

# Usable Security on Sun SPOTs

Vipul Gupta, Iljya Kalai, Sun Microsystems Laboratories  
 {vipul.gupta, iljya.kalai}@sun.com

<http://www.sunspotworld.com/>

## 1. Introduction

Next-generation, wireless computing devices exemplified by the Sun SPOTs have the potential to revolutionize a wide array of applications ranging from asset tracking to proactive health care to military surveillance. Several of these applications have strong security requirements. Security mechanisms must be efficient and user-friendly to ease adoption, especially amongst users that aren't crypto-savvy.

## 2. Goals

"Over-the-air" is the way of life for Sun SPOTs and similar devices and security is a critical requirement. Unfortunately, for most users, convenience trumps security.

Hence,

- Security mechanisms need to be efficient (low overhead)
- The hard part, trust management, should "just work" in the simplest, most common scenarios

We've developed a highly efficient, pure Java cryptographic library which supports key exchange and digital signatures based on Elliptic Curve Cryptography (ECC), and standard algorithms for hashing and bulk encryption.

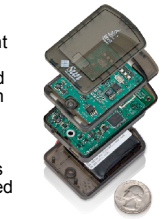


Fig 1. Sun SPOT

We've also developed user-friendly mechanisms for managing the keys used in code signing and secure communication.

## 3. Secure Code Deployment

- Digital signature used to "seal" verified byte codes on a desktop.
- Code sent over-the-air or via USB to Sun SPOT
- Signature verified on SPOT before execution
- Addresses (i) Java branding requirements (ii) General code authentication

Key management for secure code deployment:

- Balances security, convenience
- Mostly user-transparent; simple "ownership" model
- Each user's SDK has a public/private key-pair
- SPOT stores trusted public-key of "owner"
- First deployer becomes "owner"
- Owner has special privileges (can deploy new code or restore SPOT to ownerless state)
- Policy permits ownership change with physical access

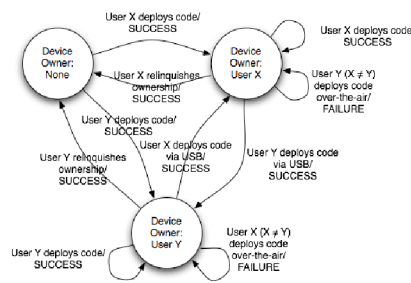


Fig 2. Device Ownership Model.

## 4. Secure Epidemic Code Deployment

- Pipelines code dissemination across a multi-hop network
- Data broken down into chunks with each chunk split across multiple packets.
- Protocol tolerates node failure and packet loss
- Each chunk can be verified as soon as it is received (without waiting for subsequent chunks).
- Verification based on hash-chains
- Node compromise does not permit unauthorized code dissemination
- User only updates version info in MANIFEST.MF and deploys to one node. Code spreads epidemically to other nodes in the network.

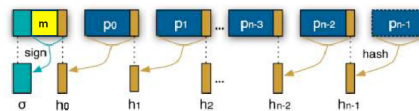


Fig 3. Hash chain implementation.

## 5. Secure Communication

- Inspired by the simplicity of "https"
- Applications requires a simple one-letter change "radiostream" to "sradiostream"
- Reuses SSL protocol underneath with ECC
- Certificate management is user-transparent in typical usage, extends key management from ownership model



SDK requests SPOT's Public Key

SPOT generates a key pair  $Pub_{SPOT}$  and  $Priv_{SPOT}$  sends  $Pub_{SPOT}$

SDK creates an X.509 certificate  $Cert_{SPOT}$  with  $Pub_{SPOT}$  and signs it using its private key  $Priv_{SDK}$ . SDK sends  $Cert_{SPOT}$  its own certificate  $Cert_{SDK}$

$Cert_{SPOT}$  stored in key store as the SPOT's personal certificate,  $Cert_{SDK}$  stored in key store as the owner's certificate

Fig 4. Key Management for Secure Communication

Key management for Secure Communication:

- SPOT uses  $Cert_{SPOT}$  to identify itself in an SSL handshake
- Each SPOT trusts its owner SDK to issue SPOT certificates
- Secure communication between SPOTs belonging to the same owner "just works"
- Additional scenarios also supported, e.g. Trusting web Certification Authorities and SPOTs belonging to other owners.

## 6. Summary

Well known cryptographic algorithms can be leveraged for security services such as authentication, data integrity and confidentiality. The hard part is managing keys and establishing trust relationships. We've developed simple techniques to accomplish this in a manner that is user-transparent in the most common case. This makes it easy even for non crypto-savvy users to use Sun SPOTs in a secure fashion.

## References

- [1] N. Gura et al., "Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs", CHES 2004, Aug. 2004.
- [2] F. Stajano and R. Anderson, "The Resurrecting Duckling: Security Issues in Adhoc Wireless Networks", in Proceedings of the 7th International Workshop on Security Protocols, April 1999.
- [3] V. Gupta et al., "User-friendly key management for secure code deployment", Research Disclosure Journal, No. 510036, Oct. 2006.
- [4] P. Lanigan et al., "Sluice: Secure Dissemination of Code Updates in Sensor Networks", in Proceedings of the 26th IEEE International Conference on Distributed Computing Systems, 2006.