



Software is becoming an increasingly important part of the electronics industry. Whether it's the design packages that enable the creation of complex chips, the operating systems empowering embedded designs or the bitstreams that program fpgas, software is everywhere.

Depending on how it's applied, software can be an enabler or a differentiator. In both cases, it needs to be protected. One instance of software as an enabler is supporting the growing need to run multiple operating systems. This has seen the rise of the hypervisor, which aims to reduce risk to minimal levels by partitioning. However, when software is a differentiator, the threat is different – theft.

So if you have software which you believe is a differentiator, how can you protect it? Kevin Morgan is chief technology officer for Arxan Technologies, a US based company which aims to protect IP from software piracy, tampering, reverse engineering and any manner of theft.

He said: "Our technology is embedded into applications and can generate a response to such things as tampering. The protection features we can provide can see what a software package is connecting to and whether that's an authorised connection. It's non disruptive and operates at run time."

Arxan's technology is based on work done at Purdue University. Morgan describes the technology as 'mature' and in its fourth generation.

"Attacks are happening across the range," he claimed, "and often driven by organised criminals. Attacks can be web based – such as theft of data and IP – but can also target embedded devices."

According to Morgan, attacks take a number of forms. "Unauthorised access allows software to be altered. This can be a second order process, following a phishing attack that identifies software with access to critical data, or via an insider.

"There is also virus insertion. Many think this is the province of 'thrill seekers', but it is mostly a criminal activity. And there's IP theft."

A recent survey by security specialist McAfee found product development

# On Guard!

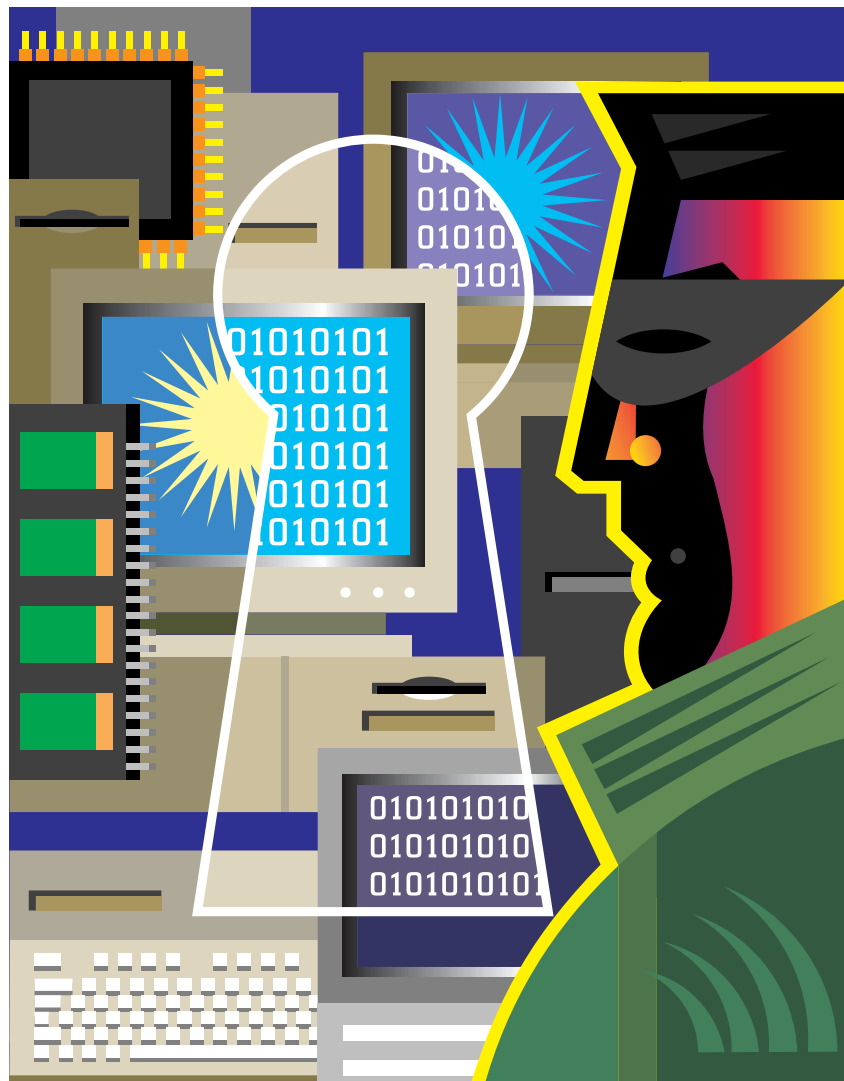
*How to protect software against threats. By Graham Pitcher.*

manufacturing companies lost on average \$4.6million per firm in the last year. Respondents lost IP worth an average of \$4.6million per firm due to security breaches, while the financial services industry suffered the highest losses.

"This means people are getting into what should be a perimeter sealed environment," Morgan observed.

Finally, there's piracy, where the code certification routines in a software package are identified and a work around developed so the package runs without licence management.

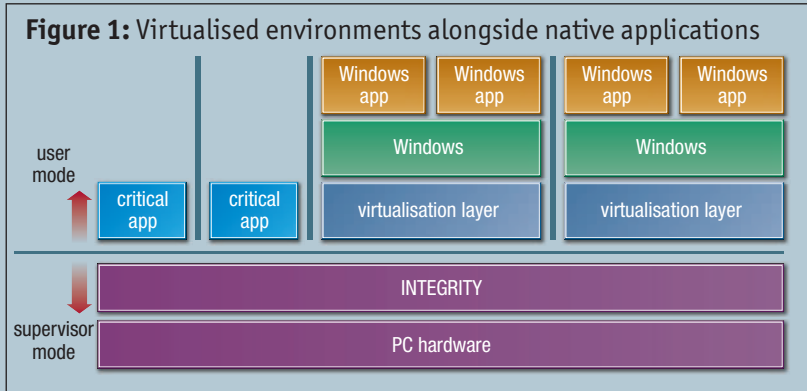
Arxan's solution is GuardIT, which enables a program to defend itself by detecting when it is under attack and to react if it is being modified. Morgan





### Virtualisation insecurity

Are hypervisors as secure as you think? By David Kleidermacher.



Hypervisor technology is beginning to sprout up in real time telecommunications, mobile devices and other electronics products. But embedded systems have different requirements from data centres and a highly secure virtualisation environment enables some compelling applications.

A number of studies of virtualisation security and successful subversions of hypervisors have been published, demonstrating the risk of an 'escape' from the virtual machine (VM) layer, exposing all the guests, is very real. According to one analyst: "Virtualisation is essentially a new operating system ... and it enables an intimate interaction between underlying hardware and the environment. The potential for messing things up is significant."

There is more to security than using the word 'secure' or 'trusted' in product names and, sadly, the world has become accustomed to the 'fail first, patch later' mentality of insecure software. Thus, many of the world's systems run

insecure operating systems and hypervisors, leaving them open to compromise.

#### Secure virtualisation

Hypervisors typically employ a monolithic architecture, which requires a large body of operating software, including device drivers and middleware, to support the execution of one or more guest environments. In addition, the monolithic architecture often uses a single virtualisation component (itself a complicated piece of software) to support multiple guest environments. Thus, a single flaw in the hypervisor may result in a compromise of the fundamental guest environment separation intended by virtualisation in the first place.

An alternative, but similarly insecure, approach uses a trimmed down hypervisor that runs in the microprocessor's privileged mode, but which employs a special guest OS to handle I/O control and services for the other guests. Thus, a complex, monolithic body of software

must still be relied upon for system security.

Green Hills Software's virtualisation architecture places virtualisation complexity and related I/O drivers and middleware into user mode applications outside the trusted computing base, which contains only the secure microkernel: GHS' INTEGRITY. The microkernel provides low level hardware support, resource partitioning and scheduling for the virtual environments. A separate instance of the virtualisation infrastructure is used for each guest environment, precluding cross VM escapes.

The combination of virtualised and native applications on one processor provides a compelling cost and power efficient operating environment, ideal for embedded electronics and portable devices (see figure 1). This hybrid model also takes advantage of multicore processors by enabling concurrent execution of native and virtualised subsystems.

The flexibility afforded by virtualisation has proven powerful in the data centre and promises even more varied and compelling advantages throughout the electronics world. However, the proper virtualisation architecture can drastically improve security without sacrificing the utility of legacy software. INTEGRITY is appropriate for electronic products that demand a high level of security, reliability, and functionality.

#### Author profile:

David Kleidermacher is Green Hills Software's chief technology officer.

• For the full version of this article, go to [embedded.newelectronics.co.uk](http://embedded.newelectronics.co.uk)

claimed the approach, which runs at the binary level, hardens applications to prevent unauthorised access. "We take an executable file," he continued, "and output a slightly modified executable. The new file operates correctly, but includes small units of software called Guards. Guards can interact with each other and there may be multiple levels of Guards. These could be aggressive, in the case of licence management and overt threats, or could take more subtle actions." Morgan added these subtle actions could include causing errors in software applications.

"When the Guard fires," he explained, "there are canned actions, but they can

also call arbitrary user functions and the designer can choose what those functions accomplish, including shutdown or exit."

He gave an example of software used in CAD and milling operations. "If the software is being used illegally, the Guards create errors in the code, which bring unexpected machining actions."

The software also has anti debug features, which block the application from executing in the presence of kernel mode debuggers or when executed in an emulated environment.

However, he emphasised that GuardIT is a tool and it is the customer's decision about what actions should be taken.

Users can also vary the actions of Guards, so responses from different copies of software are different. "Users can create the same Guard network, but with different instance," said Morgan. "This could be applied in different software builds or in every copy. If one piece of software gets cracked, it may not be applicable elsewhere." Arxan is also working on a product that allows a formal watermark to be embedded in software.

Because it is a binary solution, GuardIT relies on an instruction set architecture and the software currently runs on the x86 and PowerPC architectures, with support for Linux and Windows.