Cryptosense

Security Analysis for Cryptographic APIs

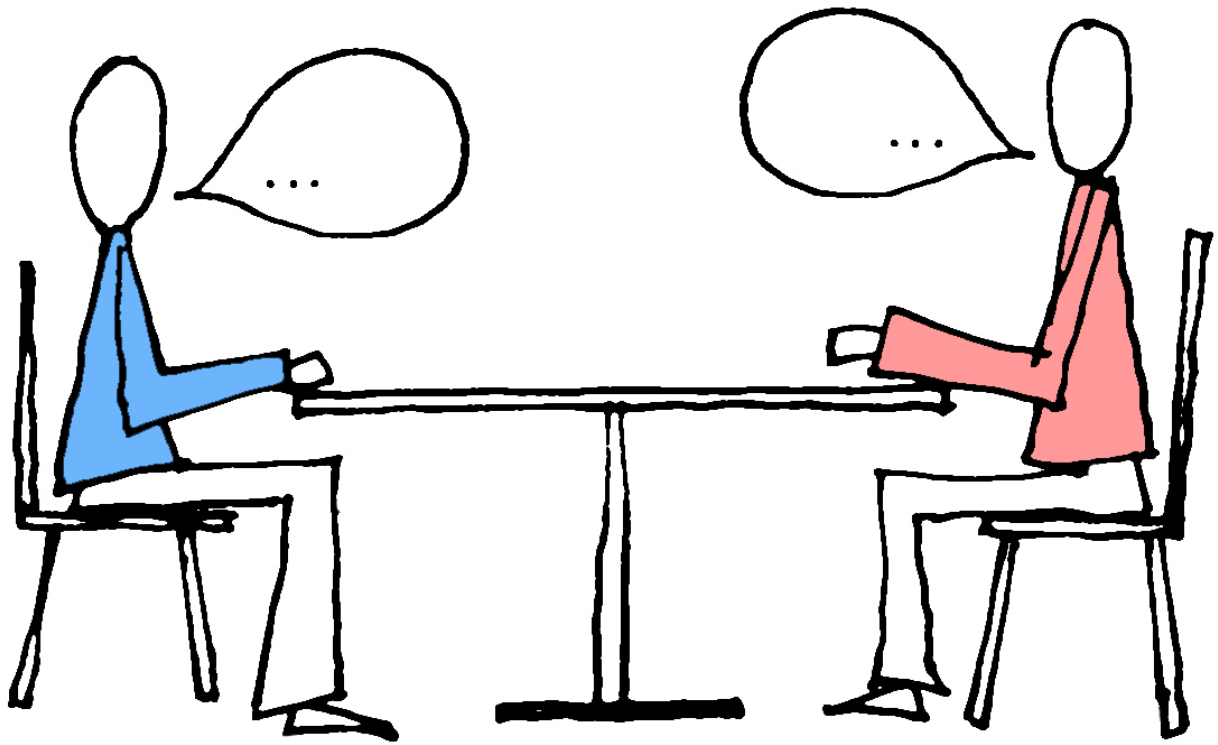
Graham Steel

Cryptographic Security Today

Companies use more and more crypto to protect sensitive data, offer new business models, comply with regulations,..

- Crypto algorithms are now very powerful
- But cryptographic systems are complex : hard to configure securely
- And they are fragile : one mistake enough to create a vulnerability

Cryptography in Practice v1

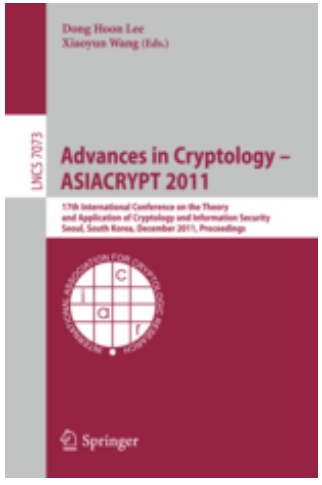


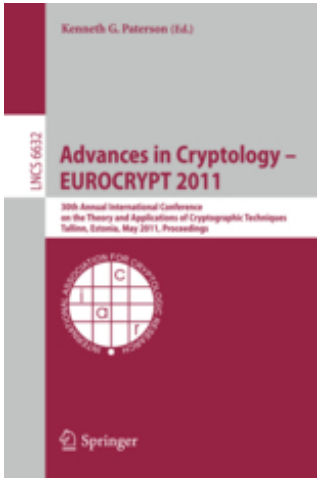
PM talks to client to understand security goals and threats

Cryptography in Practice v1

Engineer and PM discuss,
decide formal requirements of crypto schemes

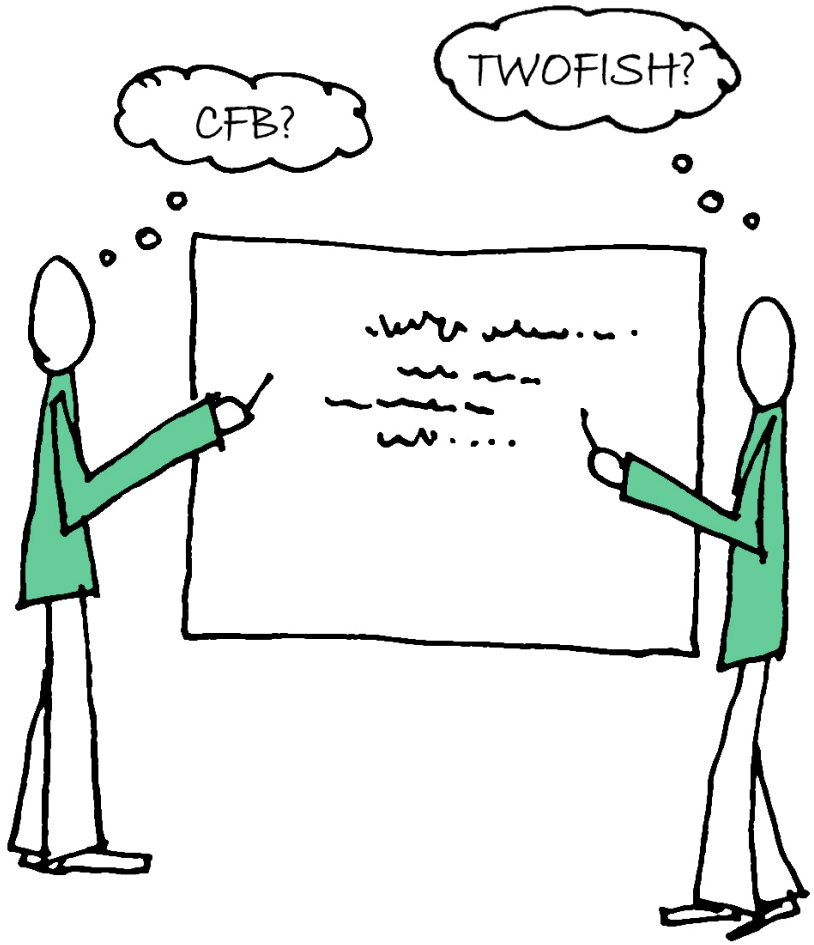
Cryptography in Practice v1





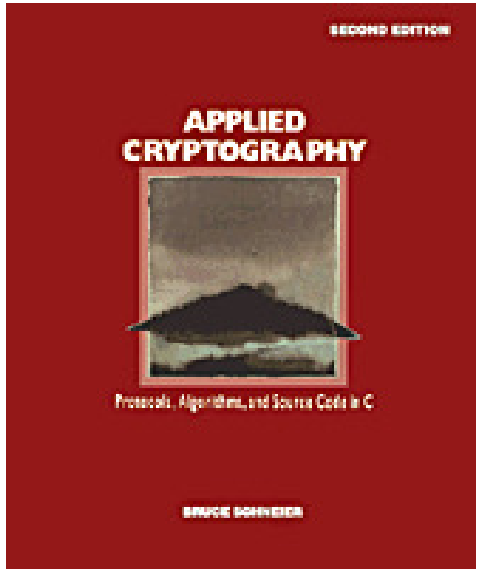
Engineer consults relevant literature for provably secure scheme

Cryptography in Practice v2



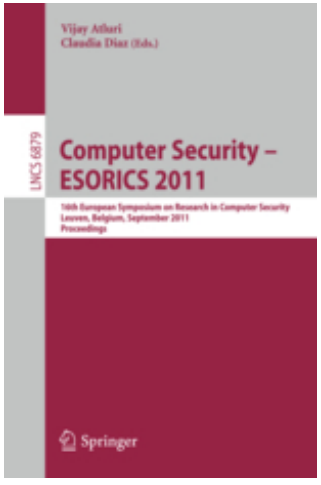
Engineers get together and plan some fun crypto stuff

Cryptography in Practice v2



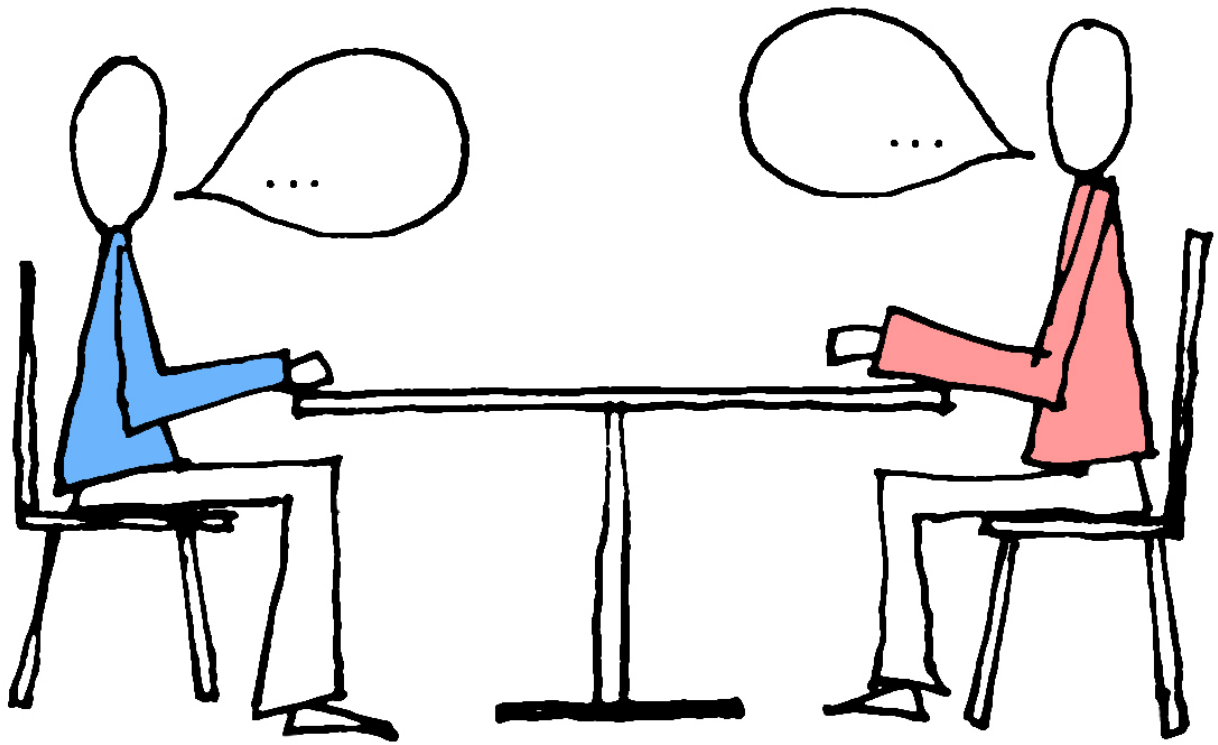
Literature consulted

Cryptography in Practice v2



Literature augmented by attacks on the system

Cryptography in Practice v3



PM instructed by client : “Solution must use PKCS#11 hardware”

Cryptography in Practice v3

PM instructs engineer : “Use PKCS#11 API”

Cryptography in Practice v3

?

RSA Public Key Cryptography Standard (PKCS) 11

PKCS #1 describes the RSA encryption algorithm, padding etc.

PKCS#11 Describes 'cryptoki' : cryptographic token interface

Ubiquitous in industry for authentication tokens, smartcards (and HSMs, other devices, ...)

Version 1.0 of PKCS#11 1995, current version 2.20 2004

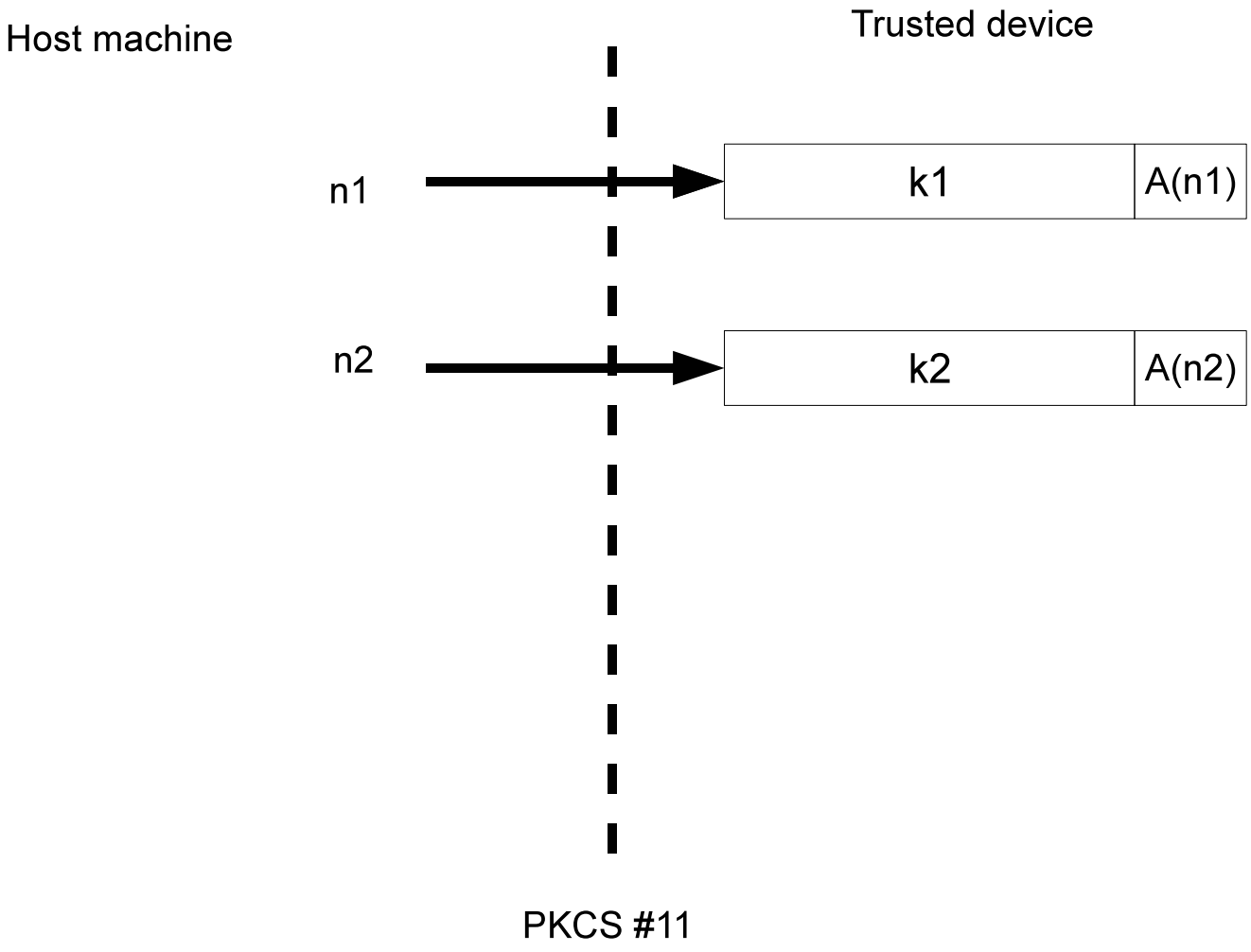
Sessions, PINs, Handles, Attributes

Application programs open a *session* with a token

Opening a session requires a *PIN*

Keys (and other objects) accessed by *handles*

Keys have *attributes* to control usage


Host machine
Trusted device
n1
k1
A(n1)
n2
k2
A(n2)
PKCS #11

Generating keys with PKCS#11

A *key template* is a partial specification of *key attributes*
Templates are used for creating, manipulating, and searching for objects

C_GenerateKey :

$$\mathcal{T} \xrightarrow{\text{new n,k}} \mathsf{h}(\mathsf{n},\mathsf{k});\mathsf{T}$$

Setting Key Attributes

$$\text{C\_SetAttributeValue} : \mathcal{T}, \mathsf{h}(\mathsf{n}, \mathsf{k}) \rightarrow \mathsf{h}(\mathsf{n}, \mathsf{k}); \mathsf{T}$$

$\mathcal{T}$ can specify new values for any attributes, but may cause `CKR_TEMPLATE_INCONSISTENT`, `CKR_ATTRIBUTE_READ_ONLY`

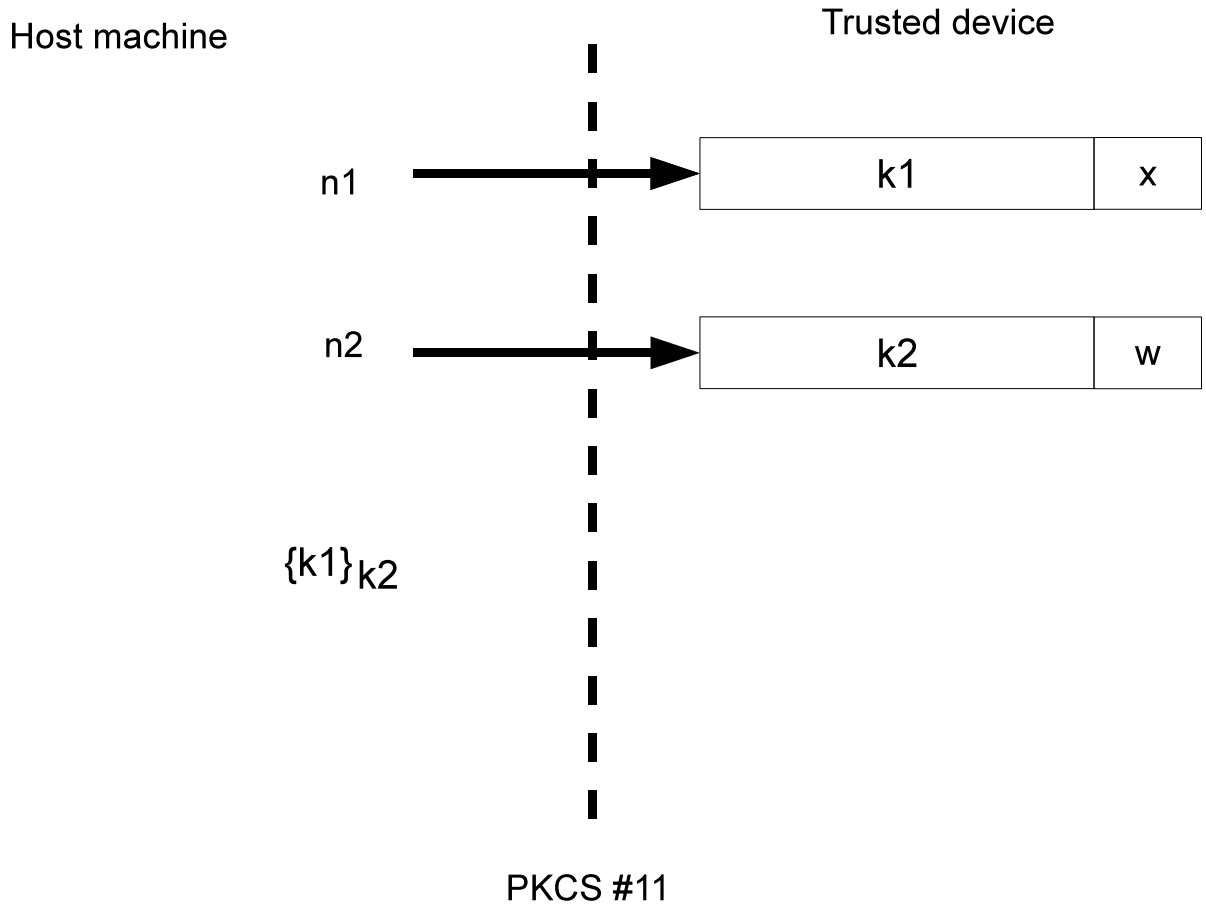
Wrap and Unwrap

Wrap :

$$h(x_1, y_1), h(x_2, y_2);\ \mathsf{wrap}(x_1), \mathsf{extract}(x_2) \quad \rightarrow \quad \{y_2\}_{y_1}$$

Unwrap :

$$h(x_2, y_2), \{y_1\}_{y_2}, \mathcal{T};\ \mathsf{unwrap}(x_2) \xrightarrow{\text{new } n_1} h(n_1, y_1);\ \mathsf{extract}(n_1),\ T$$


Host machine
Trusted device
n1
k1
x
n2
k2
w
{k1}k2
PKCS #11

Key Usage

Encrypt :

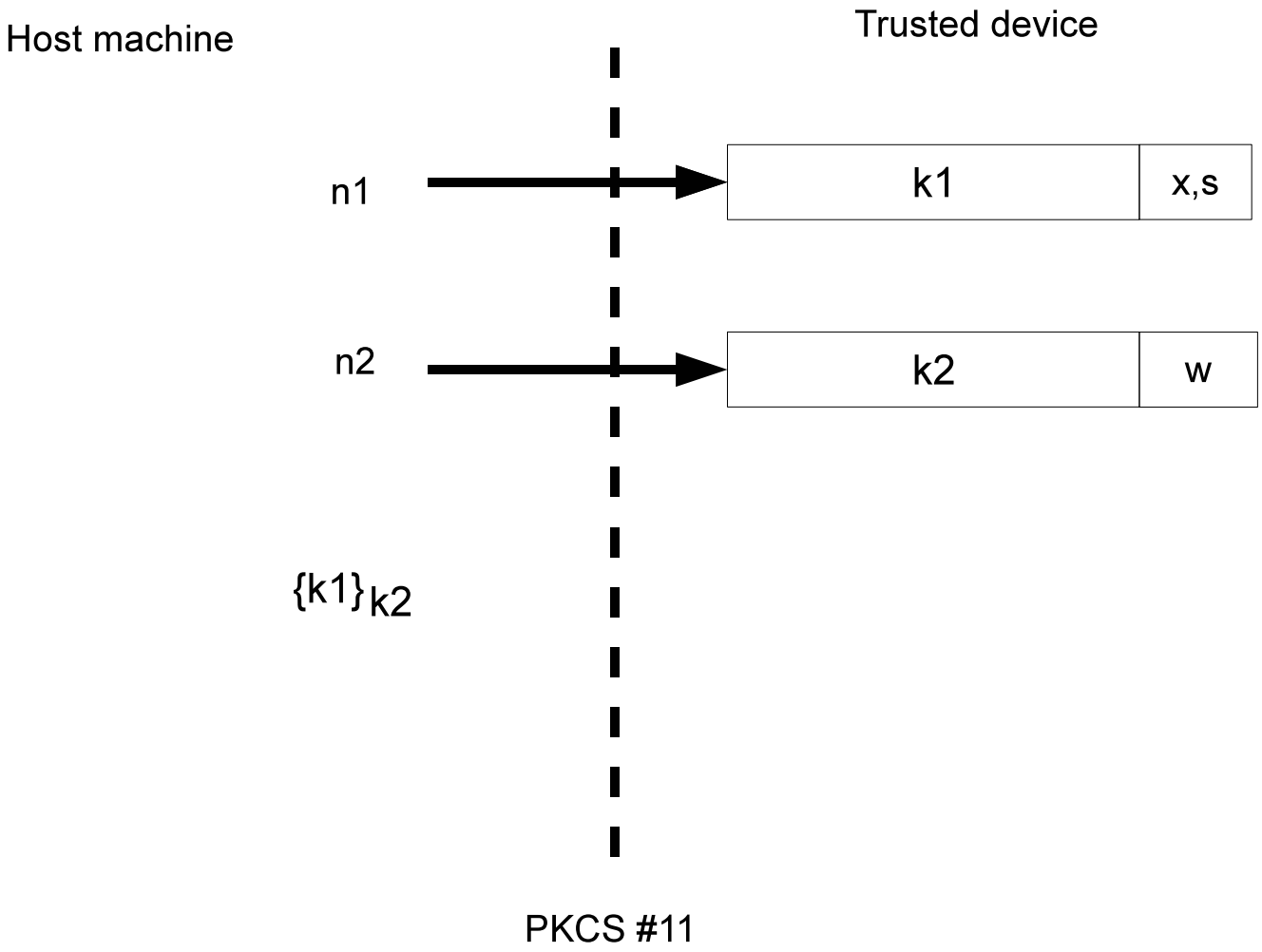
$$h(x_1, y_1), y_2;\ \text{encrypt}(x_1) \quad \rightarrow \quad \{y_2\}_{y_1}$$

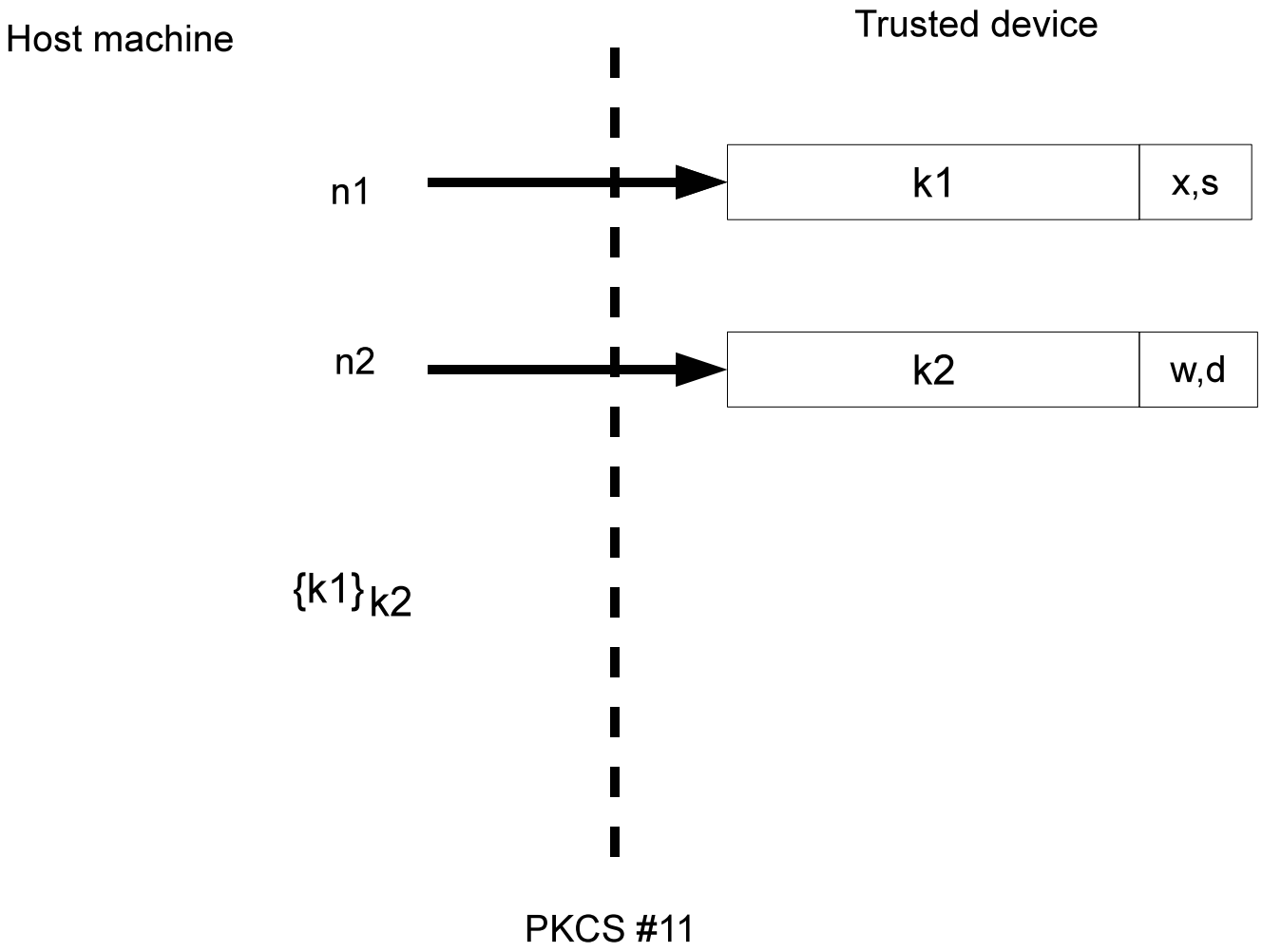
Decrypt :

$$h(x_1, y_1), \{y_2\}_{y_1};\ \text{decrypt}(x_1) \quad \rightarrow \quad y_2$$

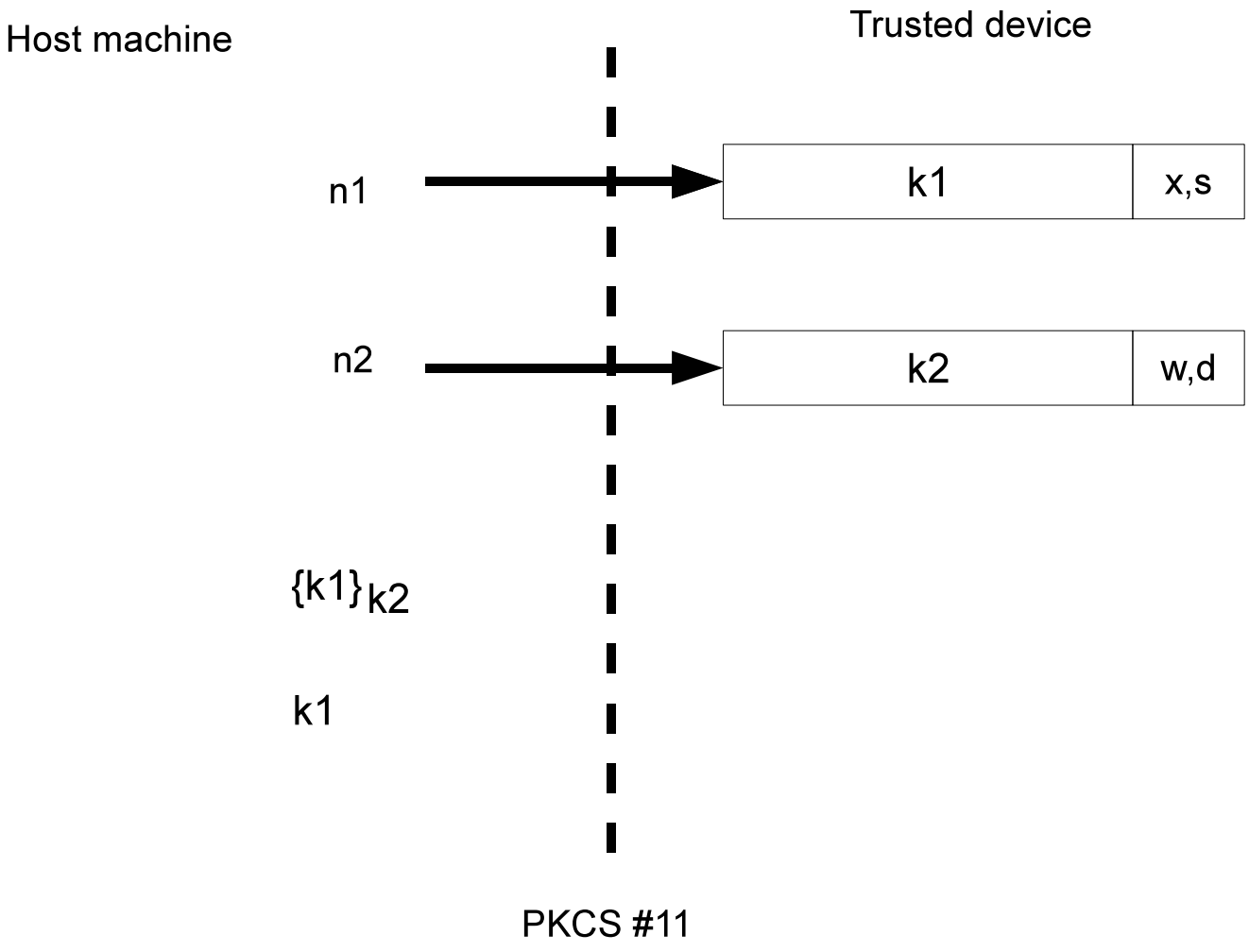
PKCS#11 Security

Section 7 of standard :
"1. Access to private objects on the token, and possibly to cryptographic functions and/or certificates on the token as well, requires a PIN.
2. Additional protection can be given to private keys and secret keys by marking them as "sensitive" or "unextractable". Sensitive keys cannot be revealed in plaintext off the token, and unextractable keys cannot be revealed off the token even when encrypted"
"Rogue applications and devices may also change the commands sent to the cryptographic device to obtain services other than what the application requested [but cannot] compromise keys marked "sensitive," since a key that is sensitive will always remain sensitive. Similarly, a key that is unextractable cannot be modified to be extractable."


Host machine
Trusted device
n1
k1
x,s
n2
k2
w
{k1}k2
PKCS #11


Host machine
Trusted device
n1
k1
x,s
n2
k2
w,d
{k1}k2
PKCS #11

Clulow, CHES 2003



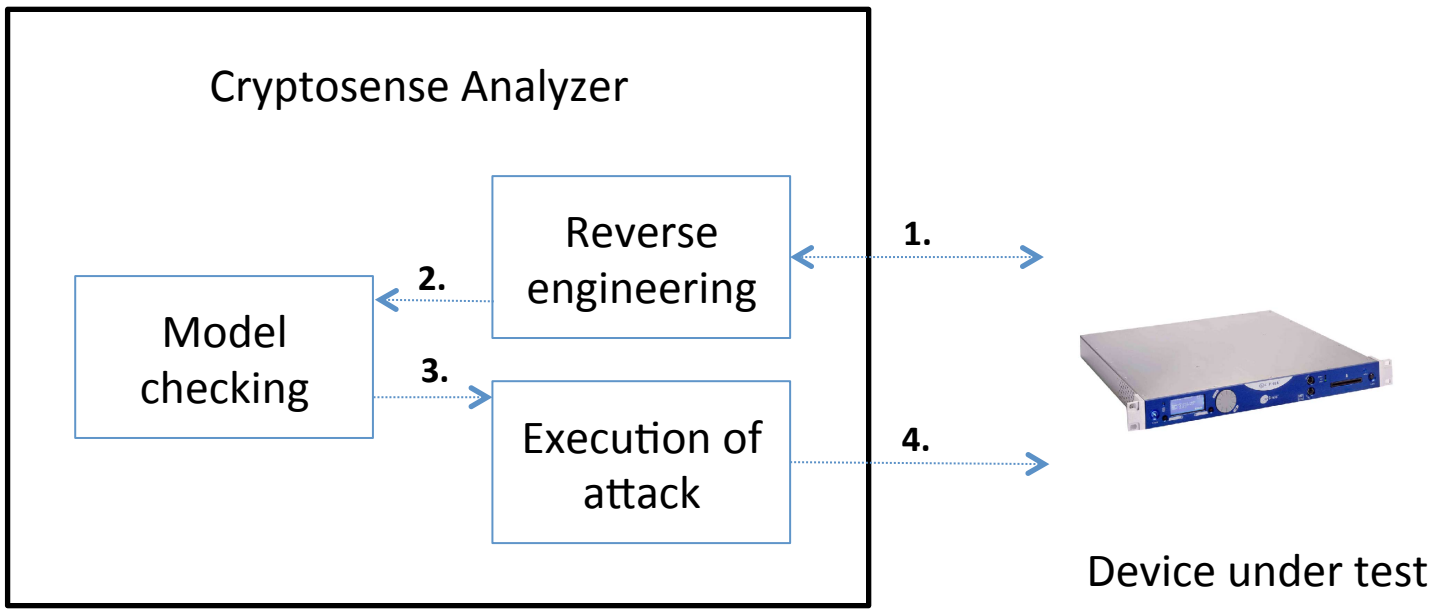
Prevent generation of decrypt and wrap keys..

Intruder knows : $h(n_1, k_1)$, $h(n_2, k_2)$, k_3
State : $\text{sensitive}(n_1), \text{extract}(n_1)$, $\text{extract}(n_2)$

Set_wrap :	$h(n_2, k_2)$	$\rightarrow$	$; \text{wrap}(n_2)$
Set_wrap :	$h(n_1, k_1)$	$\rightarrow$	$; \text{wrap}(n_1)$
Wrap :	$h(n_1, k_1), h(n_2, k_2)$	$\rightarrow$	$\{k_2\}_{k_1}$
Set_unwrap :	$h(n_1, k_1)$	$\rightarrow$	$; \text{unwrap}(n_1)$
Unwrap :	$h(n_1, k_1), \{k_2\}_{k_1}$	$\xrightarrow{\text{new } n_3}$	$h(n_3, k_2)$
Wrap :	$h(n_2, k_2), h(n_1, k_1)$	$\rightarrow$	$\{k_1\}_{k_2}$
Set_decrypt :	$h(n_3, k_2)$	$\rightarrow$	$; \text{decrypt}(n_3)$
Decrypt :	$h(n_3, k_2), \{k_1\}_{k_2}$	$\rightarrow$	k_1

[Delaune, Kremer & S., CSF 2008]

Cryptosense Analyzer

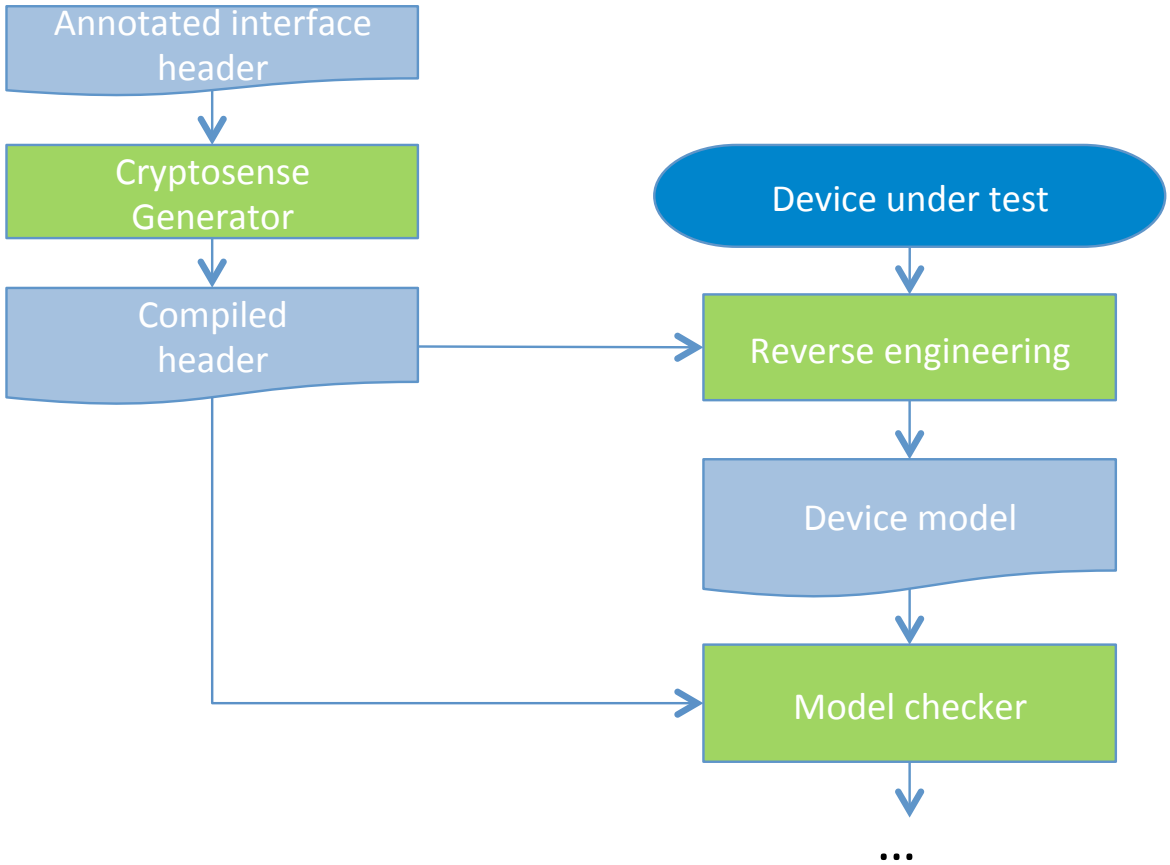


Brand	Device Model	Supported Functionality						Attacks found				Tk
		s	as	cobj	chan	w	ws	wd	rs	ru	su	
Aladdin	eToken PRO	✓	✓	✓	✓	✓	✓	✓				wd
Athena	ASEKey	✓	✓	✓								
Bull	Trustway RCI	✓	✓	✓	✓	✓	✓	✓				wd
Eutron	Crypto Id. ITSEC		✓	✓								
Feitian	StorePass2000	✓	✓	✓	✓	✓	✓	✓	✓	✓		rs
Feitian	ePass2000	✓	✓	✓	✓	✓	✓	✓	✓	✓		rs
Feitian	ePass3003Auto	✓	✓	✓	✓	✓	✓	✓	✓	✓		rs
Gemalto	SEG		✓		✓							
MXI	Stealth MXP Bio	✓	✓		✓							
RSA	SecurID 800	✓	✓	✓	✓				✓	✓	✓	rs
SafeNet	iKey 2032	✓	✓	✓		✓						
Sata	DKey	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	rs
ACS	ACOS5	✓	✓	✓	✓							
Athena	ASE Smartcard	✓	✓	✓								
Gemalto	Cyberflex V2	✓	✓	✓		✓	✓	✓				wd
Gemalto	SafeSite V1		✓		✓							
Gemalto	SafeSite V2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	rs
Siemens	CardOS V4.3 B	✓	✓	✓		✓				✓		ru

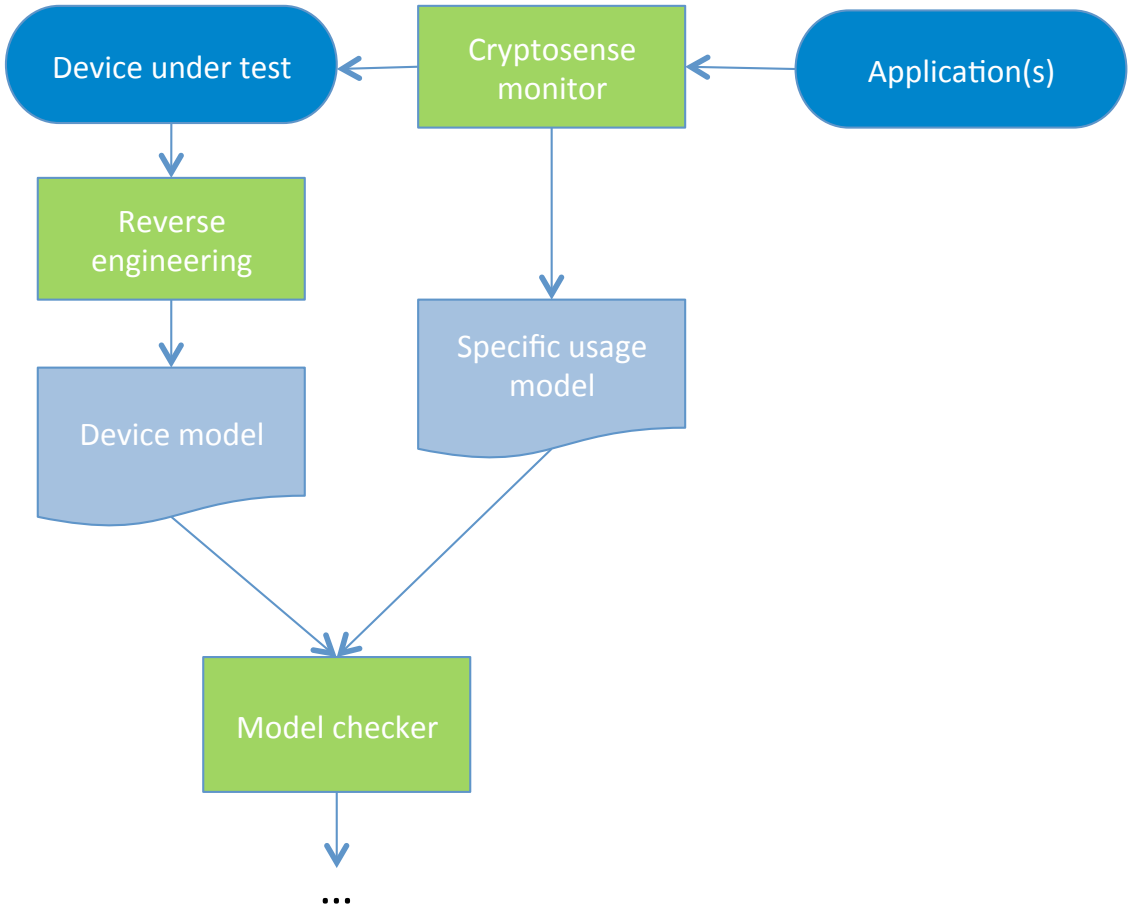
Cryptosense : from prototype to spin-off

- We were approached by a large aircraft manufacturer for a prototype in 2010. Then a major European bank in 2011.
- Analyzer was deployed in testing PKCS#11-compatible HSMs, which have a more sophisticated attribute policy, and support many configuration options
- INRIA funded a complete reimplementation by an engineer (two man-years) and some market research activities
- The spin-off company was created in September 2013 with 250k€ of funding (French Ministry of Research Prize)
- Working on MS-CAPI/CNG, Java JCA/JCE, OpenSSL engine, some proprietary APIs
- Ambitition to treat all enterprise crypto APIs

Cryptosense Generator



Cryptosense Monitor



Lessons Learned about Spin-offs

(even though we are only at the beginning of the story..)

- Everything takes much longer than you think
- More important to listen than to talk
- Lots of distractions from most important work (product/market fit)
- Thick skin required

Summary

- We started with a research project on RSA PKCS#11. Found many attacks, many approaches to securing.
- Developed Cryptosense Analyzer : an automated audit tool. Got enough traction to persuade us to create a spin-off.
- Working on other common APIs, monitor and generator, to expand to a more widely useful tool.
- Company created in September 2013, now at the Agoranov incubator in Paris with 30 other tech startups.

`www.cryptosense.com`

(we're recruiting..)