. If they explicitly buy IT-service, they also avail themselves of the required expertise to assess corresponding evidence in the form of security reports, which the IT-service provider places at their disposal.

It is a different story for consumers and companies, who do not explicitly buy the IT-service at all, but implicitly apply components and systems, which actually (would) require the IT-service of updating. Therefore, a problem exists for the consumers, who fail to grasp the complexity of this matter and its consequences; and there is a problem for business customers, who operate the IT-devices and systems which they have no idea about or are only partly familiar with. The following outline solution is intended for both of these cases. It is to be understood as an idea and food for thought.

## 3. Ideas for Secure Devices in the Internet of Things

### 3.1 Eye-Catcher and Solution Outline

The following encompasses almost all of the entire solution description. However, it is described in greater detail later using GameTheory.

- **A train racing ahead can pose an extreme danger for those riding in it and for the environment when the train driver falls asleep or for other reasons fails to fulfill his duties. Therefore, there is a so-called dead man's handle installed in the train: the driver has to activate this device every 30 seconds. If he fails to do so, he is alerted by sound. Should he not react, an emergency brake is triggered to prevent an unfortunate occurrence.**
- **Don't we need such a device for the "Internet of things"? IoT-devices have their vulnerabilities. In the event that they are not eliminated by the updating of software, they can cause great damage, and even turn into a weapon. How would it be if these devices had a limited durability? Would the software be updated in this time or the durability extended? If not, the user first obtains a warning. Then, in case no response is made, the device fails to perform its duty or reduces its operation to the extent that no danger will be led to. The regular updating of the manufacturer leads to the elimination of vulnerabilities and extends the durability for a further period.**
- **We call this "life sign control". The user recognizes that security plus by this seal/logo! It signals: This device is always "fresh"; it is still preserved and state-of-the art – on the part of the manufacturer. This can be identified by anyone: without any technical know-how whatsoever, without any control of the software versions, etc. The user only sees the logo with the durability period and knows the manufacturer looks after it. The result? Only the regularly maintained "living" devices last in the Internet. For more security. For more quality.**[4]

It is important to note that this solution only solves the three problems identified in Section 1.2. This means the following. First: Consumers can differentiate between devices and systems of higher security than those of inferior IT-security. The affixed seal/logo signals this. Second: Similar to the eco-seal, the logo itself is no guarantee that the promise implied is actually kept. The manufacturer or service provider, however, is legally obliged and quality controls can be made at any time. As IoT-devices are mass-produced goods, the software updating is applied on a great number of devices. However, this still allows controls. In any case, this updating is carried out extensively and, as a rule, proactively, without the user having to take the initiative. If users express a preference for such devices with the sign/logo, the diffusion of the solution will be further increased.

Third: While case two for many computer systems and also for consumers is a standard one, the non-importing of the updated software, as a rule, does not lead to any consequences for the user or the manufacturer/service provider. However, because the dead man's handle, after a warning, obligatorily leads to the limitation

2 Typical "changes" are often compiled and planned in the form of a "release". All "changes" (normal and emergency changes) are implemented through the corresponding process.

3 As we always have a role-model perspective in mind, this naturally includes manufacturers offering updates as an IT-service.

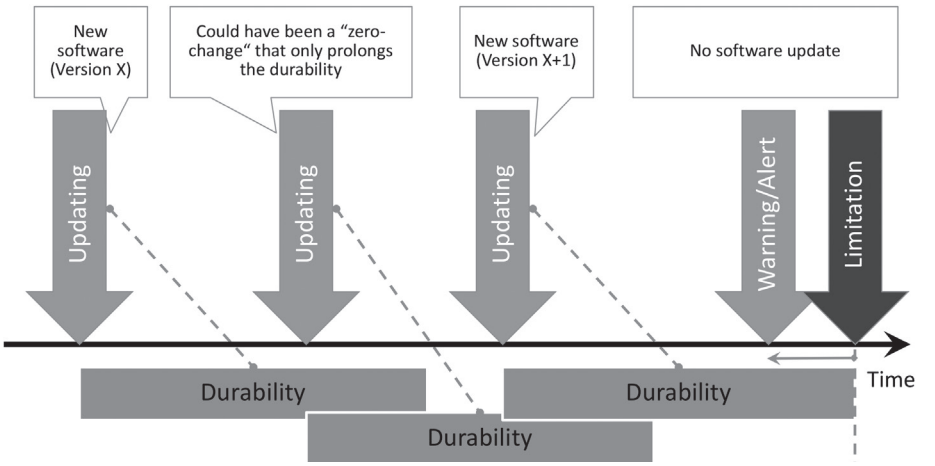
4 The term "durability" was first created at a press conference of the "Chaos Computer Club" though it was defined differently: *33C3: Hackers demand a minimum durability period for Internet-linked devices*; see: *33C3: Hacker rufen nach Mindesthaltbarkeitsdatum für vernetzte Geräte*; Heise online, 31 Dec. 2016, 2:40 p.m.

of the functionality, resp. to the complete de-activation, the consequences thus become clear. In this regard, it actually doesn't primarily have to do with devices failing to perform their service. Much more, it concerns the solution for it – that everything is being done to carry out the updating on a regular basis. According to the Game Theory, this involves the mechanism of "credible commitment" which ensures that those involved are not hindered from complying with their commitment by other, apparently more important things.

*Credible commitments* (also called precommitment or commitment devices) are "strategic moves" in a "sequential game". Such strategic commitments are intended to change expectations and behavior. In our case, the strategic move intends to have an effect both on the party giving the commitment ("meet it") and, more important, on the other market participants ("rely on the commitment"). For more detail refer to [1].

Naturally, the outlined solution is not employable for devices whose key factor is instant availability, such as, e.g., life-supporting systems. More on the limitations is to be read in Section 3.3. In the following Section 3.2, the details for the implementation are outlined.

**Figure 2 | Schematic representation of the function of the solution**




### 3.2 Implementation Information

The solution should not make high technical demands on the IoT-devices. The immediately following requirements behave independently of the implementation of the solution and are only required to be able to support the functionality of the software updating. As regards mass produced goods, this is not to be taken for granted, although even processor chip cards offer such features. Further below, the requirements are described that refer to the solution itself.

- The IoT-device has to possess the capability to load software patches and to update the software. In this regard, alongside the IT-functionality, a correspondingly larger, writable, non-volatile memory is also necessary that is capable of program execution. In order that interruptions, e.g., in the power supply do not lead to a defect of the IoT-device, the memory has to be large enough. However, the actual functions for the realization are known and state-of-the-art.
- The IoT-device has to have the possibility to be able to test the authenticity of the patch before the new software version becomes active. On the size of the memory, see above. The examination of the authenticity includes the proof of the data origin and the proof of integrity of the patch. Generally speaking, for this purpose, signatures or MACs are used, which do, however, require the capability of the IoT-device to carry out cryptographic operations with algorithms of sufficient strength and to be able to manage the applied cryptographic key.

The following functions are necessary in addition for supporting the proposed solution:

- With every software setup (patch) the IoT-device receives a date and consistently saves "time of the last update". The IoT-device has to be in the position to determine the "current time". For this purpose, it disposes of either an internal real-time clock or regularly establishes contact with a time server.
- The IoT-device regularly defines the difference between "current time" and the "time of the last update". If one uses Unix time, it is only necessary for the determination of "elapsed time" to subtract the two 32-Bit-long numbers from one another. A second such subtraction provides the comparison of "elapsed time" with the "durability period" saved in the device.[5]
- If the "elapsed time" is greater than the saved "durability period", the IoT-device must be in the position to reduce its functionality to such an extent that no danger can arise from it. If the "elapsed time" gets close to the stored "durability time", it would then be desirable for the IoT-device to be able to issue a warning.

This process is schematically depicted in the following figure (Figure 2).

The non-technical requirements include, among others, the following:

- The rules for the implementation of the solution and its associated implications have to be prepared very accurately and be publically available.
- There has to be a version of these rules which is also fully understandable to lay people. The sign/logo has to possess a sufficient recognizable and informational value.
- The sign/logo should also be made known. It signals that the party who uses this sign/logo for products or services is obliged to comply with the rules. (This also means that users and manufacturers take into account the defects of IoT-devices, should the manufacturer or the IT-service provider decide to no longer carry out the updating. This also applies even if it is no longer in the position, owing to, e.g., its business becoming insolvent.
- The sign/logo may also additionally convey the following information: type of warning (e.g., email, announcement, none) and length of the "durability period" (e.g., months, a month, weeks).

5 Although the difference is measured to the second, it probably really results in a matter of days. The "durability period" can be weeks or even months.

The technical requirements make the *credible commitment* really credible and finally enforce it. There are non-technical requirements in addition. They are often underestimated. Game Theory puts us right also here. The solution of the game strongly depends on the information being available to each player. Not knowing the strategy of the other player may cause oneself acting unpredictably (i.e. to apply a so-called mixed strategy [1]). The attempt to take advantage of the non-observability of my strategy usually leads to a result that is worse for both sides [3]. Therefore, one should try to develop the game towards one with *complete information* since in this case the preferred result can easier be achieved or becomes more likely.

Questions of a judicial nature are not the subject of this outline.

### 3.3 Limitations

It has to be pointed out that with the outlined solution there is a conflict of objective between security and availability. This is due to the fact that functional limitations occur in the event the manufacturer or the service provider fail to meet their commitment or cannot fulfill it to look after the updating of the software. This is the Achilles heel of the solution! This updating, however, is absolutely necessary, as pointed out further below. Therefore, there will be a class of IoT-devices for which this solution is not applicable! One could consider if it makes sense to offer a somewhat deviating solution for this case, the implementation of which, once again, would have to be indicated via the sign/logo.

Under software updating, it is not be understood that it is to be regularly changed. If there are no vulnerabilities which can or have to be eliminated by the manufacturer, there must be a "zero patch", which signals the IoT-device that it is at the latest level and the "time of the last updating" has to be updated only. However, this also means that the manufacturer/service provider, within the span of the "durability period", has to create and apply at least one patch.

The solution doesn't mean that the updating of the software has to be carried out fully automatically and without the involvement of the user. Here, quite different implementations may be carried out. For IoT-devices that are not serviced or maintenanced by the user, an automatic "push-service" is provided. For devices in which the user directly works with the IT-functionality, it can be beneficial to allow the user to delay the updating of the software for a certain period of time, e.g., if the updating is tied to a temporary limitation of the functionality or availability. This is, e.g., known to be the case with privately owned Windows-computers.

## 4. Discussion and Outlook

There are certainly more complex and more effective solutions to increase the security of IoT-devices. The one introduced here stands out because it doesn't require its own central infrastructure and its implementation is oriented towards the limitations of a large number of IoT-devices, like, e.g., embedded systems.[6]

This is why it should be kept in mind that the solution is only designed to solve the problem initially described in this paper: *With a certain class of devices, there is the risk that it is not identifiable if, first, the updating of the software is supported and, secondly, if it is actually carried out.*

It is based on the assumption that more complex solutions may not be implemented or only find use in a very limited area. Unlike in payment transaction systems, in which the evaluation of IT-security is prescribed and approvals are the prerequisite for the use, devices in the Internet do not represent a closed, but an open system. Furthermore, we are not dealing with a controlled, but with a free market in this context. Therefore, it was the aim to set the requirements as low as possible. This also naturally means that the effect is limited. It is hoped that the effect nevertheless is a radiating one and significant, should the solution ever come into operation.

A further assumption includes that is not smart to assume that good solutions are finally chosen over bad ones. The IT is more a "market for lemons", in which products of higher quality are rather driven out. This is particularly applicable in the segment of consumer products, but not only there. The "market for lemons" is a phenomenon described by George Akerlof (Nobel Prize for Economics 2001) arising from the assumption that consumers are less informed about the characteristics of a product (than the manufacturers) and thus tend to buy cheaper products even if they risk picking the "lemon" instead of the "orange". Refer to [2].

The way out of the dilemma (fewer lemons and higher returns) exists classically in the clearing up of the information-asymmetry: The customer is informed better, which, however, isn't the case in IT, because consumers do not have the needed know-how at their disposal to understand such information and its consequences and to draw the right conclusions from it. The sign/logo also primarily informs. But it can do much more! It signals (1) a guarantee promise and lifts (2) the manufacturer/service provider over the associated *credible, communicated commitment* to really take care of the updating. The credible, communicated commitment is reached through that, resp. strengthened to the degree that breaking the promise leads to serious consequences: The device is not as capable of functioning as expected and already paid for. This may enormously raise the level of pressure exerted on the manufacturer/service provider.[7]

Yet, what does the manufacture/service provider get out of this? What could his motivation for this be that he subject himself to these rules of the *credible commitment*? There are a great number of manufactures/service providers for whom the updating of the software is already included in the service. They will gladly grasp the idea of a sign/logo because it will not cause any additional effort for them. It will, on the contrary, make their potentially better performance more visible, because they are selling sweet "oranges". This visibility puts the producer of the sour "lemons" under pressure. The question is how the user makes the decision. Will they risk buying "lemons", while they could have had "oranges"? This question cannot be answered at this point. Too many factors influence both the purchase of IoT-devices and the dynamics of the installations.

6 Embedded Systems are systems (a) with a computer-functionality, which are, however, (b) not freely programmable, but built only for and supporting a very specific application-purpose, which, (c) as a rule, interact with their environment, hence exchange information, and (d) are specifically subject to structural-technical limitations. The latter-mentioned characteristic is of high importance here. IoT-devices often are mass-produced goods with a low price, small size and low operational cost being the key aspects.

7 It could also be considered to encourage state authorities to support this seal/logo, similar to the "Blue Angel" eco label in Germany, as both cases, in a broader sense, involve the protection of the public domain.

Which risk do the users accept and are they motivated to favor the "new" solutions? At first sight, the labeled devices have a basic fault. They fail to operate if the manufacturer/service provider is no longer prepared to be involved. But how likely is it that the market leaders, who dominate the purchasing decisions, will visibly break their once-made promises? The authors estimate the likelihood to be somewhat minimal. Of higher probability is the case of an insolvency or a change of the business model, which inevitably leads to the promise no longer being kept. Harm to one's reputation with repercussions for the follow-up business is rather not to be expected. At most, it could affect other areas of business. What has to be suspected from all of this is that the life of IT, as a rule, is limited and IT is subject to rapid changes with high renewal rates. In this context, it is absolutely intended, for security reasons, that obsolete, "forgotten" and no longer serviced devices be sorted out through the solution.[8]

On the other hand, users may accept a risk when buying IoT devices which have not such a logo. These devices obviously have a flaw right from the start; they are neither maintained nor improved. Moreover the user or operator of such devices takes the risk that the devices are abused.

However, the proposed solution is no cure-all. That has been stressed repeatedly. In particular, it is not applicable for all classes of devices. Roughly, one could put those devices or systems operating on the Internet into three classes and assign these varying solutions as follows:

- For devices, resp. systems with a high computing capacity, the described solution in Chapter 2 of a central, contractually agreed on updating of the software is to be preferred and to be regarded as adequate.
- For devices, resp. systems with a medium range of computing capacity, the new solution (see Chapter 3) is exactly the right one. These systems make for the largest group and fasted growing number of things in the Internet of Things. This is currently considered to be the main problem.
- For devices, resp. systems with a very limited computing capacity or with very high availability requirements, the first version disqualifies itself, owing to a lack of technical requirements. The second version may be realizable in technical terms, but the high availability requirements forbid it from being implemented with the new newly proposed solution provided here.

IT is complex. It is not bad if a solution doesn't heal the world. A solution suffices when it fulfills a clearly defined purpose and really keeps its promise. The present discussion paper is to be understood exactly in this context. The proposed solution of a "durability date" is to serve as food for thought as regards IoT-devices to stir the discussion about security in the Internet.

Decisions play a major role in IT security. The use of concepts of Game Theory can help to understand the decisions of participants and to shape the environment in such a way that preferred results become more likely or will even be enforced. The use of the concept of *credible commitment* (as a strategic move, [1]) and the conclusions drawn from the analysis of the lemon markets [2] are just examples. *Information* and *communication* play an important role. Game Theory helps to understand that these aspects are not just simple prerequisites. On the contrary, shaping *information* and *communication* can help to advance the game into a *game with complete information* and to obtain payoffs that make preferred behavior more likely [3]. The potential of Game Theory is, however, not exhausted with this. Mechanisms of *coalition games* [3] can help to understand the effect of agreements between the players. The study of *repeated games* with complex starting points (e.g. in the prisoners' dilemma) can show how payoffs and future prospects in form of controlled repetition and discounted payoffs can lead to more cooperation [4]. Finally, the fundamentals of Game Theory remind us that the adequate analysis of alternatives and the estimation of benefits are very important [5]. Although the payoff functions are usually constructed based upon assumptions and imperfect appraisals, the analysis of the correctly classified game mostly provides amazing clear and helpful strategic proposal.

8 The updating could also be bound on a warranty period. This would decrease the financial risk for the manufacturer due to the necessity to maintain "second-hand equipment".

## Literature


[1] Avinash K. Dixit and Barry J. Nalebuff: Spieltheorie für Einsteiger, Strategisches Know-how für Gewinner; Schäffer-Poeschel Verlag für Wirtschaft, Ulm, 1993; englisch title: Thinking Strategically; W. W. Norton & Company, 1993

[2] Ken Binmore: Fun and Games, A Text a Game Theory; D. C. Heath and Company, 1992

[3] Manfred J. Holler und Gerhard Illing: Einführung in die Spieltheorie; Springer, 7.Auflage, 2009

[4] Robert Axelrod: Die Evolution der Kooperation; R. Oldenburg Verlag, München, 1987; english title: The evolution of cooperation

[5] Morton D. Davis: Spieltheorie für Nichtmathematiker; R. Oldenburg Verlag, München, 1993; english title: Game Theory: A Nontechnical Introduction

[6] ISO/IEC 27002 – Information technology – Security techniques – Code of practice for information security management

[7] Information Security Forum (ISF) – The Standard of Good Practice for Information Security 2016

[8] ISO/IEC 20000 – Information technology – Service management – Part 1: Service management system requirements, Part 2: Guidance on the application of service management systems

[9] Eberhard von Faber and Wolfgang Behnsen: Secure ICT Service Provisioning for Cloud, Mobile and Beyond, ESARIS: The Answer to the Demands of Industrialized IT Production Balancing Between Buyers and Providers, 2017, ISBN- 978-3-658-16481-2, XIV + 368 pages – 2. UPDATED and EXTENDED edition

[10] Eberhard von Faber: Organisation der Absicherung einer industriellen IT-Produktion, Drei Handlungsfelder jenseits von Protection, Detection, Reaction; Datenschutz und Datensicherheit (DuD), Heft 10, 2016, Seiten 647-654

[11] ESARIS Security Taxonomy – Synopsis, Scope and Content; Zero Outage Industry Standard, Release 1 about Security, February 2017, https://www.zero-outage.com/security


Translation of: DOI 10.1007/s11623-017-0808-x
Journal: https://link.springer.com/journal/11623
Article: http://link.springer.com/article/10.1007/s11623-017-0808-x