Common Interface to Cryptographic Modules (CICM)

IETF 81 BOF

Lev Novikov

Terminology

- Crypto / Module
  - Hardware device that performs cryptographic operations
- Security Domain
  - System or group of systems operating under a common security policy.
- High Assurance
  - Enforcement of the separation of security domains. Typically used in government, diplomatic and military contexts
- Interface
  - API (unless specified otherwise)

Problem

- Existing cryptographic API standards do not address the needs of **high assurance** environments.
- Proprietary high assurance APIs do not enable implementations to interoperate.
- Some folks who have this problem:
  - Governments, Militaries, Diplomats
  - Multinational Organizations (e.g., NATO, EDA, ASEAN)
  - High Assurance Crypto Manufacturers

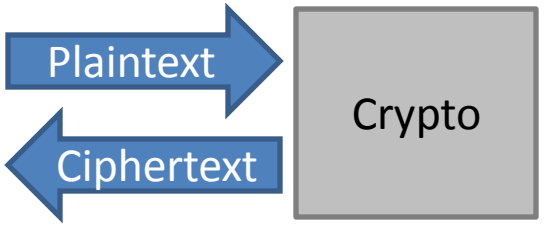
Logical Model

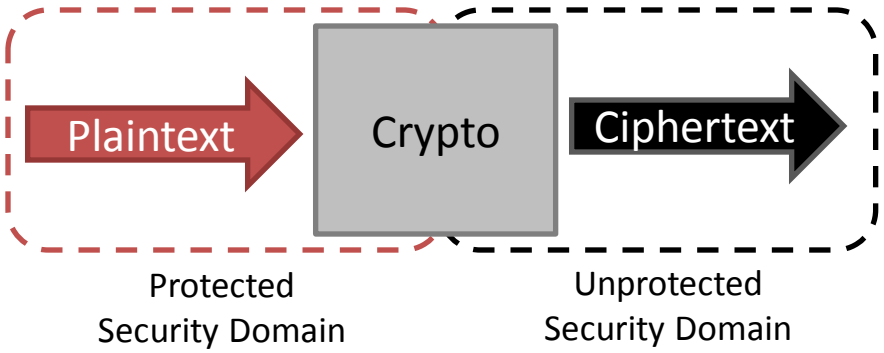
- Three major areas in which high assurance crypto modules differ in their logical model:
  - Cryptographic Transforms (Channels)
  - Cryptographic Key Material Management
  - Crypto (Module) Management

- Some other needs:
  - Avoiding dead code
  - Lock-step object creation

Cryptographic Transforms (Channels)

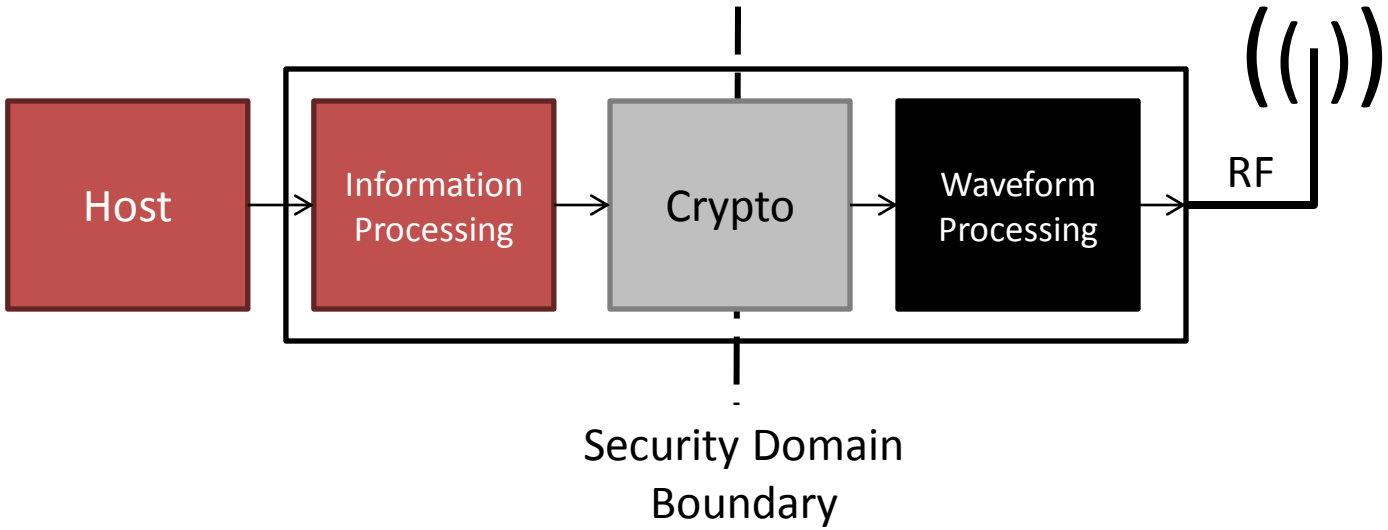
- Existing APIs assume:
  - Plaintext input
  - Ciphertext output
- High Assurance APIs:
  - Transition in Security Domain





Prototypical Use Case

- Tactical Secure Radio



Module/Key Management

- High assurance APIs require richer module and key management primitives.
- Examples:
  - Different login mechanisms (e.g., specialized hardware access tokens)
  - Running built-in tests
  - Managing module-internal logs
  - Securely updating module software
  - Flexible mechanisms to create or import key material

Other Needs

- Dead Code Elimination
  - Namespaces and Conformance rules help implementers select appropriate subset of objects/methods to implement
- Lock-Step Object Creation
  - Method parameters designed to force prerequisites (keys, algorithms, etc.)
    - Non-normative Extensions allowed; documented using Conformance rules

Some Existing APIs

- We looked at several APIs
  - Limited key / module management support
  - **Biggest Issue:** None support domain separation
    - results of crypto operations return to same process

- GSS-API
  - creating a shared context with authenticated credentials

- RSA PKCS#11
- Java Cryptography Architecture
- Microsoft CAPI
  - treat crypto as a local device / service provider