Cryptosense

GRAHAM STEEL

graham@cryptosense.com

The Problem with Cryptography

Crypto is:

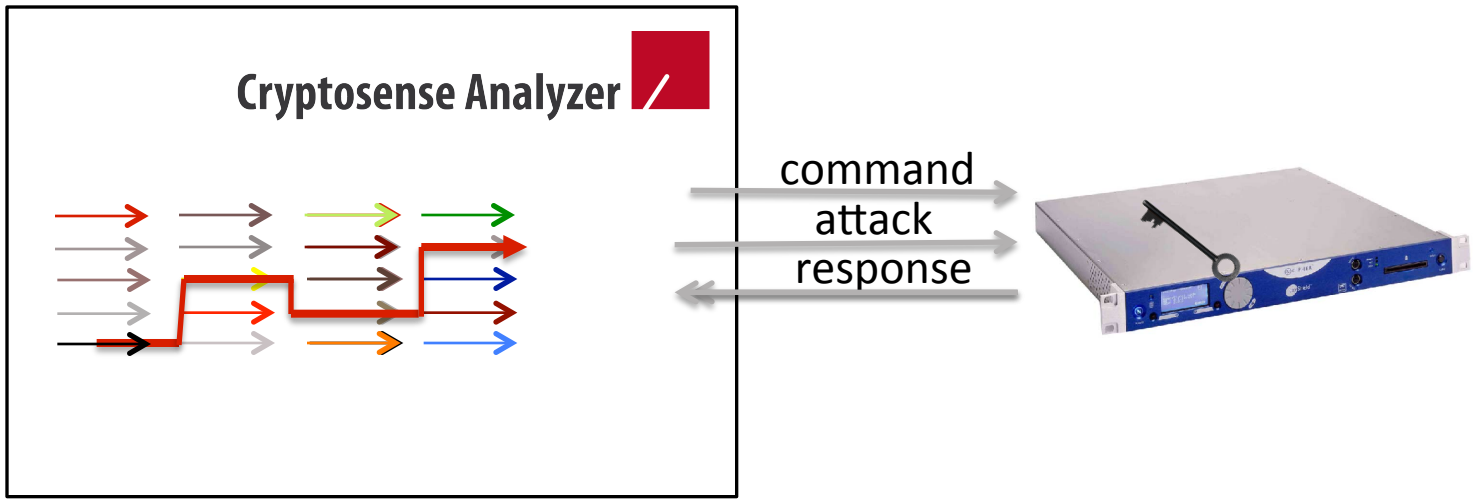
- Ubiquitous
- Powerful
- Complex
- Fragile

How can a company find and remove weaknesses, and then demonstrate its systems are secure?

Solution

- **Cryptosense**: software tools for vulnerability management of cryptographic systems.
- Vision: treat **all enterprise crypto**
- First step: focusing on **HSMs** and associated applications.
- **Cryptosense Analyzer** for PKCS#11 HSMs in use at two European national security agencies and two top five European banks.
- Currently developing/testing Thales (payment API), MS CAPI/CNG and OpenSSL versions.

How it Works



Vulnerability scanning vs Cryptosense

Standard vuln scanner	Cryptosense
Checks “fingerprint” of system against database of known vulns	Infers a model of specific system and config under test, searches this model for “new” attacks
Typically many false positives + negatives	Attacks found are executable, all attacks of a particular class can be found
Not useful for custom apps	Works just fine on custom apps
Usually accesses a cloud-based DB of vulns	Autonomous: no need to connect to an external network

Vulnerabilities Cryptosense Analyzer finds

1. Implementation discrepancies

   PKCS#11 is a 407 page document with 303 footnotes, many of which are security critical (e.g. CVE-2010-3321)

2. Configuration problems

   Even correct implementations may expose *combinations* of commands that are insecure.

3. Application incoherencies

   Even if the HSM is secure, the application can misuse it.

Cryptosense solution

1. Security audit of existing infrastructure
   - using Analyzer
2. Secure configuration
   - using App Tracer to check applications
3. Regular automatic testing of HSM infrastructure
   - using Cryptosense Monitor
   - to prove to auditors (internal or external) that risks are identified, controls are in place and implemented (weekly reports)

Compliance testing

```
  Require CKA_NEVER_EXTRACTABLE is unspecified:                                  100% (5 tests, 0 failed)
    Reference: v2.20 p82-85 s10.9-10.10
  Require CKA_ALWAYS_SENSITIVE is unspecified:                                   100% (3 tests, 0 failed)
    Reference: v2.20 p82-85 s10.9-10.10
  Cannot set CKA_TRUSTED to CK_TRUE:                                             100% (2 tests, 0 failed)
    Reference: v2.20 p66,73,81,85 s10.2,10.6.2,10.8,10.10
  Ensure CKA_ALWAYS_SENSITIVE is CK_FALSE:                                        98% (86 tests, 1 failed)
    Reference: v2.20 p128 s11.7
  Ensure CKA_NEVER_EXTRACTABLE is CK_FALSE:                                      100% (86 tests, 0 failed)
    Reference: v2.20 p128 s11.7
  Ensure CKA_LOCAL is CK_FALSE:                                                  100% (86 tests, 0 failed)
    Reference: v2.20 p79 s10.7
  Require CKA_KEY_TYPE is specified:                                             100% (50 tests, 0 failed)
    Reference: v2.20 p79 s10.7
C_CreateObject (DES3)
  Template matches:                                                              100% (87 tests, 0 failed)
    Reference: v2.20 p63 s10.1.1
  Default value of CKA_TOKEN is CK_FALSE:                                        100% (63 tests, 0 failed)
    Reference: v2.20 p71 s10.4
  Default value of CKA_MODIFIABLE is CK_TRUE:                                    100% (66 tests, 0 failed)
    Reference: v2.20 p71 s10.4
  Require CKA_LOCAL is unspecified:                                              100% (45 tests, 0 failed)
    Reference: v2.20 p79 s10.7
  Require CKA_KEY_GEN_MECHANISM is unspecified:                                  100% (55 tests, 0 failed)
    Reference: v2.20 p80 s10.7
  CKA_KEY_GEN_MECHANISM has a value if and only if CKA_LOCAL is CK_TRUE:          N/A (untested)
    Reference: v2.20 p80 s10.7
  Require CKA_NEVER_EXTRACTABLE is unspecified:                                  100% (2 tests, 0 failed)
    Reference: v2.20 p82-85 s10.9-10.10
```

Get a free demo at http://cryptosense.com

PKCS#11 Future

In early 2013 PKCS#11 moved from RSA to OASIS.

v2.40 now in public draft at http://docs.oasis-open.org/pkcs11

Some significant changes to available crypto - see blog articles at http://cryptosense.com/tag/pkcs11

v3.0 (2015) brief is to address key protection, multiple user profiles/authentication, updated crypto mechanisms..

W3C Web Crypto API

"..expose trusted cryptographic primitives from the browser. This will promote higher security insofar as Web application developers will no longer have to create their own or use untrusted third-party libraries for cryptographic primitives."

April 2012: Group Formation

April 2014: Last call on v1.0

October 2013: Exit Last Call (?)

Jan 2015: Expected Recommendation (?)

Cryptosense W3C API Tracer


```
{
 command : importKey
 input : {
    format : raw
    keyData : 0x00112233445566778899AABBCCDDEEFF
    algorithm : {
       name : AES-CBC
    }
    extractable : true
    keyUsages : [
    ◦ unwrapKey
    ◦ wrapKey
    ◦ decrypt
    ◦ encrypt
    ]
 }
-output : {
    usages : [
    ◦ unwrapKey
    ◦ wrapKey
    ◦ decrypt
    ◦ encrypt
    ]
   -algorithm : {
      -length : 128
       name : AES-CBC
    }
    extractable : true
    type : secret
    creationDate : 2014-08-04T14:56:20.186Z
    id : 1
    used : [
    ◦ 0
    ◦ 0
    ◦ 0
    ◦ 0
    ◦ 0
    ◦ 0
    ◦ 0
    ◦ 0
    ◦ 0
    ]
 }
}
```


Get it from https://github.com/cryptosense

More information

Video demo and various details at

cryptosense.com

White paper on PKCS#11 vulnerabilities available

graham@cryptosense.com