Attacking and Fixing PKCS#11 Security Tokens with Tookan

Graham Steel

LSV, INRIA & CNRS & ENS-Cachan

(joint work with Riccardo Focardi, Matteo Bortolozzo & Matteo Centenaro, Università Ca' Foscari, Venezia)


RSA
SecurID
acs
IRONKEY
ePass2000

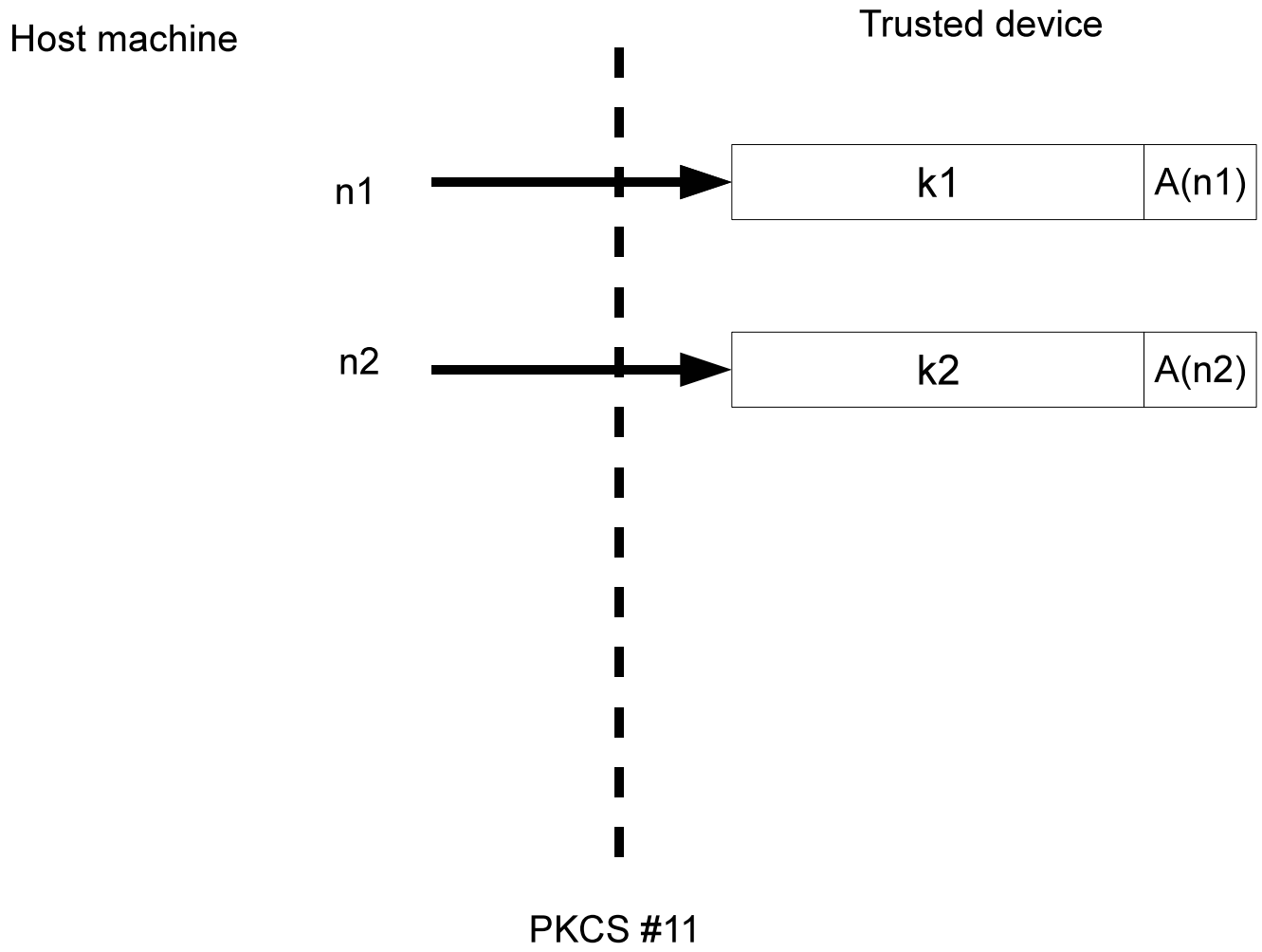
RSA Public Key Cryptographic Standard (PKCS) 11

Describes ‘cryptoki’: cryptographic token interface

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

Keys (etc.) stored on the device and accessed by *handles*

Attributes stored with keys to control usage


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

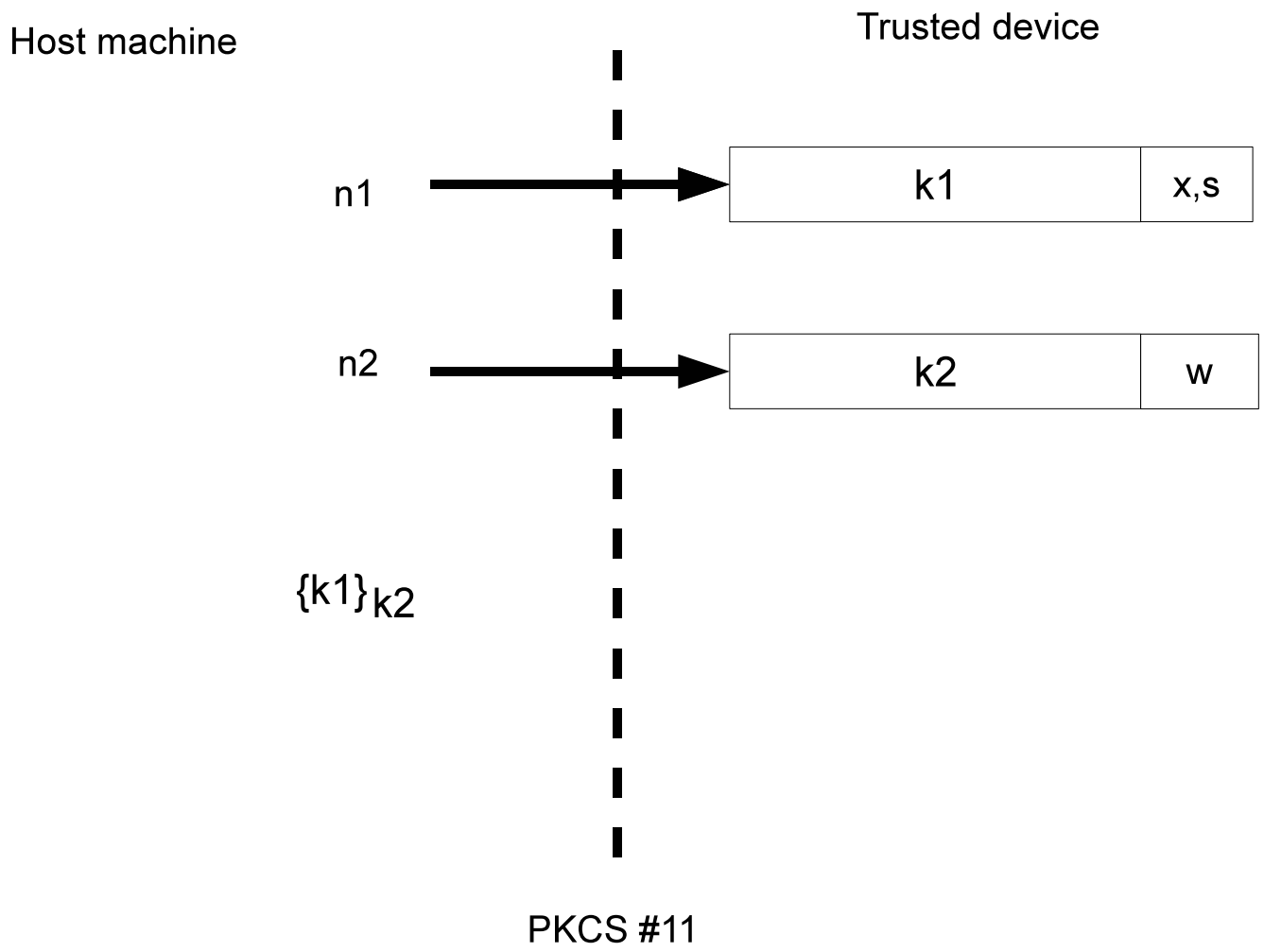
PKCS#11 Security

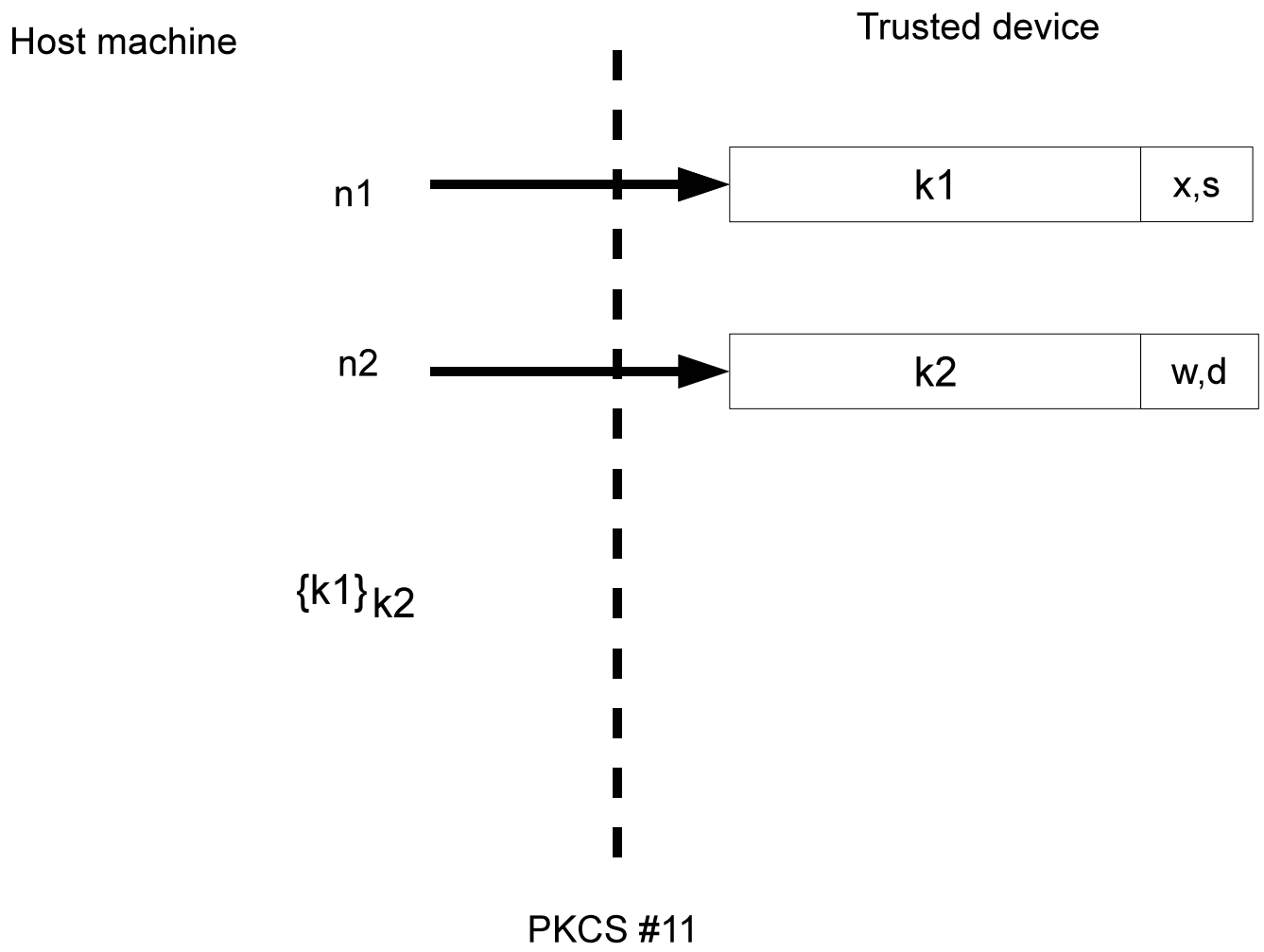
Section 7 of standard:

“1. Access to private objects on the token, and possibly to cryptographic functions and/or certificates on the token as well, requires a PIN.

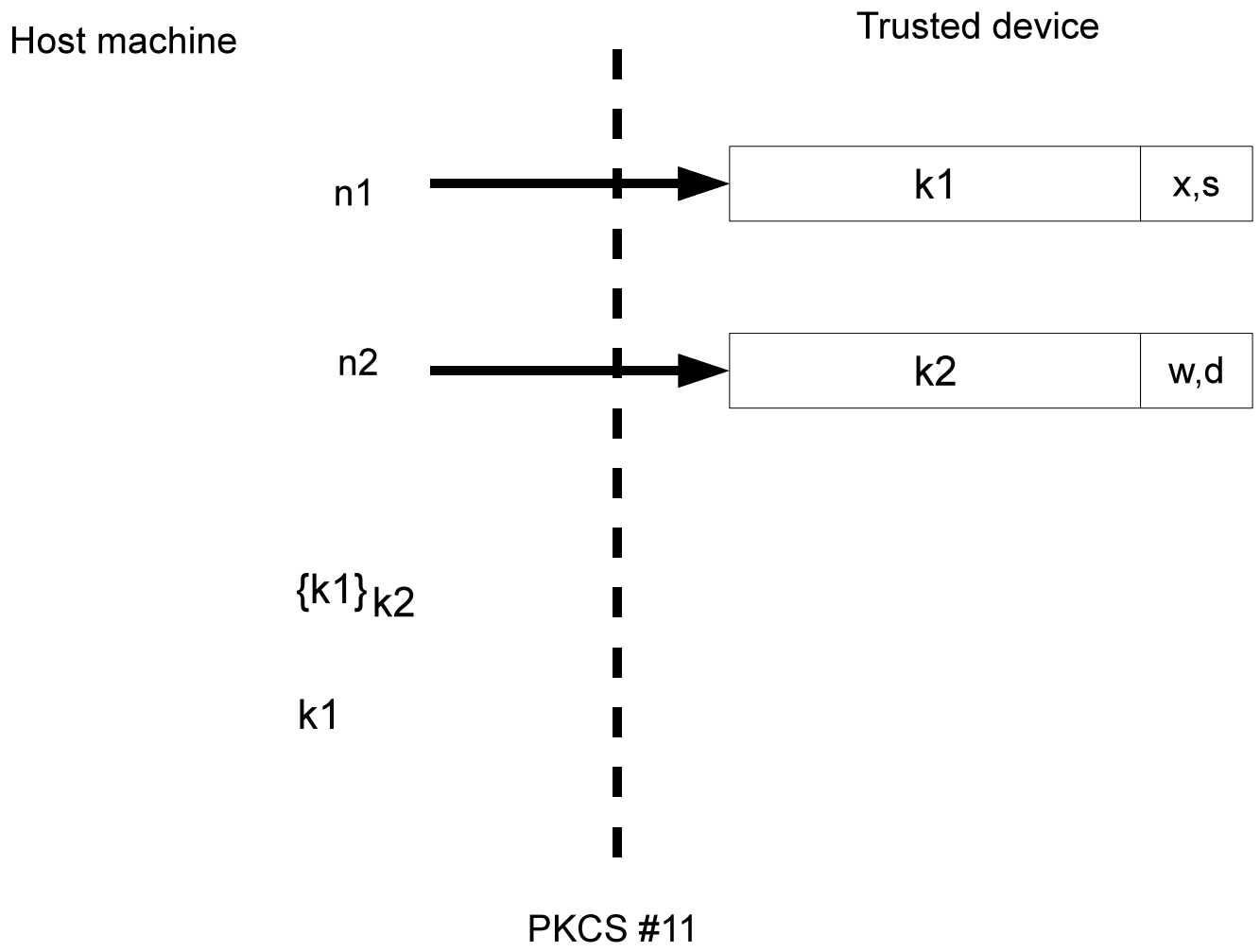
2. Additional protection can be given to private keys and secret keys by marking them as “sensitive” or “unextractable”. Sensitive keys cannot be revealed in plaintext off the token, and unextractable keys cannot be revealed off the token even when encrypted”

“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 a key from doing decrypt and wrap..

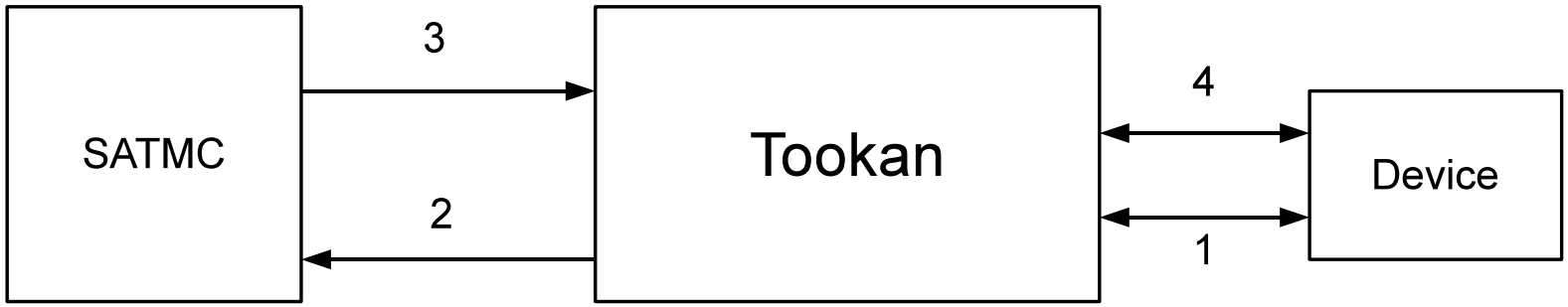
Intruder knows: $h(n_1, k_1)$, $h(n_2, k_2)$, k_3

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

TOOKAN

‘Tool for cryptoKi Analysis’



Configuration Language

Functions

Attributes

Always on/off

Conflicts

Tied

Templates

Flags

(see http://secgroup.ext.dsi.unive.it/tookan for full description)


TOOKAN

Device		Supported Functionality						Attacks found				
Brand	Model	s	as	cobj	chan	w	ws	wd	rs	ru	su	Tookan
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

Manufacturer Reaction

All 7 received notification at least 5 months before publication.

We offered to publish responses on project website

RSA sent response, registered vulnerability with Mitre (CVE-2010-3321), issued security advisory 6 Oct 2010

Aladdin (now Safenet) sent a 2-page response for website

Minimal response from anyone else (e.g. requests to know who else is vulnerable)

Since the first presentation of Tookan (CCS Chicago Oct ’10),
sold licences to Boeing and Barclays.

OpencryptokiX

IBM Opencryptoki is a library including a software token

Vulnerable to many attacks

We have coded two fixed versions

- one implements config from Fröschle & Steel WITS '09
- one is a new fix with no new crypto mechanisms

  Uses a carefully chosen set of templates $\mathcal{G} = \{wu, ed\}, \mathcal{U} = \{eu\}$

Available to download from

`http://secgroup.ext.dsi.unive.it/cryptokix`

Bees

- Library to assist programming PKCS#11 devices
- Offers a C++ and Java interface similar to model language
- Windows and Linux supported
- Used to construct the Tookan tool

Available to download from https://github.com/bugant/

Conclusions

Tookan: our tool for formal analysis of PKCS#11 configurations

OpencryptokiX: a sandbox for trying token configurations

Bees: a library for programming PKCS#11 tokens using symbolic model language

State of art of tokens not great (10/18 vulnerable, the rest very limited functionality)

Some manufacturers patching, no reaction from others

Recently: new attacks using error oracles

Project webpage:

http://secgroup.ext.dsi.unive.it/tookan