Continental Automated Buildings Association

Information Series

IS 2003-29

The Inside Story: Embedded Security for Constrained Devices

CABA
www.caba.org

The Inside Story: Embedded Security for Constrained Devices

Report Date: August 2003

Reprint Date: December 2003

certicom™

The Inside Story

embedded security for constrained devices

The Certicom 'Got Security?' White Paper Series

August 2003

certicom™

THE RISKS

The Meta Group (2002) reports that, over the next few years, more than 50 percent of enterprises will deploy PDAs and smart phones for communication, coordination, planning and other corporate activities. The introduction of 2.5G and 3G devices makes more — and more complex — applications possible. Packed into constrained devices, these applications often create loopholes with the potential to compromise security.

The risks:

- Loss of service (SPAM)
- Malicious attacks (Virus/Trojans)
- Data/ identity theft
- Unauthorized access to enterprise resources

The Security Challenge

Security has become an undeniable requirement for wireless and other constrained devices. Fast on the heels of consumer uptake, corporations and government organizations are making wireless technology part of their daily lives – using it to handle private and sometimes sensitive information.

Obviously, that information has to be protected. The question for device manufacturers is how? This paper explores the answer.

What complicates matters is the fact that every user group has different expectations when it comes to security.

Wireless service providers, for example, require solutions that allow them to deliver and store data not only securely but also ***reliably***. Their business depends on it. Interoperability is equally essential – meaning that any solution adopted has to be standards-based. Finally, to remain competitive, service providers need whatever solution they adopt to be bandwidth-friendly.

Enterprises have even more complex needs. With numerous users on the move, they have to extend their internal network applications to wireless devices. Enterprises therefore insist on strong user authentication and access control, as well as interoperability with their existing, costly infrastructures.

Government security requirements go one step further by adding a specific cryptographic standard such as FIPS 140-2 to the list of enterprise demands. FIPS is the U.S. Federal Information Processing Standard instituted by NIST: the National Institute for Standards in Technology.

Because the same device may be required to deliver a level of security that meets any of these groups' needs, device manufacturers must provide – as a default – a rich, flexible and interoperable set of security components.

To Embed or Not To Embed

Developers have basically two options for securing wireless and other constrained devices: they can utilize add-on components or extensions, or they can embed security functionality within the device itself.

In reality, add-ons add not only functionality but also complexity. They can be cumbersome and inflexible. They require specialized vendor support. And they create an element of risk for developers by taking away an aspect of control over device performance.

As proprietary offerings, add-ons can also introduce problems of interoperability leading to longer-term risks as well: should the add-on solution suddenly go off the market, a user's whole security investment could be lost.

Finally — and perhaps most importantly — add-ons disrupt the unity of a device's security architecture.

For these reasons, embedding standards-based security seems to be the more practical alternative. Of course, building new core functionality into a constrained device has its own challenges, especially when that functionality is as sophisticated, power-hungry, and potentially resource-intensive as security. But in the end, embedded security creates a strong, unified platform for creating applications that meet diverse user needs and at the same time conserve internal resources.

This last remark points to the heart of the developer's challenge, which has really to do with striking a balance between security and such crucial considerations as device performance, interoperability and reliability.

Developers must first determine how much security they need to build in. This helps clarify design parameters such as memory requirements, processing power, time to market and cost.

According to NIST — and in light of advances in cryptanalytic attacks and increases in computational speed —128-bit protection is recommended to achieve relatively lasting security (to the year 2036 and beyond). This has been echoed by findings of RSA Laboratories, and is reflected in the development of emerging standards (see Figure 1).

minimum bit-security level	80	112	128
protection lifetime of data	present-2015	2016-2035	2036 and beyond

source: NIST SP 800-57

Figure 1: *How long will it last? Protection lifetime vs. level of encryption*

Starting Points

Before delving deeper into the technological problem of how to enable 128-bit protection in constrained devices, it's worthwhile to take a step back and consider the larger picture of wireless security.

The first observation to make is that wireless is just another connectivity option; the challenge of achieving wireless security has many parallels with the challenge of securing data in the wired world.

Technologies such as SSL, IPSec and S/MIME, which are used widely in wired systems, can be adapted to the demands of wireless communication relatively easily. Doing so achieves two aims: 1) it improves interoperability through standardization; and 2) it creates a consistent security policy across the enterprise.

Corporations and government departments largely insist on standards-based security because the protocols involved are interoperable with existing enterprise policies and infrastructure, and because they've also proved their effectiveness over time.

This latter point is particularly important, as experience has shown. Attempts to develop quick solutions to security challenges often fail: the Wired Equivalent Privacy (WEP) for IEEE 802.11b is a case in point. Developed expressly for wireless applications, WEP was comprehensively broken within a relatively short time after its introduction. Rather than reinvent the wheel, it seems sensible to use the trusted protocols from the wired world.

What follows is a brief exploration of ways in which those standard protocols can be used to help secure wireless devices.

Secure Socket Layer (SSL)
Outfitting a wireless device with an SSL-enabled browser is an effective means of providing secure access to web mail and the corporate intranet. Numerous small and medium-sized organizations already use SSL-based virtual private networks (VPNs) to provide access to corporate e-mail and intranet sites.

IPSec
Many medium-sized and large organizations use VPN gateways to control access to their enterprise networks. The VPN represents a secure extension of the LAN, and enables users to work as though on the LAN from virtually anywhere in the world. Extending this relatively unfettered connectivity to wireless users requires their devices to run a VPN client application with IPSec embedded.

CRYPTOGRAPHY AT A GLANCE

In the digital world, there are basically two types of cryptographic systems:

- **symmetric-key cryptosystems; and**
- **asymmetric (public-key) cryptosystems.**

In a symmetrical system, a single digital 'key' is used to both encrypt and decrypt data. The key is shared privately between users.

In a public-key system, encryption and decryption are handled separately. Encryption is done using a public key; decryption with a private key — hence the label 'asymmetric'.

While symmetric-key systems process data quite rapidly, public-key systems offer greater scalability. In practice, the two are often combined: the public side used to exchange a secret common key; the symmetric side used to 'do the math' — to perform bulk data encryption using the key exchanged.

S/MIME

S/MIME secures e-mail by storing it in encrypted format on mail servers, and by making messages readable only by their intended recipients. It enables non-repudiation by using digital signature to prove that the e-mail messages come from the person who claims to have sent it. These signatures carry the same validity as signed documents. For S/MIME to work on a mobile device, it must be supported by a wireless e-mail client or plug-in that interoperates with desktop e-mail clients and e-mail servers, and by digital certificates for digital signatures.

While each of these standard technologies has a potential role to play in wireless security, they all lead back to the issue of resource constraints. Unlike desktop PCs, wireless devices do not have the capacity to run multiple cryptographic services at once. Consequently, they must offer a single set of reliable, interoperable services that can be used to build *any* secure application: a common cryptographic service provider, if you will.

This reinforces the conclusion arrived at earlier — that embedded security is preferable to add-on security — and brings us back to the subject of cryptographic standards.

Security: the cryptographic core

Computationally intensive, cryptography has a fundamental impact on device performance, power and speed. The size and complexity of the algorithms used therefore exert a significant influence over the functional limitations of any built-in security set.

Until recently, the Data Encryption Standard (DES) and 3DES were the most widely deployed algorithms for symmetrical encryption; today, given recent advancements, the preferred choice is AES, the Advanced Encryption Standard. AES comes in three security strengths: 128 bits, 192 bits and 256 bits — a significant improvement over the 56 and 112 bits provided by DES and 3DES. As a result, the U.S. federal government has made AES its standard-setting security algorithm.

Good security practice dictates that keys for symmetric algorithms like AES must be matched in strength by asymmetric (public-key) algorithms such as RSA or Elliptic Curve Cryptography (ECC). In fact, this is a requirement for FIPS 140-2 validated products. As asymmetric keys tend to be much larger than symmetric keys, developers must choose carefully which algorithm to use. For constrained devices, ECC holds clear advantages.

NIST guidelines for public key sizes for AES

ECC KEY SIZE (Bits)	RSA KEY SIZE (Bits)	KEY SIZE RATIO	AES KEY SIZE (Bits)
163	1024	1 : 6	
256	3072	1 : 12	128
384	7680	1 : 20	192
512	15 360	1 : 30	256

Supplied by NIST to ANSI X9F1

Figure 2: *Size does matter: Relative key sizes, RSA vs. ECC*

As *Figure 2* shows, to match the security of a 128-bit AES symmetric key, RSA must generate a sizable 3,072-bit asymmetric key. ECC, on the other hand, scales linearly with AES and remains fairly compact at all security levels. For 128-bit AES security, ECC requires a key size of just 256 bits.

Computationally, the relative performance advantage of ECC versus RSA is indicated not by key size but rather by the cube of each key size. The difference becomes dramatic as the key sizes scale upward. An increase in RSA key sizes leads to considerable increase in computational cost. For example, progressing from a 1024-bit RSA key to a 3072-bit RSA key requires about 27 times (3^3) as much computation; the same increase in ECC key raise the computational cost by just over 4 times (1.6^3).

ECC presents advantages in terms of hardware resources as well, whether a system is space-optimized or speed-optimized. As *Figure 3* shows, the gate count for ECC is vastly lower than that required for RSA.

Comparing RSA-3072 and ECC-283 (128-bit security level)

Mode	**RSA-3072**	**ECC-283**
Space-optimized (same clock speed)	(VLSI Cores) 184 ms 50,000 gates	(Grobschädl) 29 ms (16 ms for Koblitz curve) 6,660 gates
Speed-optimized (same clock speed)	(VLSI Cores) 110 ms 189,200 gates	(Orlando and Paar) 1.3 ms 80,100 gates

Figure 3: *Hardware implications: RSA vs. ECC*

The benefits of ECC are ultimately many: linear scalability, a small software footprint, low hardware implementation costs, low bandwidth requirements, and high device performance. Standardized, it ensures interoperability between devices. ECC is also a well-researched and proven system, having been studied for nearly 20 years. As such, it answers any concerns developers or users may have about reliability.

For these reasons, ECC has gained the support of a number of leading companies – Motorola and Pitney Bowes among them – and of many accredited standards organizations, including:

- ISO (in ISO 14888-3: ECDSA and other ECC-based signature schemes)
- IEEE (in IEEE 1363-2000 for public-key cryptography)
- NIST (in FIPS 186-2)
- NIST (in SP 800-56: Special Publication on Key management)
- The American National Standards Institute (in ANSI X9: cryptography for financial-services industry)

Architecture

Having established that, for constrained devices, ECC is the most compact and practical asymmetric algorithm for security, we can proceed to the question of architecture – how to build the common cryptographic service provider (CSP) mentioned earlier.

In constrained devices, the ideal architecture has a cryptographic engine at its core and a common set of APIs capable of supporting a wide variety of security requirements, allowing developers to embed interoperable security protocols quickly and easily. Together, this engine and these APIs constitute the CSP.

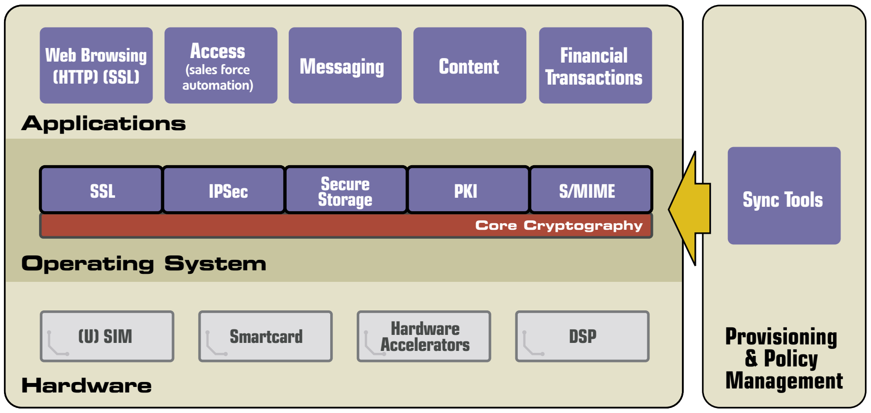

Figure 4: *Division of labor: sharing the security burden*

THE COMMON CRYPTOGRAPHIC SERVICE PROVIDER (CSP)

- **Simple API allows developers to focus on system-level issues; eliminates hardware security-integration process.**
- **Interoperates with proven standards and protocols, including SSL, IPSec, PKI and authentication standards.**
- **Cryptographic acceleration, secure key-storage and true random-number generation operations are all done on the chip.**
- **Offloading public key to DSP increases speed of operations.**
- **Achieves all of the above with extreme code efficiency and minimal footprint.**

While it is possible to build an entirely software-based CSP, performance can be dramatically enhanced by utilizing a combination of software and hardware elements. Experience has shown that certain approaches to hardware can improve security at very little cost, for example through:

- true hardware-based random-number generation;
- crypto hardware acceleration supporting DES, Triple DES, AES, SHA-1 and other standards-based security operations;
- a secure bootloader for device-code integrity including code signing; and
- a secure-execution mode enabling secure key storage and runtime authentication.

The upper-layer security protocols dealt with earlier – SSL, IPSec and S/MIME – change over time and are too big to implement in hardware. The same is true of specific authentication methods. Consequently, these components should be optimized to run on the target platform within the embedded operating system (OS) of a constrained device. With the CSP on the processor, security components in the operating system (such as algorithms, SSL, IPSec and PKI) execute natively at the hardware level. This enables them to take advantage of the many resources available in hardware, from acceleration to security-object storage.

Implementation

The final consideration for developers tasked with embedding security in wireless and other constrained devices relates to tools. For this particular set of tasks, what should a toolkit ideally provide?

First and foremost, the software must impose as small a footprint as possible to conserve device resources and leave ample room for other programming elements. This can be achieved in a number of ways: by offering the ability to compile out unused algorithms, for example, and by having a minimal code base to begin with – one that's optimized for the target platform.

Flexibility is another critical ingredient. To capitalize on the potential advantages of accelerated processing speed, longer battery life and higher bandwidth utilization, it's clear that an effective security toolkit should support standards-based, public key schemes such as ECC. But it should also support the implementation of RSA in order to not narrow developers' available range of security options.

Interoperability, as has been established throughout this paper, is absolutely necessary. Standards and protocols such as SSL, IPSec and PKI must be supported, as these are at the foundation of organizations' existing security infrastructures.

From a developer's perspective, of course, a toolkit should help reduce the complexity of the challenge at hand. Supporting multiple platforms is one way this can be done — again, leaving as many doors open as possible. Minimizing the number of APIs required is another. With a single API to work with, for example, developers are able to accelerate their progress and help get products to market sooner — an invaluable competitive advantage.

Conclusions

As has been shown, to meet the diverse needs of today's wireless users, security must become an embedded function of devices themselves. That security functionality must be interoperable, reliable, and as sparing as possible of system resources.

To meet the first two requirements, any built-in security solution must be standardized — open and proven effective. To meet the latter, it must be algorithmically and architecturally efficient.

By incorporating a common cryptographic service provider (CSP) augmented with the right APIs, developers can take full advantage of a constrained system's hardware and software resources to perform computationally intensive cryptographic functions. Choosing the best algorithm for a given situation is also necessary; and for the most part, where constrained devices are concerned, ECC is the most compact and practical asymmetric algorithm for security.

It's clear that security is an essential component of wireless communication today — and will continue to be for the long term. What device manufacturers need to deliver that security is, in the end, a trusted platform capable of developing simple *or* sophisticated applications.

Other Certicom "Got Security?" White Papers

Many Happy Returns: The ROI of Embedded Security

Welcome to the Real World: Embedded Security in Action

These white papers are all available from **www.certicom.com/gotsecurity**

About Certicom

Certicom is a leading provider of wireless security solutions, enabling developers, governments and enterprises to add strong security to their devices, networks and applications. Designed for constrained devices, Certicom's patented technologies are unsurpassed in delivering the strongest cryptography with the smallest impact on performance and usability.

Contact Certicom

Corporate Headquarters

5520 Explorer Drive
Mississauga, Ontario
L4W 5L1
Tel: +1-905-507-4220
Fax: +1-905-507-4230
E-mail: info@certicom.com

Sales Offices

Canada

5520 Explorer Drive
Mississauga, Ontario
L4W 5L1
Tel: 905-507-4220
Fax: 905-507-4230
E-mail: info@certicom.com

Ottawa

84 Hines Road
Kanata, Ontario
K2K 3G3
Tel: +1-613-254-9270
Fax: +1-613-254-9275

US Western Regional Office

1810 Gateway Drive, Suite 220
San Mateo, CA 94404
Tel: +1-650-655-3950
Fax: +1-650-655-3951
E-mail: sales@certicom.com

US Eastern Regional Office

1175 Herndon Parkway, Suite 750
Herndon, Virginia 20170
Tel: +1-571-203-0700
Fax: +1-571-203-9653
E-mail: sales@certicom.com

Europe

Golden Cross House
8 Duncannon Street
London WC2N 4JF UK
Tel: +44 20 7484 5025
Fax: +44 20 7484 5150

www.certicom.com