GARRISON

Imagining A Hardsec Future

Introduction

The development of information technology over the past 5 decades represents an extraordinary torrent of innovation: from mainframes to the microcomputer, to the cloud and on to edge computing, and from punched cards to the GUI to deep learning. But at the heart of this evolution has been the von Neumann architecture processor – these days instantiated in the overwhelming majority of cases as either an Intel-architecture device or an Arm-architecture device. Even the GPU – which underpins the computational power behind recent advances in machine learning techniques – is really just a tweak on this same underlying architecture.

The von Neumann architecture processor has triumphed because it is a practical implementation of a powerful theoretical concept – the Universal Turing Machine. By supplying different software, the same hardware can play an infinite number of different roles. This is the magic that has, in Marc Andreesen’s now classic words, allowed software to eat the world.

But this universality has a dark side: security vulnerabilities and the potential for malware. The very power that allows us to develop any imaginable functionality simply by supplying the right instructions is also the power that can allow an attacker who spots an unexpected behavioural quirk to substitute their instructions and subvert the entire function of the computing platform. Two vast global industries have emerged. One seeking to find ways to take advantage of this potential for vulnerability in order to further their aims, whether those be criminal gain, intelligence gathering or military advantage. And another seeking to find ways of impeding the actions of the first.

Of course, the hardware manufacturers have not been standing still. Processor vendors have innovated a series of security enhancements on top of the basic von Neumann architecture: techniques such as virtual memory, hardware virtualisation support and trusted boot. Most security tools provided by the global cybersecurity industry are themselves software – instructions for a Turing machine processor. But the techniques adopted by processor vendors are in some cases more fundamental: “hard wired” logic that is implemented at a lower level of computational complexity than a Turing machine, using simpler digital logic that is less vulnerable to exploitation.

Processor vendors face however a difficult set of commercial and economic trade-offs. Their hardware must be universal, which means their security features must be universal. The principal basis for buying decisions is performance, meaning that security features cannot significantly adversely affect performance. And because security features are hard-wired into the processor design, they cannot be changed post-manufacture, which means a huge investment of time and effort to minimise the risk of a flaw in the design or implementation. The scope for security techniques in the processor is therefore quite limited.

Because – as a result – most security tools are provided as software, they themselves suffer from the same tendency to vulnerability as any other software. And indeed, vulnerability databases are full of examples where security tools themselves have turned out to be exploitable. To make things worse, in many cases security tools demand a privileged position in order to be able to carry out their advertised role: if these tools are exploited, an attacker can get a shortcut to the heart of the systems, bypassing even the security controls built in to the processor hardware to protect against malicious software.

This is a dark and dirty truth at the heart of the security industry. Indeed, for those organisations who have historically cared most about cybersecurity – national security organisations such as military and intelligence agencies – this truth makes the mainstream security industry fundamentally unreliable. While their tools might be acceptable for protecting the relatively humdrum and mundane, they are not trusted to protect the truly secret or significant. Those organisations have therefore spent many years searching for an alternative approach to security. The answer is a technology known as hardsec.

Although the heart of the information technology revolution has been the von Neumann architecture processor – an instantiation of the universal Turing machine concept – it is not the only sort of silicon in use. Of the nearly $100bn spent on digital logic in 2018 (source: PwC), more than 75% is represented by sales of processors and GPUs. But there is also a market for specialised silicon devices: devices tuned specifically for particular tasks such as network switching or digital signal processing. Many of these devices are themselves actually processors under the covers – but others use non-Turing machine logic to implement their functions, achieving faster performance by stripping the functionality down to its very essence.

By definition, these specialised devices are not universal. But could they form the basis of security tools that did not suffer the inherent vulnerability of software? In principle, this would be possible, but in practice it is rare that the economics will work. Although we have described a market for devices that are specialised, these specialised tasks must nonetheless be extremely widespread in order to justify the costs of design and manufacturing. There is certainly a role for such devices in cybersecurity, but that role must be limited.

There is however an intermediate class of device. Representing a niche but nonetheless significant global market of approximately $5bn per annum, this is the market for a class of device called a Field Programmable Gate Array, or FPGA. What hardsec recognises is that these devices can form the basis of a pervasive security revolution.

To understand this revolution, we need to look at two things. Firstly, what is the core hardsec concept? How do FPGA devices allow the development of security functionality that does not suffer the inherent vulnerability of software but which overcomes the economic limitations of hardware? And secondly, how can we take advantage of that insight to map out a path to a future where we can retain the extraordinary advantages that the von Neumann architecture processor has delivered, while adding to it the level of robust security that we are going to require?

FPGAs and the hardsec concept

Field Programmable Gate Array devices allow the specification of low-level electronic circuits using "programming languages" (actually technically called "hardware definition languages") that can be typed by an engineer. These circuits can be programmed into the devices in the field (hence "Field Programmable") meaning that the FPGA device can have one set of circuits at one time, and another set of circuits at another time.

Of course, one thing that an engineer could do would be to programme the FPGA with circuits that implement a von Neumann architecture processor. Indeed, engineers who develop processor chips do exactly that, as a way of testing the behaviour of their designs before they are sent to the fab for manufacturing. That means that an FPGA can be a Turing machine.

But critically for hardsec, FPGA devices can only be programmed using specific physical pins on the device. That allows us to do something quite novel from a security perspective. Firstly, we can restrict – by physical hardware design and implementation – who can reprogramme the FPGA to those who have access to a well-protected privileged management environment. In practice, this is identical to the out-of-band management networks that are an integral part of existing data centre designs. And secondly, we can restrict the circuits that we programme to non-Turing machine designs.

With this approach, we can achieve the best of both worlds: the strength of hardware security, with its basis in non-Turing machine logic that doesn't suffer the vulnerability of software, with the flexibility of software that allows common base hardware platforms to play multiple different roles depending on how they are programmed.

There's a final piece of the puzzle that turns hardsec from an interesting academic idea into one which is truly practical, and this is an idea that emerged from the UK's National Cyber Security Centre (a part of the GCHQ intelligence agency). Their idea – often known as "transform and verify" – is to use regular software (on regular von Neumann architecture processors) to transform data into a format which is easy to verify. The idea fits perfectly with hardsec: we can use software to transform input data into a form which is easy to verify using non-Turing machine logic in an FPGA. On the output side, we can of course re-transform our verified data into the original format before delivery to the destination.

Imagining the hardsec future

How can the insight of hardsec be exploited to deliver a practical future? The answer is certainly not to replace our von Neumann architecture processors with FPGAs. Of course, we could in principle do that – given access to the programming pins, an FPGA device is a universal Turing machine. But we would sacrifice performance, and we would make the jobs of software developers significantly more difficult, throwing away decades of investment in development tools and techniques. If we want to see continuing innovation in areas such as AI and machine learning, we must build on the processing architectures we have. The trick to exploiting hardsec is to think about isolation.

Isolation has been at the heart of security architecture for decades. The security tools built in to the processors we use today are designed above all to enforce isolation: to ensure that different processes or virtual machines running on the same physical processor cannot see or change each other's data. What hardsec allows us to do is to apply that principle of isolation at the application level.

Concepts such as processes and virtual machines are universal, meaning that non-Turing machine features to enforce isolation between them can be built into processor hardware. But applications are unique. Using FPGAs with hardsec principles, we can build unique isolation solutions that are application-specific: lifting non-Turing machine isolation up the stack to deliver a new era of robust security.

Underlying the hardsec future is the concept of trust domains. Software is inherently vulnerable, and data comes in varying flavours of trust and value. Sometimes, it is necessary to use software to process high-risk data: the result may be software exploitation and compromise. Other data is high value and must be carefully protected: it is important that the software which processes it is not exploited. On our journey towards the hardsec future, we will – in so far as possible – draw trust domain boundaries in order to isolate the two software implementations from each other, so that if the first software is exploited, it cannot access the high value data processed by the second.

But inevitably, even after drawing these trust domain boundaries, the two software applications will need to interoperate. In many cases, the application that processes the high-value data will need information which can only be derived from the processing of the high-risk data. We must therefore focus our security attention on these "cross-trust-domain" interfaces: taking careful steps to ensure that data produced in the high-risk domain is safe before allowing it ingress into the high-value domain.

This is hardly a new concept: every web developer in the world should be aware of the need to sanitise untrusted user input in order to avoid exploits such as SQL injection. That well-known sanitisation approach can be extended to much broader classes of data interface – ensuring well-formedness for REST APIs, for JSON or XML schemas, and even for complex file objects. But when we try to implement this sanitisation approach today, we run into two key problems.

Firstly, the application-level data rests on a complex stack of underlying communication protocols – TLS, HTTP, TCP/IP, Ethernet – and to apply the sanitisation approach correctly requires sanitisation not just at each of these layers, but also of the potential interactions between the layers. This is hugely complex. Secondly, the software we write to implement this sanitisation is itself likely to be vulnerable to exploitation. And if an attacker can exploit the sanitisation software, they can use that as a basis for injecting high-risk data into high-value domains.

Hardsec allows us to take the sanitisation approach and make it work for us. By terminating the protocol stacks in software and then using FPGA-based non-Turing-machine logic to carry out the application-level data sanitisation, we have a robust security mechanism that can be relied on to transfer data safely between different trust domains.

Of course, implementation is far from trivial: the market for hardsec tools is still embryonic and development of hardsec-based sanitisation remains today a task for specialists. But the technology is developing rapidly, and tools are emerging to allow the specification of hardsec sanitisation using familiar techniques such as Regular Expressions and XML. With these tools, we can imagine widely-deployed hardsec platforms that can be used by developers and security teams to secure key interfaces between trust domains

Remembering the human

One day, that may be all there is to it. But the picture we have painted so far ignores one critical legacy feature which continues to persist in today's information technology landscape: human beings. In some cases, humans can themselves be neatly partitioned into trust domains: in many cases the marketing team do not need access to critical operational systems. But far too often, individual humans need to interact with systems in many different trust domains.

Human beings need user interface devices in order to interact with systems: and these user interface devices (laptops, phones and tablets) then become another potential interface between the trust domains. In the past decade, these human interface devices have often been the target of choice for attackers: with a carefully crafted phishing email from the low-trust Internet, an attacker who can compromise the human interface device may then be able from there to gain access to the most trusted data and systems.

There is little point adopting a hardsec-based approach to cross-domain interface sanitisation if human interface devices provide an easy-to-exploit back door. The hardsec future must therefore go hand in glove with an approach to ensuring that human interface devices themselves maintain isolation between different trust domains – ie ensuring that those devices are not themselves vulnerable to exploitation. This is a non-trivial requirement, because there is a huge investment in existing human interface devices, which depend on software running on von-Neumann architecture Turing machines.

It is therefore certainly not practical to re-engineer the fundamentals of the human interface device. But once again, an interface sanitisation approach can be adopted. The actual interface between the device and the human is relatively limited – primarily electromagnetic waves generated by pixels and sound vibrations generated by PCM digital audio. The good news is that these are data formats which are trivially sanitisable. If we can sanitise the data that is delivered to human interface devices, we can maintain high trust that these devices have not been compromised. My company produces a platform that does exactly this – sanitising human interface inputs using non-Turing-machine implementations based around the hardsec principle.

From virtualisation to the crunchy cloud

Having identified a way of securing cross-trust-domain interfaces, and avoiding the risk of the human interface device, the final question is how we should maintain the fundamental isolation between different trust domains

In many cases, today's answer is the use of hardware-based virtualisation isolation implemented within processor silicon in the cloud. In an ideal world, that would be the end of the story. And yet, the practical truth is that the security of today's virtualisation is far from bullet-proof, and some security-conscious organisations as a result are reluctant to entrust the isolation of their most sensitive systems to the cloud. For those organisations, isolation is maintained by implementing different trust domains on different physical machines.

Why is it that today's virtualisation is considered too weak by the most security-conscious organisations? After all, hardware virtualisation support includes the use of non-Turing-machine techniques for isolation enforcement. The answer is less in-principle than in-practice: the commercial reality of buying decisions means that flexibility, cost and performance have been prized above security, and where they have conflicted, security features have been weakened as a result.

That balance is changing, and increased focus combined with new security designs hold out the promise of a truly "crunchy" cloud, where the isolation barriers between virtual machines are genuinely "hard". With the underpinnings of a future crunchy cloud and hardsec-based interface sanitisation, the hardsec future promises a world where continued software-based innovation can truly coexist with strong levels of security.

GARRISON

Email	info@garrison.com
UK telephone	+44 (0) 203 890 4504
US telephone	+1 (646) 690-8824

www.garrison.com