OpenTC Newsletter
April 2009

From the Open Trusted Computing (OpenTC) research project, sponsored by the European Union.

In this issue:

Editorial: Fighting theft of confidential information

By: Arnd Weber, ITAS, Forschungszentrum Karlsruhe, Germany

Dear Reader,

It has been a while since we sent out the last newsletter. We are making up for the wait with three interesting articles:

We start with an interview that addresses how a secure hypervisor can help to protect against economic espionage. In “A hypervisor against ferrying away data” Chris Dalton from Hewlett Packard Laboratories, UK, not only addresses the future development of threats, but also how a hypervisor architecture as produced by OpenTC could help, what progress has been made in the project, and what the remaining issues are for the future.

In addition, we present work on virtualisation for small, mobile devices. In “A Secure Wallet for a mobile Trusted Platform” a mechanism is presented for keeping user authentication data protected, such as passwords. The article is written by Eckhard Delfs, Comneon, Germany, Eimear Gallery, Royal Holloway, UK, David Jennings, Infineon, Germany, and Hans Löhr, Ruhr University, Germany.

OpenTC project partners IAIK (Graz University of Technology, Austria), Portakal Teknoloji (Turkey) and the University of Cambridge (UK) lead the creation of the Java Standard API for Trusted Computing, the Java Specification Request # 321. Now a draft of the specifications is openly available for review and comments are asked for. See the article by Ronald Toegl of IAIK on “Standardizing a Java API for Trusted Computing, the open way” to learn more.

Let me take the opportunity of writing this editorial to announce a Common Criteria V3.1 Protection Profile for a “High Assurance Security Kernel”. This document can be used to evaluate future kernels, for instance, products using the OpenTC architecture. It was produced by the OpenTC partner Ruhr University, together with Sirrix AG, atsec information security and the German Federal Office for Information Security (BSI) (see [1] and [2] for details).

As usual, we close with an announcement of new research papers published by members of the OpenTC consortium. It is a long list because it covers nine months. The reason why you had to wait for this newsletter is that we are preparing the publication of our final proof of concept demonstrators – stay tuned!

References:

[1] Sirrix AG: *High-Assurance Security Kernel EAL 5 Protection Profile, according to the Common Criteria v3.1 R2, 2007, certified by the Federal Office for Information Security (BSI).* 2008. www.sirrix.com/media/downloads/54500.pdf
[2] Löhr, Hans; Sadeghi, Ahmad-Reza; Stüble, Christian; Weber, Marion; Winandy, Marcel: *Modeling Trusted Computing Support in a Protection Profile for High Assurance Security Kernels.* TRUST 2009, April 2009.

Contact: arnd.weber (at) itas.fzk.de

Acknowledgements: Our thanks go to Alison Hepper, Dirk Kuhlmann and Herbert Petautschnig for their help in preparing this issue.

A hypervisor against ferrying away data

Interview with Chris Dalton, Hewlett Packard Laboratories, Bristol, UK

Which threats can be countered by the kind of hypervisor being built by the OpenTC consortium? This is first question addressed in the interview, which subsequently discusses how these threats can be reduced, what progress has been made here in the project, and what the remaining future issues are. Chris Dalton from Hewlett Packard, who worked on the project from the beginning, was interviewed by Franco Furger and Arnd Weber from ITAS, Forschungszentrum Karlsruhe, Germany, on January 27, 2009, in Bristol, UK.

Arnd: Do you think there are certain risks or attacks which have already occurred that could be addressed with a hypervisor-like approach?

In general, I think the answer is “yes”. Code is always going to have bugs that can be potentially compromised. The big thing that virtualisation brings is the ability to compartmentalise, to contain environments. This puts boundaries around that code. You accept that some code can be compromised, but you have, if you like, a ring-fence around it to defend against those compromises. An example would be a platform with Vista running on top of the virtualisation layer. Even if somebody manages to compromise Vista, which is likely, given its size – 50 million lines of code I think – there is a chance that you could have supplementary controls, within the virtualisation layer itself. Even if Vista is compromised, those controls are still effective.

Arnd: If there were an attack on Vista, how would virtualisation help?

Some of the attacks that we are aware of involve compromise of an end-user’s platform. Data from that end-user system are then ferried away to a remote location.
If you had network controls that were outside Vista, but enforced by the virtualisation layer, they would then have to compromise both Vista and your virtualisation layer. In that regard –

as a simple example – virtualisation could be quite useful for to setting up a network policy which remains in place even if Vista is compromised.
Moving forward, users want to use applications. They do not care so much about the actual operating system. For example, if you are using a web browser, I don't think you particularly care whether it is running under Windows or Linux. Similarly with something like iTunes, it would be nice if it were running on Linux as well as Mac/Windows. The user is interested in the application, not the operating system. With virtualisation, you could provide the applications separately. For example, you could have one virtual machine running your iTunes application, you might have one running your web browser, and you might have your office suite. In that way again, you manage to partition the amount of code that is in any particular environment. Hopefully, then you can contain any damage if your office suite is compromised.
The big challenge – and this is something we have been struggling with at HP for a number of years – is that isolation itself isn't that useful. You need isolation plus controlled sharing, and I think this is still an issue. At the moment with mainstream operating systems, at first everything is wide open and then controls are put in. Virtualisation is a better approach, where at the start "nothing is allowed" and those controls are selectively relaxed. I think that's a better starting point. Unfortunately, you may end up in the same place. To me this is a research question, but potentially isolated environments running the applications that people want to use with controlled sharing is a more robust starting point. But then we have to worry about ease of use. That's the killer.
One of the reasons that Vista has grown so big is because Microsoft is trying to satisfy users' expectations, i.e., give them the experiences they want. When we introduce security controls, we can't afford to compromise those user experiences: that's a challenge.

Franco: Is this feasible on a technical level? Do you need a separate OS for every set or family of applications? If you have 20 applications, that sounds like you need 40 GB of RAM?

You end up with groups of applications within a VM. You do not necessarily need to run them all at the same time. You might have, say a Skype application, which you only need when making a call.
Certainly on the server side over the past few years, hardware has started to accommodate the needs of virtualisation. For example, the number of memory chips you can put in has increased. The hardware is evolving to meet the needs of virtualisation. On the client side, that will also happen.

Arnd: You mentioned that "data were ferried away". I remember hearing on the news that there were organised crime attacks on home-banking solutions, and I have read various reports of a kind of organised crime where a competitor tries to get hold of confidential company data. Are you referring to one of these two or to a third option of which I am unaware?

I am certainly referring to the first two. I can give you some public examples (see references). From the references, it seems that state-organized industrial espionage is something to be concerned about too, though it is less of a concern for an individual using their home banking.

Arnd: Can you close the channel back to the espionage organisation?

In general, it seems that the attacks are quite specific, for example targeted against a particular Windows vulnerability. The problem with the current security solutions is that they typically only work against those specific attacks.

With virtualisation, our feeling is one can deal with more general classes of attacks as opposed to specific ones. Let us take the network controls as an example: Windows might have a particular vulnerability that allows data to be sent over a particular network port by exploiting a particular application. There may be a patch that patches that application. But with the controls on the virtualisation layer, the user is immune from this.
There are different classes of attacks that virtualisation is much better at preventing than currently existing security technology, which is more reactive. The latter copes with known threats and attacks, whereas virtualisation provides a certain degree of ability to cope with unknown attacks and threats. The general class of attacks may be known, but not the specifics. I think this is where virtualisation has an advantage in terms of security.
The downside is that if virtualisation becomes more prevalent, people will start attacking the virtualisation layer. In principle, the virtualisation layer can be quite small and simple. If you look at lines of code and bugs per line of code, it should be more robust and more easily maintainable, in theory.
In practice, current virtualisation layers are still pretty big. Hopefully that will change.
What we have seen in the virtualisation space is that there has been more concentration on virtualisation features in general rather than security.
Because the threats are not well known at the moment, or because people do not care about the current threats, it may be that more can be done within the main operating system than what is done at the moment, and this might be effective for a short while. Under, say, XP there were a number of incoming ports opened and, for a few years, there were a number of attacks based on that. But still, ultimately, having independent controls around an operating system is a good thing. I still do worry about the robustness of that underlying layer.

Arnd: Now that the OpenTC project is almost over, how far did we get in producing a relatively secure hypervisor?

I think the project has reached a good understanding of how a trustworthy or trusted virtualisation layer should look in terms of privilege separation, maintaining manageability, for instance, and, theoretically, which security services are required by a guest virtual machine. Where there is still work to be done is on the underlying robustness of the particular implementations on the one hand, and the practicality of implementations on the other. If we look, for example, at L4, it has good robustness and is a worthy placeholder for trust. The problems are its practicality in terms of the guest operating systems it can support, e.g. Windows, and also the overall manageability of an L4-based system. Enterprises have their own standard management tools and expect to be able to use them; they are not going to run under L4, or not going to run easily.
XEN, the other implementation choice, on the other hand *is* usable, since it supports Vista and Win 7, for example. With its domain 0, it has a management partition where standard management tools can run. As an underlying code base, however, it is still pretty huge.
There is still some distance to go. Theoretically, we have an architecture that should be able to merge the two, but in practical terms the OpenTC project has not managed to implement this.

Arnd: What is the problem here? Improving XEN or making L4 more manageable? Or to merge the two?

In the L4 case, there is other work going on outside OpenTC to make it a more general platform. It may be just a matter of time for this to happen.
On the XEN side: It is large, because of the feature set, and it needs to be competitive in the marketplace. L4 does not really have that problem, because they do not have a large share of

the virtualisation market. People expect XEN to have all the features of VMware, which means more code, and more potential vulnerabilities.
There are some research angles surrounding a modular hypervisor. An example for XEN might be a pretty static environment which does not need migration functionality: don't have that code on the system. By restructuring in this way, people could have a cut-down system. On the plus side, and this will benefit both L4 and XEN, we are seeing more features being provided in hardware that allow a reduction in the complexity of the hypervisor. We have already seen the early Intel VMX extensions, and AMD have similar extensions, that allow unmodified operating systems to be run without the need for all the complicated code that VMware had to use – they had to do binary re-writing. With modern hardware, this is unnecessary. For example, around memory management for guest virtual machines, there is a lot to reduce the complexity.
Having said that, for a commercial product such as XEN, there is the problem of supporting old hardware, which is quite a big issue. If we said that all the hardware that will be used had to have certain features, that would allow us to take out a whole chunk of code.

Arnd: But if we are thinking of an OpenTC-like hypervisor, users will have to use hardware with a new TPM anyway?

But the problem from OpenTC's perspective is that the project has been working the mainline XEN source tree and has not made any large deviations from that. You could take the XEN source tree and cut it to match the exact hardware being used by OpenTC, but OpenTC has not done that. The difficulty in taking the XEN tree and cutting it down is: Who is going to maintain it? If we take the standard tree, maintenance is taken care of. I think it is too much for OpenTC to take on the maintenance of their own version of it.
This is the main problem for OpenTC: We have a theoretical architecture, but no real means of supporting that practically in an ongoing fashion. Our prototypes are very useful in terms of putting systems in front of people and saying: What do you think about this? But in terms of maintainable systems, it is not practical for OpenTC to support them.

Arnd: What are good ways to have a future virtualisation layer on PCs? Pursue the TC idea, pursue the open source idea, or is a third path more realistic?

Both of the first two constitute the right approach. My hope is that, as more threats emerge, more emphasis will be placed on security in general. There will be more willingness to accept the re-architecting of virtualisation layers. The problem is that proper implementation of the architecture developed by OpenTC would require large-scale restructuring of something like XEN.

Arnd: In the German media, there have been reports of Asian competitors hacking into corporate systems. Do too few people take the threats seriously? Is this a motivation for Citrix and the developers of XEN? Are the attacks small in economic terms?

I think people will become more aware of the threats out there. To take XEN as an example, there is also a XEN branding issue. If XEN becomes the Microsoft Windows of the virtualisation world, they are going to worry more about their system being compromised. They don't have that to worry about at the moment.

Franco: Do you expect this change to occur fairly soon or might it just go on as it is today for years to come?

In the next 2 or 3 years.

Franco: Are you saying we need some kind of catastrophic attacks?

I think more attacks are occurring than we hear about. And I think we can expect this to get worse. If you look at some of the developments in the US (see reference), you see that the US government is starting to tackle this at a national level. You can expect this to take place in the Western world, too. The problem is that, by and large, companies are stuck with the products that are out there that they can buy. OpenTC is research, XEN is an open source project, and it takes time to move research results into actual products. Certainly over the next year or so there are not going to be any products out there that people can buy that could really help. Certainly over the next 2 or 3 years, I think people are going to become gradually more aware of the threats, and what they should be protecting against.

Franco: When you say "people" do you mean IT security people?

It certainly has to start at the top. It has to be the board.

Arnd: We have sometimes discussed the issue of runtime protection. How do you view this issue today?

What TC provides is a set of properties that are verified at boot-time. This is very useful – to establish that your trustworthy hypervisor is running. Trusted Computing gives you that first anchor.
There are more mechanisms that can be built into software that would allow you to do run-time integrity checks. But these are not going to be tied into the TC hardware. Since the hardware is merely a simple, essentially recording chip which costs less than 1 US dollar, it does not do very much. Hardware mechanisms like Intel TXT exist that attempt to build on this. But even so, you still end up with an initial software running on the system and are dependent on that software doing the right thing. It is very hard to get guarantees from the hardware that the software is going to do the right thing. I don't see that changing.

Franco: Assuming the software is compromised after an initial check. At the next boot, would it be possible to determine that the system has been compromised?

Yes, the problem is, it might be compromised again while it is running. You can always get your system to boot up in a known good state. Like a server platform, it might be running for months.
TPM and associated mechanisms allow you to record measurements of a known good state. Some of that state may change legitimately, some may change illegitimately. It is very hard to distinguish between the two in terms of data. In terms of binaries, it is easier. Let us assume, there is a version of Office on it which should always have the same measurements every time you boot. But if you start to include data in your known good state, you may have a macro-virus in there. Trusted Computing would not be effective to measure it. You cannot just measure all your data, it would become too inflexible.
Trusted Computing is a good foundation for getting your system up and running. You then have to apply additional mechanisms, from your operating system or hypervisor, to enforce runtime security.

Arnd: What other research issues remain now that the research project is almost over? For example, the potential weaknesses, Trojans etc. in the hardware, problems with the methods

or evaluation methods used, or the graphics output which we try to control. What are the big open issues?

All of these. The big open issue is moving from a theoretically strong architecture to a practically strong architecture, without compromising user experiences. This may include user experiences surrounding performance and resource usage or supported functionalities. Can you still run Windows?
In terms of the graphics, the work that HP here and Cambridge University are doing around Gallium will give us a good answer on how to convey trust to somebody on the screen. That has been quite successful for OpenTC. It was a big issue when we started. The challenge was performance, giving users fancy 3D-graphics, and at the same time confidence in what they are seeing is what they are getting. I think the big challenge is moving OpenTC into something that people can really use, commercially buy, and get access and support for.
Research-wise: A lot of hardware support allows a simpler software layer. That pushes trust into the hardware. Of course, there is less visibility into the hardware: that is an issue. I don't know what we can do about that. In general, hardware tends to have better quality control than software, I guess primarily it is harder to fix hardware bugs once it has been released. Pushing things to hardware may be a good thing, but how do you build people's confidence in that hardware other than saying: "That's Intel. I trust them", or "That's AMD. I trust them".
We have talked about theoretical architecture, but practical implementation of, say, XEN that cannot follow the architecture for commercial reasons, is an open issue. They know the code is big. They know it should be smaller. But how do you do that? They want to be widely used. We know that the implementation of the underlying hypervisor is not perfect, but that will happen over time. The most important thing that OpenTC has done is to have developed scenarios that can be tested with people, for example the PET prototype (see newsletter of January 2008), and the Virtual Data Center, and whether different customers/users see value in these scenarios. Assuming that the virtualisation layer underneath is strong, i.e., robust and trustworthy, are these particular scenarios of interest to you? That then motivates the strengthening of the underlying technologies. Without any scenarios, any customers or enterprises, asking for the technology, I don't think it is going to develop by itself, in particular for scenarios where isolation is important.

References:

[1] CBS 3: *Researchers: Chinese Cyberspies Strike In Canada. Computer Break-Ins May Have Targeted Tibetans In Exile.* March 28, 2009.
http://cbs3.com/topstories/china.canada.computers.2.970521.html
[2] Security Service MI5: *Espionage.* http://www.mi5.gov.uk/output/espionage.html
[3] Senate Committee on Homeland Security & Governmental Affairs: *Lieberman and Collins Step up Scrutiny of Cyber Security Initiative* (Press release of May 2, 2008).
http://hsgac.senate.gov/public/index.cfm?FuseAction=PressReleases.Detail&Affiliation=C&PressRelease_id=a32aba11-4443-4577-b9a5-3b2ea2c2f826&Month=5&Year=2008

About the interviewee: Chris Dalton is a Principal Research Engineer at Hewlett Packard Laboratories, based in Bristol, UK.

Contact: cid (at) hp.com

A Secure Wallet for a mobile Trusted Platform

By: Eckhard Delfs, Comneon GmbH, Nürnberg, Germany; Eimear Gallery, Royal Holloway, University of London, UK; David Jennings, Infineon Technologies AG, Munich, Germany; and Hans Löhr, Ruhr-Universität Bochum, Germany

Introduction

The ubiquitous adoption of Personal Digital Assistants (PDAs) and mobile phones with ever expanding feature sets is expected to effectuate an increased use of such platforms for security sensitive applications such as online banking and e-commerce. This makes them future targets for types of crimeware known from the PC environment, where so-called phishing attacks already pose a prominent threat. For such an attack, phishers attempt to fraudulently acquire sensitive information, such as usernames, passwords, PINs and credit card details by masquerading as a trustworthy entity in an electronic communication. To make things worse, mobile Internet users will have to remember an increasing number of passwords for different electronic services. This tends to result in users forgetting their passwords, choosing weak passwords, or using common passwords for different sites.

In order to address this problem, mechanisms like federated identity management and web wallets have been suggested, and, in this article, we introduce the "*Secure Wallet*", a mechanism designed and developed within OpenTC project's Work Package on "TC for embedded controllers in mobile phones" (WP8) to protect a user's authentication data.

The *Secure Wallet* is used to store user authentication data and to automate web logins. When a user visits a website where the wallet has not yet been used, the *Secure Wallet* chooses a strong random password on behalf of the user which is unique for this site. This ensures that if a password associated with one service provider is leaked it cannot be used anywhere else. Since the actual passwords are not displayed to the users, they cannot accidentally reveal them to phishers. Moreover, the *Secure Wallet* checks the SSL/TLS certificates of the site hosting the services and only transfers passwords to those with a valid certificate. Even the web browser itself never obtains a password from the *Secure Wallet*. To prevent misuse of the automated logins, it is important that only authenticated users can use the *Secure Wallet*.

In order to ensure a secure implementation of the *Secure Wallet* mechanism, both *isolation* and *secure storage* must be provided by the system. Isolation is necessary to prevent unauthorized processes, for example, the browser or any other application, from accessing the *Secure Wallet*'s data. Without isolation, unauthorized processes might be able to read the memory of the *Secure Wallet* directly, thus circumventing any policy enforced by the wallet itself.

Secure Storage is necessary to protect the data stored by the wallet. It must be ensured that only trusted applications, in this case, the *Secure Wallet* application, are able to access and read the stored data. Moreover, offline-attackers, for example, in the case of a stolen phone, must not be able to circumvent this protection, for instance by booting an alternative system.

Enabling technologies: Trusted Computing and virtualization

The key enabling technologies for the provision of isolation and secure storage are *Trusted Computing* and *Virtualization*.

Trusted computing, as currently defined by the Trusted Computing Group (TCG), is built upon five fundamental concepts: an *integrity measurement* (i.e. the cryptographic digest or hash of a platform component*)*, *authenticated boot* (where the integrity of a pre-defined set of

platform components are measured and the resulting measurements condensed and reliably stored), *secure boot* (where a pre-defined set of platform components are reliably measured, the resulting measurements verified against their expected values, and stored), *platform attestation* (where the state of the platform is reliably reported), and *protected storage*, upon which we focus. It should be noted, however, that we do not limit the definition of "Trusted Computing" to that specified by the TCG: in the mobile space, hardware manufacturers are currently employing feature enhanced chips which provide comparable platform security, but which are not directly related to the TCG specifications. Through the work of the Open Mobile Terminal Platform forum, stakeholders from the mobile phone value chain are in the process of standardizing security system requirements for mobile platforms.

First of all, we consider TCG technology.

The TCG's Trusted Platform Module (TPM) is central to the implementation of a TCG-compliant trusted computing platform. The TPM specifications describe a microcontroller with cryptographic coprocessor capabilities that provides a platform with a number of special purpose registers for recording platform state information; a means of reporting this state to remote entities; secure volatile and non-volatile memory; random number generation; a SHA-1 hashing engine; and asymmetric key generation, encryption and digital signature capabilities. The current documentation from the TCG also encompasses a vast set of specifications ranging from those relating to trusted personal computers, server systems, and to specifications for trusted networking. The TCG Mobile Phone Working Group (MPWG) has published the TCG Mobile Trusted Module (MTM) specification (namely, a TPM designed for a handheld mobile device) which enables the development of a Trusted Mobile Platform (TMP). There are various ways of implementing an MTM on a mobile platform. It could be based on a discrete TPM component, but there is also the option of running a software based MTM on a CPU within a chip [1].

It is assumed that a mobile platform will typically contain multiple MTMs to support multiple mobile device stakeholders. More specifically, two types of MTM have been defined a *Mobile Local-owner Trusted Module (MLTM)* and a *Mobile Remote-owner Trusted Module (MRTM)*. A *MLTM* supports *authenticated boot*, *platform attestation*, and *protected storage* and is controlled by an entity with physical access to the platform. An *MRTM* also supports *authenticated boot*, *platform attestation*, and *protected storage*. It moreover enables a remote entity (such as the device manufacturer or network operator) to predetermine the state into which some parts of the phone must boot (*secure boot*).

A facet of the Trusted Computing protected storage mechanism, namely sealing, is deployed in order to support a secure implementation of the *Secure Wallet*. *Sealing* is the process of encrypting data so that it is only a particular TPM/MTM can decrypt it when the TPM/MTM host platform is in a particular software state. The ability to seal data to a specific system configuration enables the *Secure Wallet* to protect authentication data against unauthorized access.

Virtualization enables the creation of separate execution environments on a platform, called *compartments*, where applications or even complete legacy operating systems can be run isolated from each other. Within Work Package 8 isolation is provided through the deployment of a security kernel based on the L4 microkernel and a resource management layer. Thus the *Secure Wallet* is isolated from the browser and other applications, which are running in a separate compartment with a full Linux operating system.

A trusted execution environment (which supports isolation and secure storage) has also been defined by the Open Mobile Terminal Platform (OMTP) organization [2]. The OMTP is an operator-sponsored forum which aims to serve all stakeholders in the mobile phone value

chain by gathering and driving requirements for such an environment. The requirements are technology platform neutral and aim to promote the adaption of new services across a range of platforms. A set of requirements defining a Trusted Environment profile (TR0) was defined in 2006, and more recently (mid 2008) an extended profile TR1 was published to address requirements for an Advanced Trusted Environment. These recommendations were defined independently of the TCG Mobile Trusted Module (MTM) specifications. The OMTP TR1 security enablers are of particular relevance, when considering a robust implementation of a secure wallet solution based on an integrated MTM.

Flexible Secure Boot (FSB) requirements ensure the integrity of the ME (Mobile Equipment) code base at boot time, and also define requirements for securely updating the code image via an external connection to the ME. These requirements are within the scope of a defined set of threats. FSB is a mechanism rooted in hardware, initiated during mobile phone initialization, which hierarchically verifies the integrity and authenticity of all relevant software components prior to its execution. In terms of platform security *Flexible Secure Boot* is a fundamental requirement to enforce a system booting into a specified state. All security which relies on software depends on a secure boot of that software.

The *Trusted Execution Environment* (TEE) ensures an environment for executing security sensitive programs. It meets a defined set of security requirements, resists a defined set of threats, and within this scope ensures that a program executes as it was designed to execute.

A *Secure Storage* facility helps the TEE to handle security sensitive objects. It stores the sensitive objects such that the security properties of the objects are maintained within the scope of a defined set of threats. For this purpose it makes use of the hardware unique key, an immutable secret key which is used as a root key for the *Secure Storage* and protected within the TEE.

A Secure Access to User Input/Output Facility (SUIO) controls the input and output of data between the user and the TEE where a trusted application is executing. SUIO assets are only required to be defending against software attacks. The IO facilities here could be a keyboard, touch-screen, display etc.

The basic platform security for OMTP TR1 is based on a secure boot. In this project, we used the Infineon X-GOLD^{TM} 208 baseband controller [3] as a mobile platform, which can also provide general platform integrity through its secure boot feature. One of the most important security enablers of TR1 is the TEE. The *Secure Wallet* prototype architecture can be viewed as a set of communicating Trusted Execution Engines. The prototype also provides both a Secure Storage and Secure Access to User Input/Output facility which is within the scope of an OMTP TR1 profile 1 classification. The secure storage is made possible due to the hardware cryptographic functions and a unique key provided by the X-GOLD^{TM} 208 hardware.

Secure Wallet deployment scenario on X-GOLD 208

Now we briefly describe the architecture of a possible realization of the *Secure Wallet* application on the Infineon baseband controller X-GOLD-208. This is an EDGE enabled multimedia enhanced component featuring an ARM926 microcontroller, various hardware peripherals and in particular a set of hardware security features, including secure boot, a crypto engine [3] with acceleration for SHA-1, AES, RSA, a Random Number Generator, and a hardware unique key.

At the lowest software level of the *Secure Wallet* prototype, the L4 microkernel provides only those services which need to be executed in privileged mode. These are services related to threads, address spaces and inter-process communication. The ARMv5 architecture provides memory virtualization via a Memory Management Unit (MMU). The processor core itself

supports seven modes, six of which operate in privileged mode and one in user mode. These features enable the microkernel to provide *isolation* properties, and in particular it can assign temporary ownership of selected hardware to L4 tasks.

The L4 environment layer is built on a set of servers which manage basic system resources such as memory, tasks and I/O resources. To facilitate a convenient programming environment, various libraries enable L4 applications to make use of these resources. On top of this environment, a set of security services provide security functionalities to user applications. The *Secure Wallet* and TPM emulator [4] are realized as standalone L4 applications in this architecture proposal. The TPM emulator contains a dedicated Secure Storage facility in order to securely protect its non-volatile data. The TPM emulator implementation in our prototype exploits security hardware extensions of the X-GOLD-208 baseband controller.

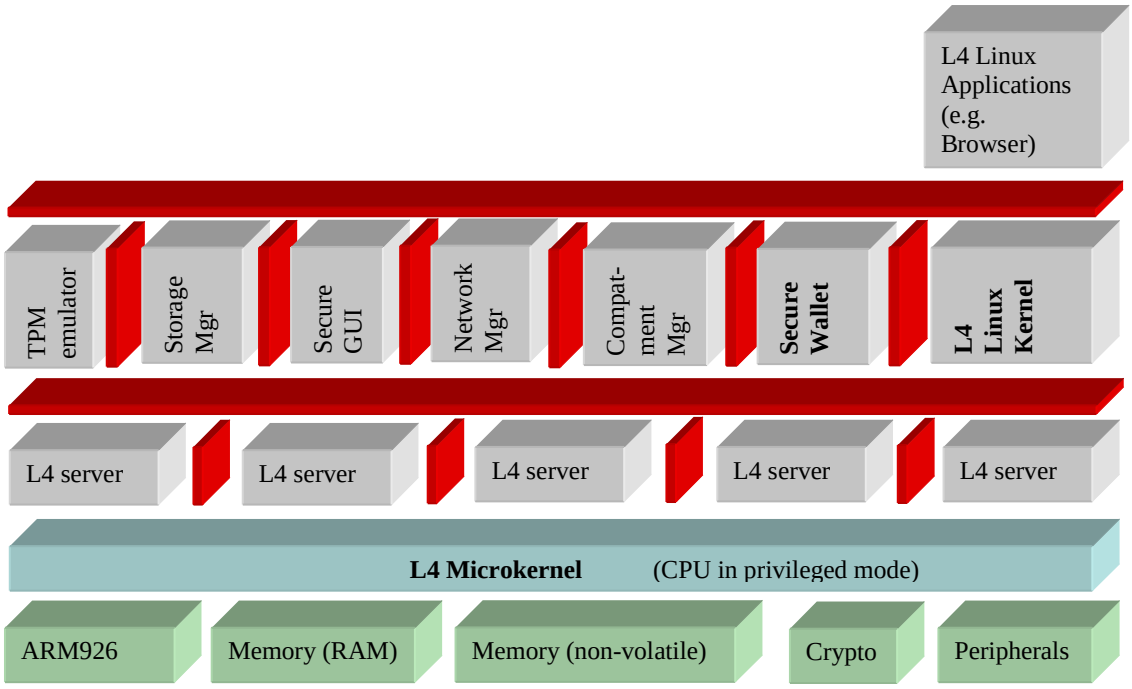

Figure 1: Possible architecture for a secure wallet application on an XGOLD-208 baseband controller

The Compartment Manager is responsible for measuring, starting and stopping compartments such as the Secure Wallet or L4Linux compartments. The Storage Manager provides compartments with secure storage which can be bound to integrity measurements to prevent modified (potentially malicious) compartments from accessing stored data. For this, the Storage Manager relies on the sealing functionality as provided by the TPM emulator. The Network Manager connects different compartments via a virtual network. In our application scenario, the *Secure Wallet* acts as a proxy for the browser's connections. The Secure GUI provides a trusted path from the user to an application. For example, it is important for the security of our solution that the user always knows if she interacts with the untrusted browser or the *Secure Wallet*. The Secure GUI prevents the browser or any other application spoofing the interface of trusted applications.

In OpenTC WP8, we are currently working on a demonstrator, where only a part of this architecture is running on a mobile platform. For prototyping, other parts, such as the *Secure Wallet* application, are running on conventional PC hardware. Moreover, the prototypes for the secure wallet and some security services are implemented as minimally configured L4Linux compartments instead of native L4 applications. More details about the architecture and proto-

type can be found in the OpenTC deliverable D08.2 “Study and Prototyping of a Trusted Mobile Application” [5], which will be published on the OpenTC website in 2009. Standards by both the TCG and OMTP have been taken into account in our work.

Conclusion and outlook

Our preliminary results indicate that the *Secure Wallet* seems to be a reasonable approach to protecting authentication data for web logins. Furthermore, our experience with the prototype shows that it is possible to implement the *Secure Wallet* on mobile platforms, although the current implementation is still partially based on a PC. However, some workflows require a different user behaviour than web authentication without the *Secure Wallet.* For instance, the user has to switch via the secure user interface between browser and wallet to set up a new account. Therefore, a user study (e.g., with a PC-based prototype) is needed to confirm the actual usefulness of our approach. Moreover, a major disadvantage of the *Secure Wallet* is that the protected authentication data can be used only on one dedicated device. To address this issue, we are currently working on a proposal for trusted migration of wallet data to other platforms.

References:

[1] Trusted Computing Group: *Mobile Trusted Module Specification FAQ,* June 2007, https://www.trustedcomputinggroup.org/specs/mobilephone/MTM_Specification_Technical_FAQ_062007.pdf
[2] Open Mobile Terminal Platform, www.omtp.org
[3] X-GOLD™ 208 - PMB 8877, www.infineon.com
[4] Strasser, Mario: *Software based TPM Emulator,* ETH Zurich, http://tpm-emulator.berlios.de/
[5] OpenTC Project Deliverable D08.2: *Study and Prototyping of a Trusted Mobile Application.* Report (October 2008), to be published on http://www.opentc.net

Contact: eckhard.delfs (at) comneon.com; e.m.gallery (at) rhul.ac.uk; david.jennings (at) infineon.com; hans.loehr (at) trust.rub.de

Acknowledgements: Our thanks go to Peter Lipp.

Standardizing a Java API for Trusted Computing, the open way

By: Ronald Toegl, IAIK, Graz University of Technology, Austria

Why Trusted Java Applications are needed

The concept of Trusted Computing (TC) promises a way of improving the security of computer systems. The core functionality, based on a hardware component known as the Trusted Platform Module (TPM), is being integrated into commonly available hardware. Although hundreds of millions of TPMs have shipped so far, only limited software support exists. One reason for this might be that the Trusted Computing Group's (TCG) industrial standard software [1] that accompanies the TPM only supports the programming language C. However, a major share of the software market utilizes the platform-independent Java™ environment. The Java language provides inherent security features such as type safety and bounds checking. The runtime environment provides automated memory management, access

control checks and bytecode verification. Performance concerns with Java applications can be mitigated by using just-in-time compilation of Java bytecode. Communications and cryptography are covered by a rich set of available libraries.

This security-by-design makes the Java runtime environment a natural choice for a TC platform. While the current releases of Java do not provide support to access the TPM by default, there are already multiple demonstrated use cases for TC-enabled Java applications and services [2-6].

As a first step, the Institute for Applied Information Processing and Communications (IAIK) has designed and implemented the TCG Software Stack for Java (jTSS) [7, 8]. jTSS is one of the first libraries available that allows the TPM to be used from Java applications. It closely follows the architecture and also the interfaces specified by the TCG, which was designed for C. Thus, for a Java developer using the jTSS, the API requires a completely different approach and programming style than he or she might be accustomed to.

Creating the future standard API

The next step towards further integration of TC into Java is, therefore, to make TPM and TSS-based features available to Java developers in a consistent, object-oriented, and also easy-to-use, intuitive way. Of course, the best way to establish such an API is to standardize it. For Java APIs the well-established way of doing this is the Java Community Process (JCP). The JCP aims to produce high-quality specifications using a consensus-building approach. The central element of the process is to gather a group of industry experts who have a deep understanding of the technology in question and then conduct technical lead work with that group to create a first draft. Consensus on the form and content of the draft is then created using an iterative review process that allows an ever-widening audience to review and comment on the document.

With the aim of establishing such a standard, IAIK initiated the Java Specification Request # 321 (JSR 321). Since then, an impressive expert group has formed. Not only are the Open-TC partners IAIK, Portakal Teknoloji and University of Cambridge contributing, but also major companies such as Intel, Samsung and Sun, and a number of highly qualified and experienced, individual Java engineers.

In the spirit of the Open-TC project, this JSR 321 has chosen an open, transparent and agile working style. Thus, the technical discussion is also open for non-members of the JCP, allowing for further cooperation with, and integration into, the open source community. To increase the transparency and trustworthiness of the process and its results, both the reference implementation and the test suite will be released as open source software. And going even further in this direction, the open source and Java community are invited to partake in the design as well as in the implementations.

In February 2009, the JCP program management office praised four projects from all ongoing JSRs for their openness. Among those presented in the special report [9] is JSR 321. It is acclaimed as one of the most transparent Java Specifications Requests. The other mentioned JSRs are run by such illustrious organizations as Sun, MIT, or Apache.

Everyone has a chance to shape this standard

In Spring 2009, a major milestone in the standardization process was achieved. The expert group released its first complete API draft to the general public as an “Early Draft Review”. The aim is to get valuable feedback in order to develop the most useful specifications. The Expert Group kindly invites all readers to comment on the current publicly available Early Draft. We will then carefully examine and discuss all your comments, and do our best to improve the specifications.

Conclusions

JSR 321 will allow developers to make use of TC functionality in their Java applications. Striving for a new simplified design, the resulting API will be easier to use than the interfaces available today. As an industry standard, it will allow Java developers to use TPM-based functionality independent of the operating system and TSS implementation.

This and the fact that all results will be released under an open source license will hopefully foster the use of trusted technology for research, open source and also commercial applications.

More details about the JSR and development process followed can be found at http://jsr321.dev.java.net/. The specifications can be downloaded from http://www.jcp.org/en/jsr/detail?id=321. Comments and suggestions regarding the JSR should be sent to jsr-321-comments@jcp.org. The review phase will end June 8th, 2009.

References:

[1] Trusted Computing Group: *TCG Software Stack (TSS) Specification*, Version 1.2, Level 1, August 06, 2006
[2] K. Dietrich, M. Pirker, T. Vejda, R. Toegl, T. Winkler and P. Lipp: *A Practical Approach for Establishing Trust Relationships between Remote Platforms using Trusted Computing.* In Proceedings of the 2007 Symposium on Trustworthy Global Computing, LNCS 4912, Springer Verlag, 2007.
[3] J. Lyle: *Trustable Remote Verification of Web Services.* Proceedings of Trust 2009 conference, LNCS 5471, Springer Verlag, 2009.
[4] L. Sarmenta, M. van Dijk, Ch. O'Donnell, J. Rhodes, and S. Devadas: *Virtual Monotonic Counters and Count-Limited Objects using a TPM without a Trusted OS*. The 1st ACM Workshop on Scalable Trusted Computing (STC'06), at ACM CCS '06, November 2006.
[5] T. Vejda, R. Toegl, M. Pirker, T. Winkler: *Towards Trust Services for Language-Based Virtual Machines for Grid Computing.* In Proceedings of Trust 2008, Lecture Note in Computer Science, Springer Verlag, in print, 2008.
[6] S. Yoshihama, T. Ebringer, M. Nakamura, S. Munetoh, and H. Maruyama, 2005. *WS-Attestation: Efficient and Fine-Grained Remote Attestation on Web Services.* In Proceedings of the IEEE international Conference on Web Services (July 11 - 15, 2005) ICWS. IEEE Computer Society, Washington, 2005.
[7] M. Pirker, T. Winkler, R. Toegl and T. Vejda. *Trusted Computing for the Java Platform.* http://trustedjava.sourceforge.net/, 2007.
[8] R.Toegl, M. Pirker: *An ongoing Game of Tetris: Integrating Trusted Computing in Java, block-by-block.* In: Proceedings of Future of Trust in Computing Conference, Berlin, Vieweg+Teubner, 2008.
[9] Susan Mitchell: *Downloads: Making Drafts and Collateral Available.* JCP Report,

http://www.jcp.org/en/press/news/transparency, 2009.
[10] R. Toegl, Open-TC Deliverable D03.d7: *Integrating Trusted Computing into the Java Programming Language: Java-API Standardization*. 2009

Contact: ronald.toegl (at) iaik.tugraz.at

Acknowledgements: The author thanks the JSR 321 Expert Group.

Recent OpenTC publications

Since publication of the last newsletter, the OpenTC project partners have published the following:

Armknecht, F.; Gasmi, Y.; Sadeghi, A. R.; Stewin, P.; Unger, M.; Ramunno, G.; Vernizzi, D.: *An Efficient Implementation of Trusted Channels Based on OpenSSL.* In: Proceedings of the 2008 ACM workshop on Scalable Trusted Computing, Fairfax, Virginia (USA), 1/10/2008, pp. 41-50, 2008.

Cabuk, Serdar; Dalton, Chris I.; Eriksson, Konrad; Kuhlmann, Dirk; Ramasamy, HariGovind V.; Ramunno, Gianluca; Sadeghi, Ahmad-Reza; Schunter, Matthias; Stüble, Christian: *Towards automated security policy enforcement in multi-tenant virtual data centers.* In: Journal of Computer Security, Special Issue on EU's ICT Security Research (2009), pp 1–33.

Cesena, E.; Ramunno, G.; Vernizzi, D.: *Towards Trusted Broadcast Encryption.* In: Proceedings of the 2008 International Symposium on Trusted Computing. TrustCom 2008, Zhang Jia Jie, Hunan, China, November 18-20, 2008.

Chen, Liqun; Löhr, Hans; Manulis, Mark; Sadeghi, Ahmad-Reza: *Property-Based Attestation without a Trusted Third Party.* Information Security: 11th International Conference (ISC 2008), LNCS Vol. 5222, 2008.

Dietrich, Kurt; Winter, Johannes: *Secure Boot Revisited.* In: Proceedings of the 2008 International Symposium on Trusted Computing. TrustCom 2008, Zhang Jia Jie, Hunan, China, November 18-20, 2008.

Dietrich, Kurt: *A Secure and Reliable Platform Configuration Change Reporting Mechanism for Trusted Computing Enhanced Secure Channels.* In: Proceedings of the 2008 International Symposium on Trusted Computing. TrustCom 2008, Zhang Jia Jie, Hunan, China, November 18-20, 2008.

Dietrich, Kurt; Winter, Johannes: *Implementation Aspects of Mobile and Embedded Trusted Computing*. TRUST 2009, April 2009.

Gasmi, Yacine; Husseiki, Rani; Sadeghi, Ahmad-Reza; Stewin, Patrick; Stüble, Christian; Unger, Martin; Winandy, Marcel: *Flexible and Secure Enterprise Rights Management based on Trusted Virtual Domains.* Proceedings of the 3rd ACM Workshop on Scalable Trusted Computing (STC 2008), Fairfax, Virginia, USA, October 2008.

Hein, Daniel M.; Toegl, Ronald: *An Autonomous Attestation Token to Secure Mobile Agents in Disaster Response.* The First International ICST Conference on Security and Privacy in Mobile Information and Communication Systems (MobiSec2009), 3-5 June 2009, Turin, Italy, Springer (in print)

Kursawe, Klaus; Schellekens, Dries: *Flexible µTPMs through Disembedding.* In: Proceedings of the 2009 ACM Symposium on Information, Computer and Communications Security, (ASIACCS 2009).

Lioy, A.; Ramunno, G.; Vernizzi, D.: *Trusted-Computing Technologies for the Protection of Critical Information Systems.* In: Advances in Intelligent and Soft Computing, Springer. CISIS'08 - Int. Workshop on Computational Intelligence in Security for Information Systems, Genova (Italy) 23-24/10/2008, pp. 77-83, 2008

Löhr, Hans; Sadeghi, Ahmad-Reza; Stüble, Christian; Weber, Marion; Winandy, Marcel: *Modeling Trusted Computing Support in a Protection Profile for High Assurance Security Kernels.* TRUST 2009, April 2009.

Löhr, Hans; Sadeghi, Ahmad-Reza; Vishik, Claire; Winandy, Marcel: *Trusted Privacy Domains – Challenges for Trusted Computing in Privacy-Protecting Information Sharing.* The 5th Information Security Practice and Experience Conference (ISPEC'09), April 2009.

Pirker, M.; Toegl, R.; Hein, D.; Danner, P.: *A PrivacyCA for Anonymity and Trust.* In: Trusted Computing (2009), pp. 101-119. International Conference on Trusted Computing and Trust in Information Technologies (TRUST); 2009, LNCS, Springer

Sadeghi, Ahmad-Reza; Stüble, Christian; Winandy, Marcel: *Property-Based TPM Virtualization.* Information Security: 11th International Conference (ISC 2008), LNCS Vol. 5222, Springer, 2008.

Schulz, Steffen; Sadeghi, Ahmad-Reza: *Secure VPNs for Trusted Computing Environments.* TRUST 2009, April 2009.

Toegl, R.; Hofferek, G.; Greimel, K.; Leung, A.; Phan, R. C.; Bloem, R. P.: *Formal Analysis of a TPM-Based Secrets Distribution and Storage Scheme.* In: International Symposium on Trusted Computing (TrustCom 2008). Proceedings, in 9th ICYCS Conference Proceedings (2008), S. 2289 – 2294.

Toegl, Ronald: *Tagging the Turtle: Local Attestation for Kiosk Computing.* 3rd International Conference on Information Security and Assurance (ISA-09), June 25-27, 2009 Korea University, Seoul, Korea, Springer (in print)

Edited by the Institute for Technology Assessment and Systems Analysis, Forschungszentrum Karlsruhe, Germany, on behalf of the OpenTC research project consortium, in co-operation with all partners.

Editor: Arnd Weber, Forschungszentrum Karlsruhe GmbH, ITAS, Hermann-von-Helmholtz-Platz 1, D-76344 Eggenstein-Leopoldshafen, Telephone: + 49 7247 82 3737.

Contact: editor (at) opentc.net

The OpenTC-project is a research project supported by the European Commission, project IST-027635. Its 23 partners are: Technikon Forschungs- und Planungsgesellschaft mbH (project coordination, AT); Hewlett-Packard Ltd (technical leader, UK); AMD Saxony LLC & Co. KG (DE); Budapest University of Technology and Economics (HU); Commissariat à l'Energie Atomique – LIST (FR); COMNEON GmbH (DE); Forschungszentrum Karlsruhe GmbH – ITAS (DE); Horst Goertz Institute for IT Security, Ruhr-Universitaet Bochum (DE); IBM Research GmbH (CH); Infineon Technologies AG (DE); INTEK Closed Joint Stock Company (RU); ISECOM (ES); Katholieke Universiteit Leuven (BE); Politecnico di Torino (IT); Portakal Teknoloji (TR); Royal Holloway, University of London (UK); SUSE Linux Products GmbH (DE); Technische Universitaet Dresden (DE); Technische Universitaet Graz (AT); Technische Universitaet Muenchen (DE); Technical University of Sofia (BR); TUBITAK – UEKAE (TR); and University of Cambridge (UK).

For more information about the project, see: http://www.opentc.net

Feedback to the consortium: http://www.opentc.net/feedback

Archive of newsletters: http://www.opentc.net/newsletter

Subscription: To subscribe or unsubscribe to the newsletter, write an email to <subscribe (at) opentc.net> or <unsubscribe (at) opentc.net>.