TKM

An IKEv2 Key Manager written in Ada 2012

Reto Buerki
Adrian-Ken Rueegsegger

Institute for Networked Solutions
21.09.2017

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

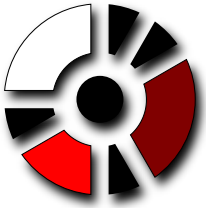
Overview

- Introduction
- Terminology
- Component-based Architecture
- Methods & Best Practices
- Conclusion

Who are we?

- MSc in Engineering from HSR
- 9+ years in IT security industry
- Researchers @ INS
- Focus on
  - Development of trustworthy systems
  - Platform security
  - Component-based systems

What are we currently doing?



Muen Separation Kernel

`https://muen.sk`

Trusted Computing Base (TCB)

A small amount of software and hardware that security depends on and that we distinguish from a much larger amount that can misbehave without affecting security. [4]

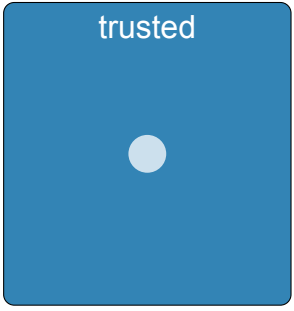
Trust and Trustworthiness

*Any component/system that has the potential to break your security policy is by definition **trusted**.*

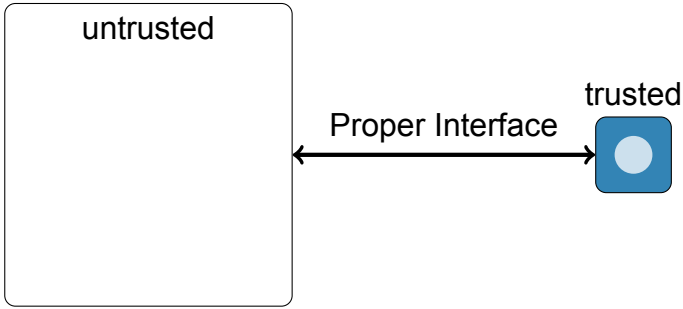
*In contrast, **trustworthy** refers to whether that trust is warranted. [1]*

Reducing Complexity of Trusted Code

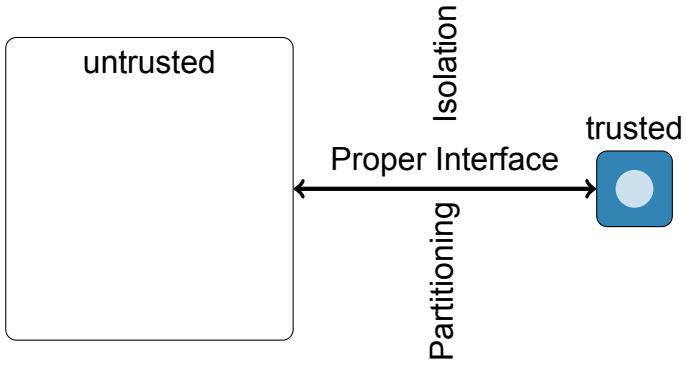
Reducing Complexity of Trusted Code



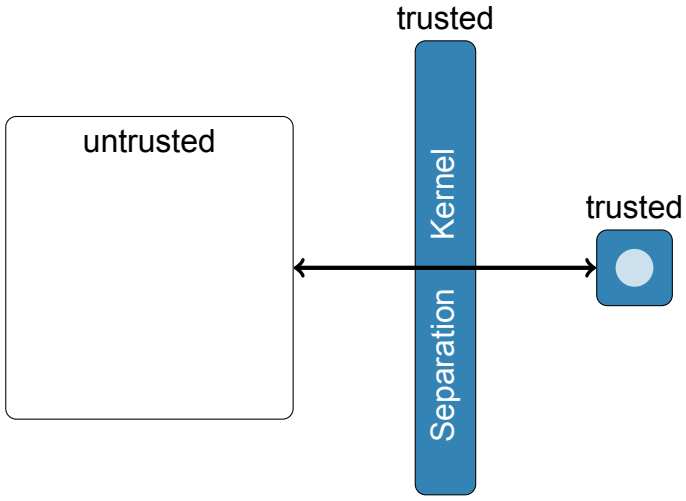
Reducing Complexity of Trusted Code



Reducing Complexity of Trusted Code



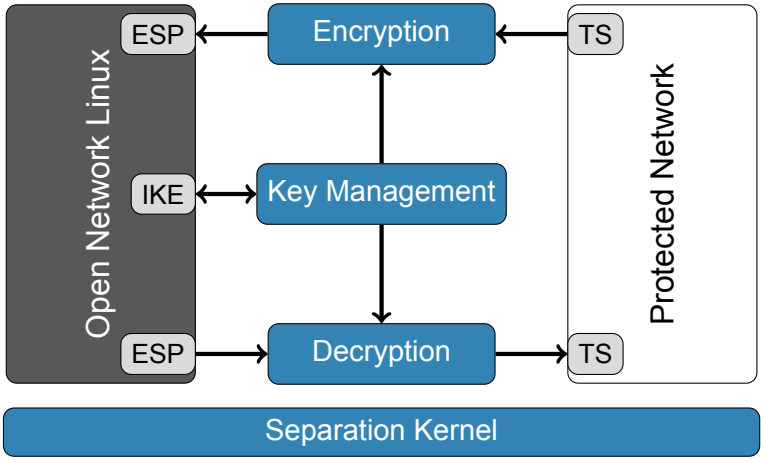
Reducing Complexity of Trusted Code



Component-based Architecture

- Split system in trusted/untrusted functionality
- Specify component interfaces & resources
- Reuse existing code for untrusted parts
- Maximize assurance of trusted components
- Shift complexity into untrusted components
- Employ solid separation (e.g. Muen SK)

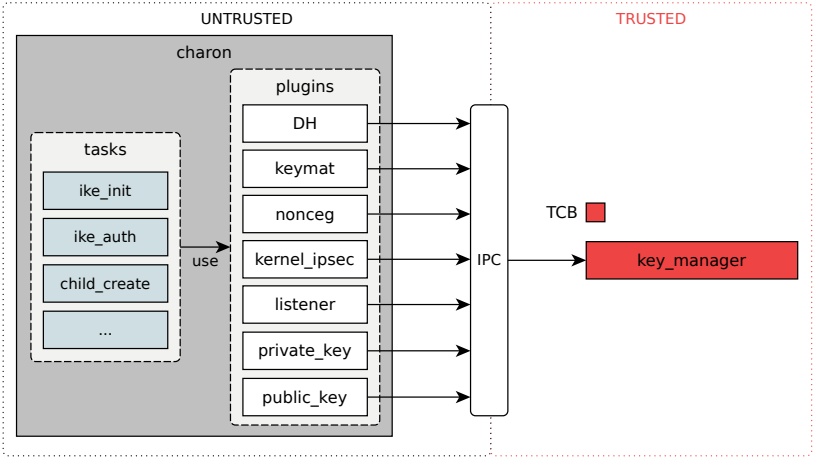
Architecting Secure Systems



Identify Security Properties

- Required to enforce security guarantees of system
- Functionality to extract are a consequence of desired security
- These properties must hold even in the presence of an attacker
- Examples:
  - Only trusted components have access to sensitive data
  - Only trusted components have access to cryptographic keying material

Split of strongSwan



Methods & Best Practices

- Low complexity, KISS
- Programming language
- Formal methods
- Proper interfaces
- State machines
- Combine with
  - TDD
  - CI
  - Design and code reviews
  - Pair programming

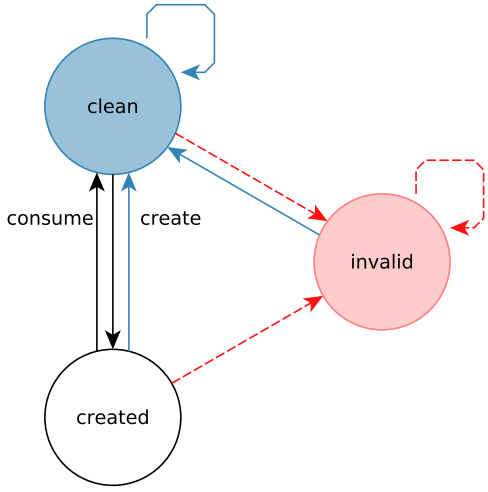
Low Complexity

- In trusted code, *every* line counts and should be avoided
- Shift complexity out of trusted components
- Design system in order to minimize overall TCB
- KISS
  - Greatly improves readability of code
  - Facilitates effective code review by third party
  - Simplifies/Enables formal verification

Interfaces

- Transitions *untrusted* $\rightarrow$ *trusted* are critical
- Proper data validation is key
- Use state machines to enforce valid transitions
  1. Create new Nonce
  2. Consume Nonce

Modeling Finite State Machines (FSM)



FSM: Specification

```
1  <transition name="create">
2    <descr>Create new nonce.</descr>
3    <source_states>
4      <state name="clean"/>
5    </source_states>
6    <target>
7      <state name="created"/>
8      <field name="nonce">nonce</field>
9    </target>
10 </transition>
```

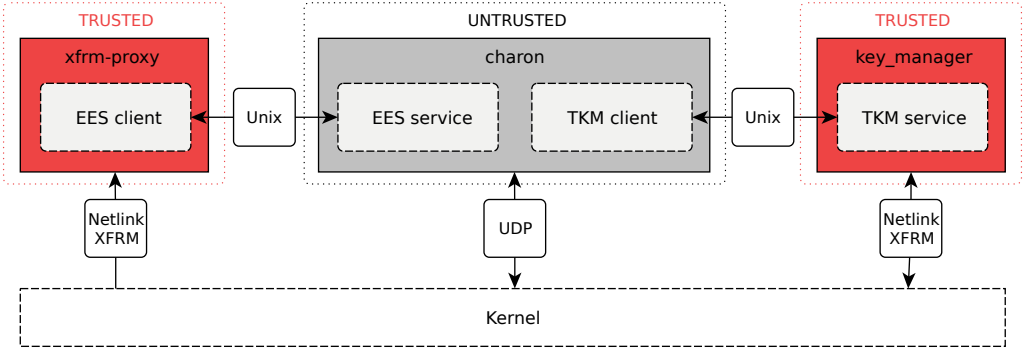
FSM: Code

```
1  --  Create new nonce.
2  procedure Create
3    (Id    : Types.nc_id_type;
4     nonce : Types.nonce_type)
5  with
6    Pre  => Is_Valid (Id) and then
7            (Has_State (Id, clean)),
8    Post => Has_State (Id, created) and
9            Has_nonce (Id, nonce);
```

FSM: Summary

- Complete specification of all transitions
- Assert correct state transitions via contracts
- Go to *Invalid* if violation on protocol $\rightarrow$ explicit recovery
- Automated generation of
  - Code
  - Graphs
  - Documentation

System Overview



Summary

- Secure software is limited in complexity
- Separation of untrusted components essential
- Crucial to be aware of dependencies (TCB)
- KISS - every feature comes at a cost
- Composition enables construction of trustworthy systems
- Formal verification can be done under commercial constraints
- Design systems so one bug does not mean *Game Over*

Discussion

Trusted Key Manager

`https://www.codelabs.ch/tkm/`

References I

David Burke, Joe Hurd, and Aaron Tomb.
High Assurance Software Development.
White paper, Galois Inc., August 2010.
http://code.galois.com/paper/2010/HighAssuranceSoftwareDevelopment.pdf.

Hanno Böck, Aaron Zauner, Sean Devlin, Juraj Somorovsky, and Philipp Jovanovic.
Nonce-disrespecting adversaries: Practical forgery attacks on gcm in tls.
Cryptology ePrint Archive, Report 2016/475, 2016.
http://eprint.iacr.org/.

References II

- Michael Hohmuth, Michael Peter, Hermann Härtig, and Jonathan S. Shapiro.
  Reducing TCB Size by Using Untrusted Components: Small Kernels Versus Virtual-machine Monitors.
  In *Proceedings of the 11th Workshop on ACM SIGOPS European Workshop*, EW 11, New York, NY, USA, 2004. ACM.

- Butler Lampson, Martín Abadi, Michael Burrows, and Edward Wobber.
  Authentication in distributed systems: theory and practice.
  SIGOPS Oper. Syst. Rev., 25(5):165–182, September 1991.

References III

Reto Buerki and Adrian-Ken Rueegsegger.
IKEv2 Separation: Extraction of security critical components into a Trusted Computing Base (TCB).
Technical report, HSR University of Applied Sciences Rapperswil, February 2013.
https://codelabs.ch/tkm/ike-separation.pdf.