WHITE PAPER

Securing Edge Devices and Data in Distributed Industrial Applications

Johnny Fang
Product Manager, Moxa Inc.

Kyle Pearson
Technical Writer, Moxa Inc.

MOXA®

Securing Data in the Smart Grid, Solar Grid, & Transport Grid

In the last few months there has been a resurgence of interest in the Internet of Things, suggesting that the era for building distributed big data industrial applications has at last dawned. Power companies are working hard to build the Smart Grid by melding IT technology with electricity services. Oil companies are integrating their machinery into digital oil fields. Photovoltaic technology is being developed as a consumer product, to integrate with the Smart Grid centralized solar farms, and distributed residential solar stations. Urban environments are deploying monitoring systems for public security and accountability, while intelligent transportation systems are preparing for a future where cars drive themselves, and those that don't are advised by traffic systems that communicate instructions relevant to one's location and destination, to reduce or eliminate bottlenecks and jams.

Yet each of these future systems faces one significant challenge: data security, not just for user-space input/output, but for protection of low-level OS integrity—including the boot loader and BIOS—as well. Moreover, unlike IT environments, where AAA security protocols (authentication, authorization, and accounting) may be served from machines located behind a very large physical security layer, for these industrial cloud deployments the computers and networking devices that will serve them must be located in public and commercial spaces, or in private residences—places where the actual owners of the devices (say, the electrical company) have no real control over their physical security. This makes the physical security of these cloud devices an easy attack vector, and is the one layer that traditional IT has not much addressed.

Enhancing Conventional Wireless Security

At the same time, there remain the conventional challenges of securing data across the open Internet, and over local wireless links like 3G cellular, or Wi-Fi. Electrical smart meters must not be corruptible by simple home hacks. ITS systems must not be vulnerable to wireless DoS or man-in-the-middle attacks. Residential and public data must maintain appropriate standards

Released on July 1, 2014

Moxa is a leading manufacturer of industrial networking, computing, and automation solutions. With over 25 years of industry experience, Moxa has connected more than 30 million devices worldwide and has a distribution and service network that reaches customers in more than 70 countries. Moxa delivers lasting business value by empowering industry with reliable networks and sincere service for automation systems. Information about Moxa's solutions is available at www.moxa.com. You may also contact Moxa by email at info@moxa.com.

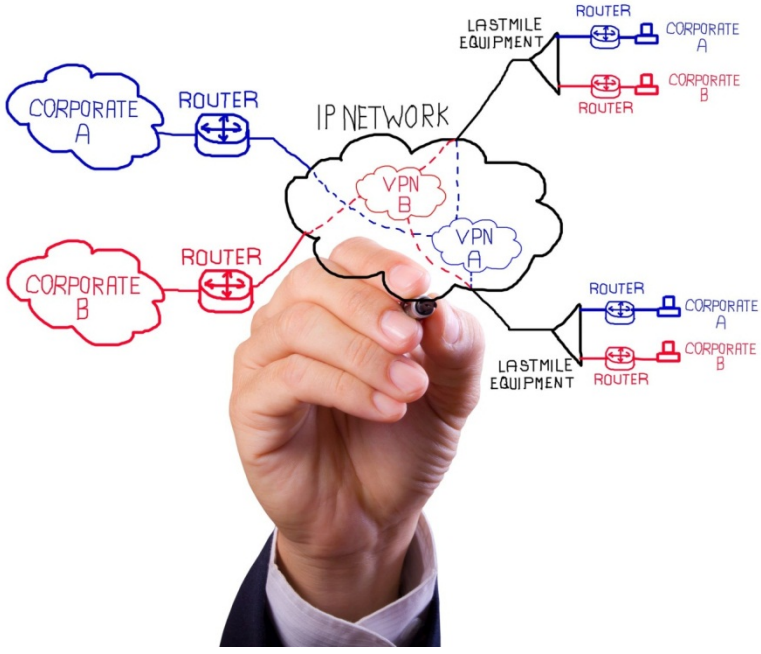
How to contact Moxa
Tel: 1-714-528-6777
Fax: 1-714-528-6778

MOXA®
Reliable Networks ▲ Sincere Service

of confidentiality, and not be liable to unauthorized manipulation. As data travels from remote, edge devices like traffic monitors, smart meters, and mobile devices, the information will often first travel through wireless gateways and then be forwarded over the open Internet. Fortunately, IT solutions already provide trustworthy guarantees for these links: wireless encryption of wireless links into WPA2, backbone communications over encrypted virtual private networks (VPN), and strong packet filtering with fully implemented AAA protections. Before exploring how to integrate data security into the device's physical layers, let's first examine how packet filtering and VPNs work together to build highly secure networks.

Packet Filtering and Firewalls

Packet filtering and firewalls are a relatively simple consideration. Essentially, network engineers must cut off all available entry points to a client, and restrict communications between it and the server to only those packets and ports which are absolutely necessary. Some features are a requirement, today: stateful and application-aware firewalls are a must, for most cloud applications, for instance. Fortunately, there are very powerful tools freely available to help implement these technologies. The Linux-based Netfilter/IPtables package is a stateful, application-aware firewall that is among the most widely used on the Internet. Netfilter's modular design, flexible configuration options, and scalability allow for practically limitless deployment and feature expansions. The only drawback to using Netfilter/IPtables is doing the hard work of designing multiple layers of packet filtering security over a widely distributed computing architecture. Widely distributed networks like those found in the industrial cloud will require multiple layers of overlapping security zones, and while Netfilter admirably fulfills its role as the-only-packet-filter-you'll-ever-need, the task of designing a highly secure filtering layer into a network composed of thousands of devices will require a lot of time-consuming and minutely detailed design and testing.



Firewalls strive to keep unauthorized intruders and services from gaining network access. In contrast, a virtual private network, or VPN, strives to keep communications across the open Internet a private affair, giving remote clients full access to a private network by wrapping all communications between the two sides in a heavily encrypted stream managed by a dedicated VPN gateway. IPsec is currently the most commonly used form of VPN encryption, a strong algorithm with full AAA functionality. By using a proven VPN suite, network engineers may configure a remote device to connect to a server using either IPsec or TLS, making it very, very difficult for malicious attackers to intercept or interfere with the data stream.

The Basics of VPN Tunneling

On a VPN's server side, cryptographic keys (or "passwords") are set up to allow remote clients to identify themselves and connect to the central network. The clients' cryptographic keys are kept secret by a process similar to what occurs with wireless encryption, and allow the server not only to authorize the device to connect, but also to verify the integrity of all data received from it. These two aspects represent the mix of authorization (determining what—if anything—the client may do on the network), accounting (keeping track of what packets have been sent, what have been received, and whether any packets in the stream have been tampered with), and authentication (verifying that the device and its data stream are what they claim to be) that makes a VPN so powerful. When clients are configured with strong (256 bit) cryptographic keys, a VPN data stream is uncrackable by brute-force means. Additionally, VPNs may be easily integrated with RADIUS or Diameter servers, as well, which provide additional, powerful accounting protocols that log and report on resource usage by remote clients—very useful for around-the-clock services like utilities, or ITS.

While VPNs provide some very useful security and accounting features, their deployment does demand a bit more planning and care, and can affect your final network design. For instance, the reliability of a VPN is dependent upon the quality of the connection maintained by the service provider. If high reliability is a requirement, network engineers will need to work closely with the ISP to guarantee the quality of the network connections or else compensate by designing the network using distributed redundancies over shorter segments. Similarly, to guarantee future scalability, care must be taken to either use established open standards—which may be freely expanded as needed—or be at peace with the idea that future expansions may require expensive proprietary alternatives that cannot be easily integrated with the original system.

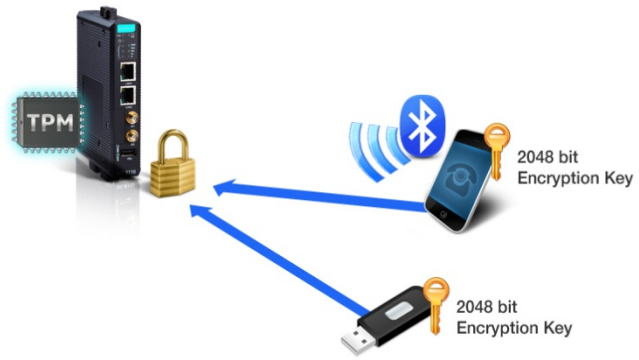
The Power of Trusted Platform Computing

While VPNs and packet filtering are a necessary part of any industrial cloud deployment, these elements only satisfy part of the requirements. As described above, industrial computing platforms that are deployed into commercial, residential, or public areas require strong guarantees for the physical security of the edge devices. Fortunately, there exists a powerful, currently under-utilized tool that fulfills just this need: the Trusted Platform Module, or TPM.

For those who aren't familiar with it, TPM was developed by a consortium of IT corporations called the Trusted Computing Group (TCG), who worked with the ISO and IEC to establish a means of developing highly secure computing platforms that provide strong encryption and security guarantees for use in large enterprise networks where every computing station on the network must be secured against tampering. First composed of a core group of industry giants AMD, Cisco, Hewlett-Packard, IBM, Intel, Microsoft, and Wave Systems Corp, today the TCG continues to evolve, with input from over 105 participating members, all of them world-class enterprise IT manufacturers. Consequently, TPM has typically been associated with the IT industry, and not least because up until recent years there has never been much of a need for encrypting data on industrial networks. Where enterprise networks were responsible for storing and manipulating sensitive, strategic business data that could be exploited by hostile competitors, IA networks were responsible solely for localized input/output, monitoring processes, and remote control. Yet in today's era of distributed big-data industrial systems,

this is now changing: the types and amounts of information available on industrial systems present a very real security risk for everyone, both individually and as a society. Data encryption is today as necessary in industrial networks as it is in IT networks.

While already quite common on enterprise IT hardware, TPM has rarely been applied to embedded RISC computers. Bringing these two tools together gives system integrators and industrial engineers a powerful new tool in their security arsenal. The virtue of TPM is that it defines a hardware standard that permanently incorporates cryptographic keys into a device's physical composition. By outfitting each hardware system with a unique, hardcoded cryptographic key, it becomes possible to give every computing platform or networking device virtually unbreakable encryption and AAA layers that protect not only all software and data, but also the physical composition of the device, including chipsets and peripherals. Because each cipher is unique to each device, TPM allows the integration of the very lowest levels of device software and firmware with physical-layer security, allowing for the possibility of creating computing platforms that can detect any physical alteration or interference in the device's normal functioning, and issue emergency shutdown orders.



To get an idea of how TPM can work, imagine the VPN situation described above. When a client requests access to a VPN, the server responds using an encrypted, asymmetric handshake that keeps all root keys (or "passwords") hidden from public view. Using this asymmetric process, the VPN server and the client exchange encrypted messages that authenticate the client and allow the server to authorize access to the remote network, all without ever exposing the private, cryptographic keys they are using to confirm the login. What TPM does is create a specific cryptographic key for each individual device, hardcoded within the platform itself. Devices may then use this key to both generate more keys, and to authenticate hardware components within the system. For instance, to verify that a read-only operating system has not changed, TPM can create a hash of the drive image, encrypt this using its highly secure cryptographic key, and store it locally, in a hardware-based platform configuration register (PCR). From that point on, TPM will be able to detect any changes in the operating system. Similarly, TPM can use the same method to test things like the BIOS, chipsets, MACs, and so forth.

Software support for TPM is already robust, with established vendors like Microsoft and Cisco providing solid, feature-filled (but closed) end solutions. At the same time, open source projects in GNU/Linux also provide numerous software packages which may be freely adapted, enhanced, and integrated to provide whatever combination of features is needed. These software systems empower TPM to take a variety of emergency security actions that range from ordinary corrective maintenance to alarms and full system shut-down. TPM can be put to many uses: password management, disk encryption, or binding and sealing of the entire platform (hardware plus software), and it can also be configured and administered using

remote, mass deployment software. Generally speaking, TPM brings only strong benefits to the industrial user.

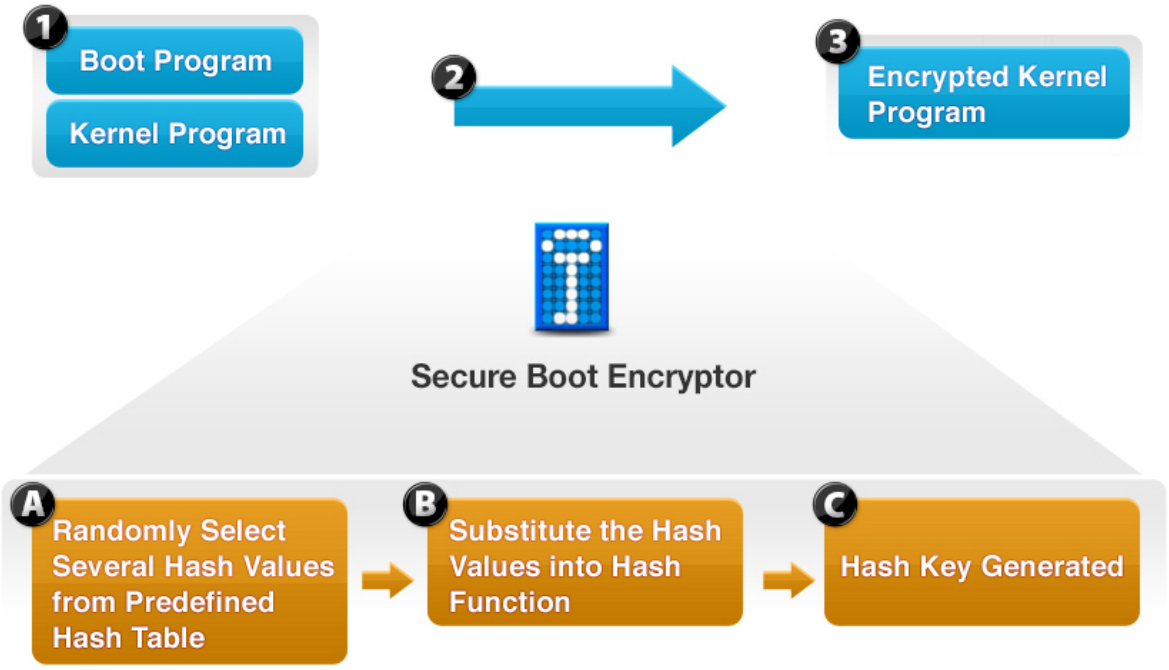
Securing the Kernel: A Self-Authenticating Boot Process

There is, however, one problem that TPM does not address: the possibility of a man-in-the-middle attack, where TPM is fooled into thinking that a local drive or memory cache is being served to the boot loader, when in fact a proxy image or cache is instead being used. Once TPM has validated and authorized an operating system, the boot loader begins the process of loading it from the drive image. If, at this point, an attacker is able to insert a data logger into the process, then it is possible to lift the TPM key from the hardware, and from that point forward anything is possible. An attacker could, for instance, spoof the OS and upload an alternate, hacked kernel into memory, one where the correct keys will always be returned regardless of what happens elsewhere in the system. From that point on, the entire platform is rooted and vulnerable to any imaginable exploit. Moreover, where it can be done once, this can be accomplished repeatedly. Thus, to give a platform full protection, a boot procedure that protects the system must be created that first authenticates the kernel before continuing any further.

Samsung, Windows, and several other groups—big corporate manufacturers, academics, and independent partnerships alike—have already developed many different patented designs for secure boot technology. Most of these designs use symmetric key technology (i.e.—a single cryptographic key is used for both encryption and decryption) to build a hash that is then stored in either volatile memory, or on the local storage drive. This approach leaves the system vulnerable, in most cases, to direct manipulation of the full OS image, or to attack vectors where volatile memory is spoofed, or otherwise manipulated. Additionally, many of these approaches suffer one or more weaknesses like a failure to include a randomized cipher, storing the cipher in ROM, or an overly resource-heavy, multi-step process.

All of these deficiencies may be gotten around rather easily by building the cipher and authentication system directly into the boot loader. A secure boot system of this nature would take a randomized cryptographic seed from the platform's kernel—say, four bytes of code, selected at random—and then, using a bitwise operation, append to it the place in the kernel from which it was pulled and the direction in which it was read (forwards or backwards). Then, after concatenating all of this into a single key, the system generates a cryptographic hash using a passkey that is either taken from TPM, or entered by the user. From this point on, whenever the system boots up the boot loader will use its locally stored cipher key to decrypt the hash, read where and how the original key was generated, and then check the kernel to make sure it has not changed. If the kernel has changed, the system exits the boot process, thus making it impossible to boot the system until a technician can come to rectify the discrepancy

Even without TPM, this secure boot process still provides precisely the same protection, guaranteeing that nobody has tampered with the system kernel. Yet while it's not necessary to use TPM, when integrated with a trusted platform module this secure boot process is a powerful enhancement to system security. When used with TPM, the boot loader is able to give the system the one last major protection trusted platform modules fail to address: protecting the system from man-in-the-middle exploits that make use of spoofed kernels.



Clearly, a secure boot process of this sort may only be used on a read-only system where the kernel and OS do not change. Yet it brings several advantages over other systems. First, and perhaps most importantly, is that the seed key (taken from the kernel) will be different on every platform where it is used, making the possibility of discerning the key by brute-force attacks impossible: even if an attacker were able to gain the root cryptographic key (whether taken from TPM or entered by hand) and start generating one's own keys, the possibility of discovering precisely which part of the kernel was used to build the seed is statistically impossible. Secondly, this approach is handled entirely by the boot loader, within the platform's boot sector: it does not draw on volatile memory, and it does not require any files to be stored on the local storage drive. Finally, it is economical, whether in terms of resource-usage, speed, or complexity.

Moreover, what this secure boot process confers is the confidence that, when used on a read-only operating system configured with TPM-integrated local storage encryption, data encryption, password protections, and suitably robust AAA protocols at both ends of the network, this remote platform will give the highest protections conceivable to your network's edge devices, whether they are located in a residential smart meter assembly, or in MIT's computer science security lab.

Locking Down the Inputs

There is one other feature that should be standard on any embedded computing device intended for deployment in wide area industrial networks: the device should be able to be fully locked down from any input interfaces. That means that all console and USB interfaces must be able to be turned off so that they cannot be used, thereby shutting down any possibility of infection from viruses, or cracking exploits. Once this has been achieved, pretty much the only possible way to further increase security would be to disconnect the computer from the Internet and lock it away in a strong box.

Bringing It All Together

A RISC platform that features TPM, a kernel-authenticating secure boot feature, strong encryption, VPN tunneling, and full interface control delivers a strongly secure device suitable for deployment in residential and commercial settings. An embedded computer featuring carefully engineered security of this sort may be used in smart metering applications, residential solar solutions, intelligent transportation systems, and any number of wide area industrial cloud applications. To find out more how these features come together to give you a powerful, compact, small-footprint embedded computing platform, visit the Moxa website and read up on the UC-8100 universal computer for distributed computing applications.