November 27th, 2014.

TRUST IN SOFT

cybersecurity provider

Secure your Software

Fabrice Derepas
CEO

Slides for
CAP'TRONIC

TRUST IN SOFT

1. Landscape
2. Value proposal & offer
3. Stories from our customers
4. Examples

TRUST IN SOFT

What is this?

TRUST IN SOFT

It is a matter of

confidence

Main domains of interest

Telecom

Rail

Internet of things

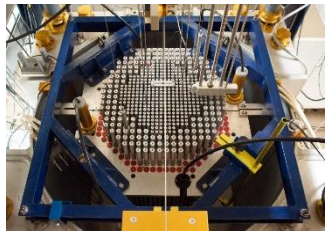
Space

Industry

Defense

Aeronautics

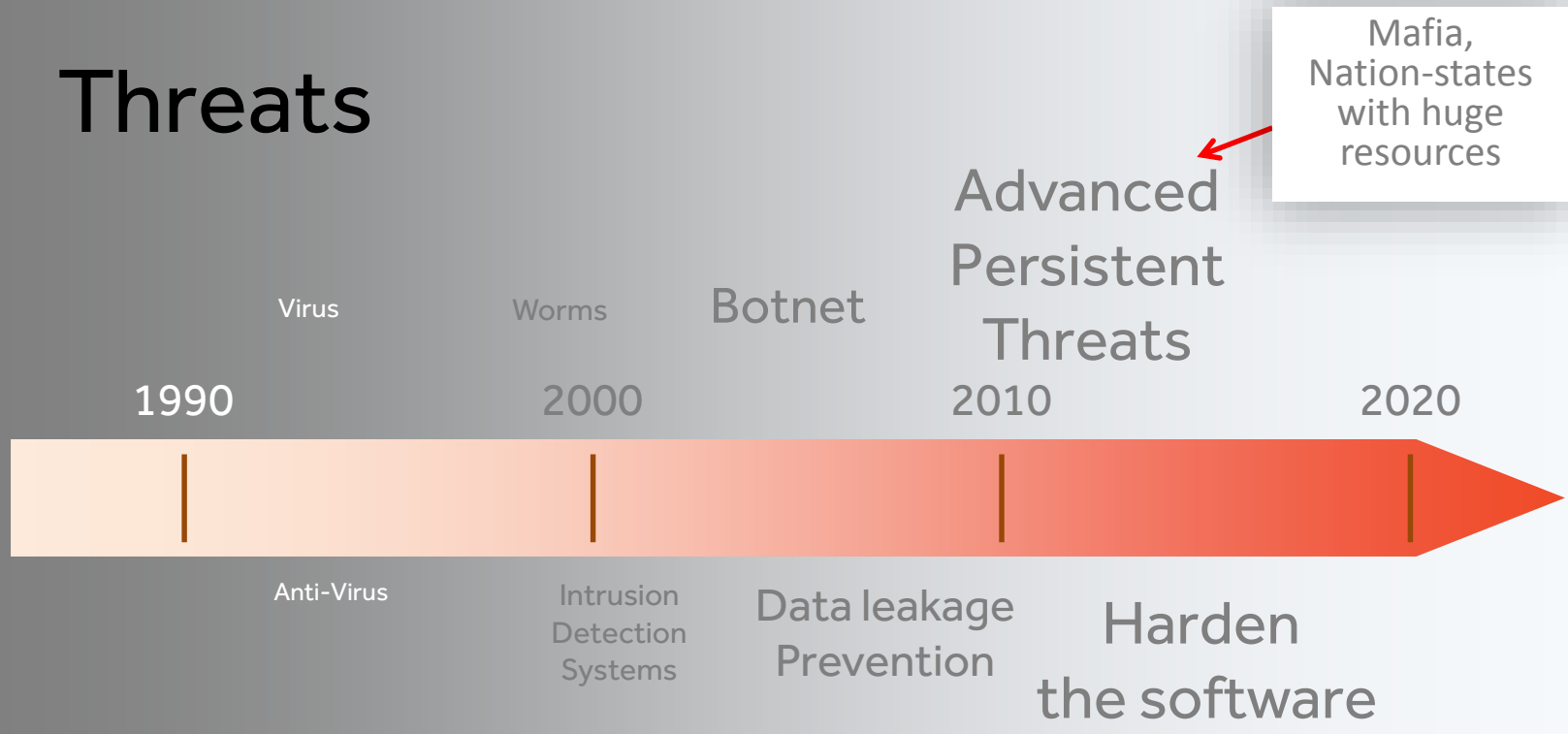
Energy



TRUST IN SOFT

Cyber Security

Threats



Solutions

TRUST IN SOFT

TRUST IN SOFT

1. Landscape
2. Value proposal & offer
3. Stories from our customers
4. Examples

TRUST IN SOFT

TrustInSoft unique value proposal

safety & security guarantees for software components

NIST National Institute of Standards and Technology U.S. Department of Commerce

cybersecurity

IRSN INSTITUT DE RADIOPROTECTION ET DE SÛRETÉ NUCLÉAIRE

nuclear industry

FEDERAL AVIATION ADMINISTRATION RTCA

aeronautics

Three offers

TRUST IN SOFT VERIFIED

Advanced Software Security Audits

We do the job for you

License of TRUST IN SOFT ANALYZER

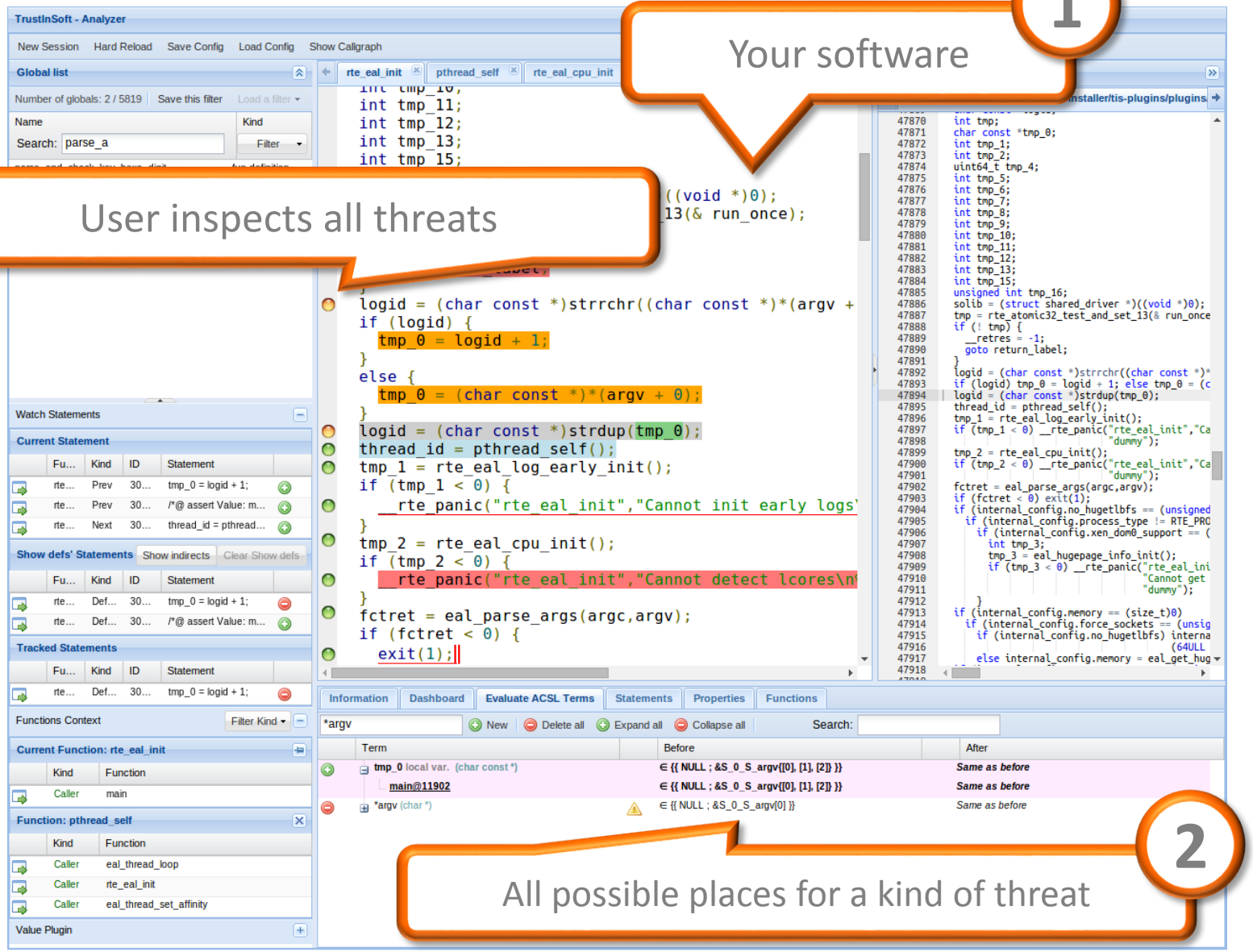
You use our technology

Expertise

Training, support,
On demande verification

TRUST IN SOFT

State-of-the-art source code analyzer



TRUST IN SOFT

An example

POLAR SSL
Straightforward, Secure Communication

TRUST IN SOFT VERIFIED

This is the world first widely used SSL stack guaranteed without buffer overflows

Polar SSL is a piece of software which enables secure transactions like HTTPS for instance

Just acquired by ARM

Using TRUST IN SOFT ANALYZER we now have a report which tells how to compile, configure and deploy Polar SSL in order to be immune against all attacks caused by CWE 119 to 127, 369, 415, 416, 457, 476, 562, 690.

TRUST IN SOFT

1. Landscape
2. Value proposal & offer
3. Stories from our customers
4. Examples

TRUST IN SOFT

You use SSL cryptography

- You can get the right configuration for your SSL stack so that your system can no be hacked by a classic attack.
- You can also sponsor other open source components which are critical for you. (IPSec stack ,…)

TRUST IN SOFT

Eradicate run time errors

- You subcontracted a piece of C software
- You want to make sure the subcontracted software has no runtime errors (Buffer overflow, invalid pointer usage, division by zero, non initialized memory read, dangling pointer, arithmetic overflow, NaN in a float computation, overflow in float to integer conversion).
- You or your subcontractor can use TrustInSoft Solutions to eradicate run-time errors.

TRUST IN SOFT

Production system which crashes

- A router uses a modified verions of the linux kernel with specific queueing policy.
- The systems happens to crash sometimes.
- We where able in 2 hours to locate the problem in the linux kernel module.

TRUST IN SOFT

Master your multithreaded software development

- You have designed a multi threaded software.
- You want to be confident in the fact that threre are no race conditions and no unexpected shared variables.
- Our solutions can help you automate the verification, or we can do it for you.

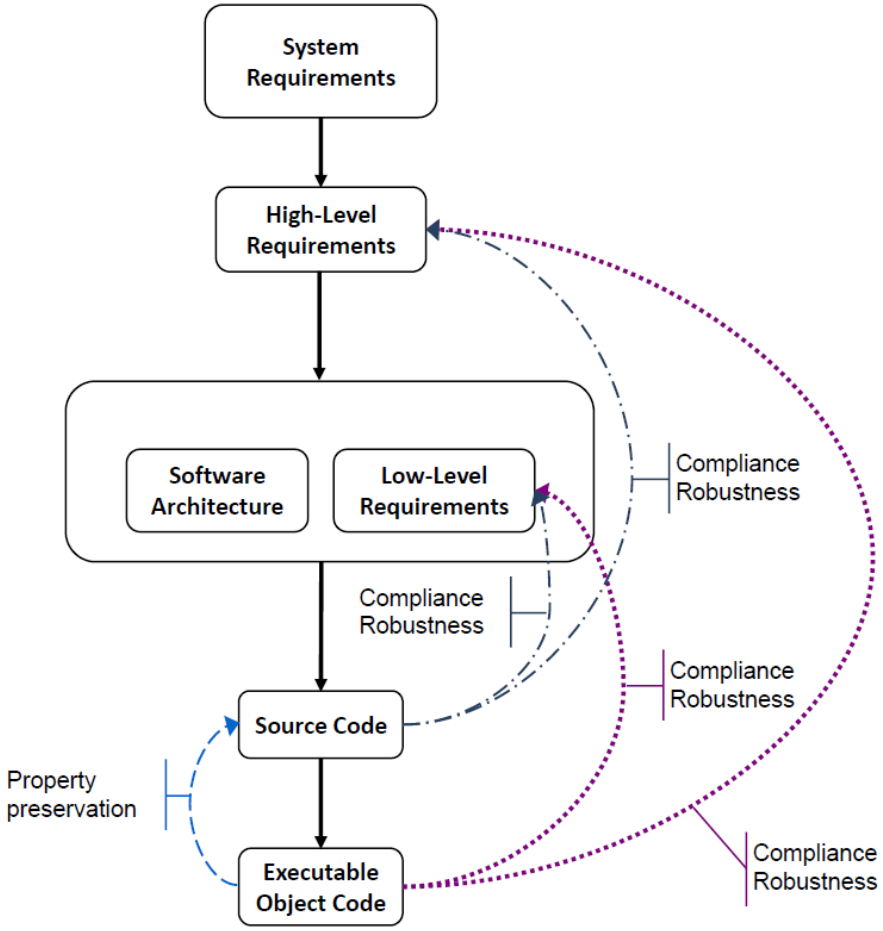
TRUST IN SOFT

Quality beyond marketing

- You know you design high quality software.
- You want to prove that you are better than your competitors.
- TrustInSoft helps you highlight the quality of your products.

TRUST IN SOFT

Follow requirements



[1] Use of formal methods in embedded software development: stakes, constraints and proposal Anthony Fernandes Pires, Thomas Polacseky, Virginie Wielsy and Stéphane Duprat

TRUST IN SOFT

High level benefits

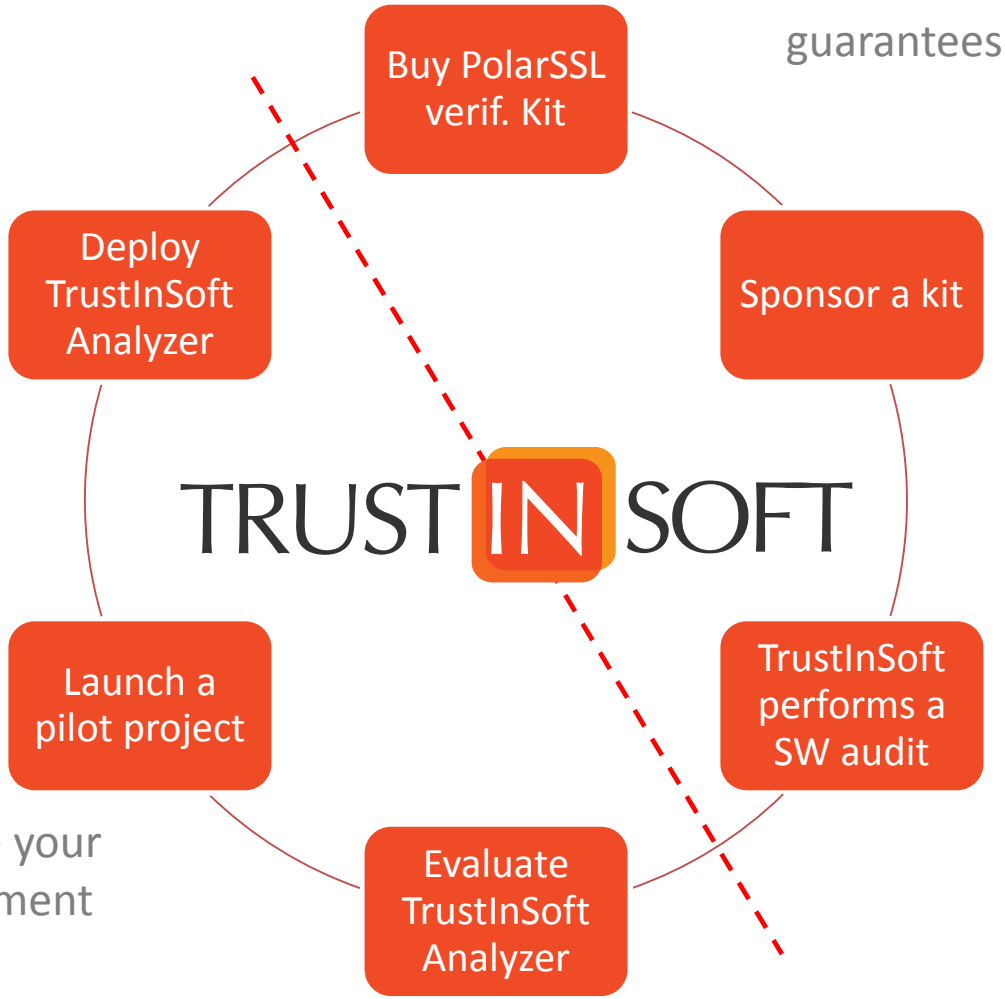
Prove that your products are safe & secure

➔to your customers
➔to national authorities

Anticipate cybersecurity regulation

Let's collaborate now!

On the shelf guarantees



Guarantees inside your software development process

TRUST IN SOFT

1. Landscape
2. Value proposal & offer
3. Stories from our customers
4. Examples

Valgrind Comparison

Valgrind comparison

- Valgrind is a popular Open Source tool used to find memory faults in program
- We present in the following slides three comparisons with Valgrind
- Some companies use Rational Purify, a widespread tool similar to Valgrind

TrusInSoft vs. Valgrind 1-0

The following code has an error: address of variable b is used outside of its scope

```
int main () {
  int * my_int= (int*)0;
  {
    int b=314;
    my_int = &b;
  }
  *my_int=22;
  return 0;
}
```

Valgrind :
No error mentioned

TrustInSoft :
locals {b} escaping the scope of a block of main through my_int

TrustInSoft vs. Valgrind 2-0

The following code has an error:

array b is accessed from a

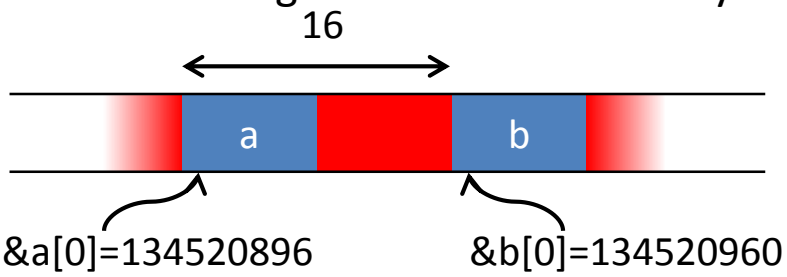
```
int a[10] = {1,1,1,1};
int b[10] = {2,2,2,2};

int main () {
  a[16]=3;
  return b[0];
} // returns 3
```

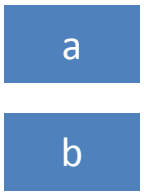
Valgrind:
No error mentioned

TrustInSoft:
Writing out of bounds
at index 16

Valgrind concrete memory model



TrustInSoft abstract memory model



TRUST IN SOFT

TrustInSoft vs. Valgrind 3-0

Source code of a virtual machine which computes 2^4

value of B not checked

```
#define ARRAY_SIZE 11
unsigned char mem[ARRAY_SIZE]= \
  {80,7,5,5,3,5,3,5,4,11,2};
#define NEXT \
  if (pos<ARRAY_SIZE-1) ++pos;
break;

int main () {
  unsigned int A=0,B=0,pos=0;
  pos=0;
  while (1) {
    switch (mem[pos] & 7) {
    // add
    case 0: A+=mem[pos]>>3; NEXT;
    // substract
    case 1: A-=mem[pos]>>3; NEXT;
    // load
    case 2: A=mem[B]; NEXT;
    //store
    case 3: mem[B]=A; NEXT;
    // exit
    case 4: return A;
    // load and add
    case 5: if (B<ARRAY_SIZE)
        A=A+mem[B]; NEXT;
    // goto A
    case 6: if (A<ARRAY_SIZE)
        pos=A; break;
    // swap A and B
    case 7: {int tmp=B;B=A;A=tmp;}
        NEXT;
    }
  if (++pos==ARRAY_SIZE) pos=0;
} }
```

TRUST IN SOFT

All virtual machines with memory size of 11

```
#define ARRAY_SIZE 11
unsigned char mem[ARRAY_SIZE] = {80,7,5,5,3,5,3,5,4,11,2};
#define NEXT if (pos<ARRAY_SIZE-1) ++pos;\
  break;

int main () {
  unsigned int A=0,B=0,pos=0;
  while (1) {
    // . . .
```

Here is the program
for a given state of the virtual machine
This program runs without error

```
#define ARRAY_SIZE 11
unsigned char mem[ARRAY_SIZE];
#define NEXT \
  if (pos<ARRAY_SIZE-1) ++pos; break;

int main () {
  unsigned int A=0,B=0,pos=0;
  for (pos=0;pos<ARRAY_SIZE;++pos) mem[pos]=Frama_C_interval(0, 255);
  pos=0;
  while (1) {
    // . . .
```

TrustInSoft Analyzer
tests all possible virtual machine of size 11.
256^{11} tests.
In a single run.

Symbolic value: all integers between 0 and 255

Comparison with other static analyzers

TRUST IN SOFT

Competitors

- Other static analyzer (Coverity, Fortify) are not able to bring guarantees on the source code
- For this reason TrustInSoft Analyzer was the only technology able to pass the NIST SATE V Ockham Criteria

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

TRUST IN SOFT

Eradicated weaknesses

Standard vulnerabilities:

- Buffer overflow, invalid pointer usage, division by zero, non initialized memory read, dangling pointer, arithmetic overflow, NaN in a float computation, overflow in float to integer conversion,

Other vulnerabilities:

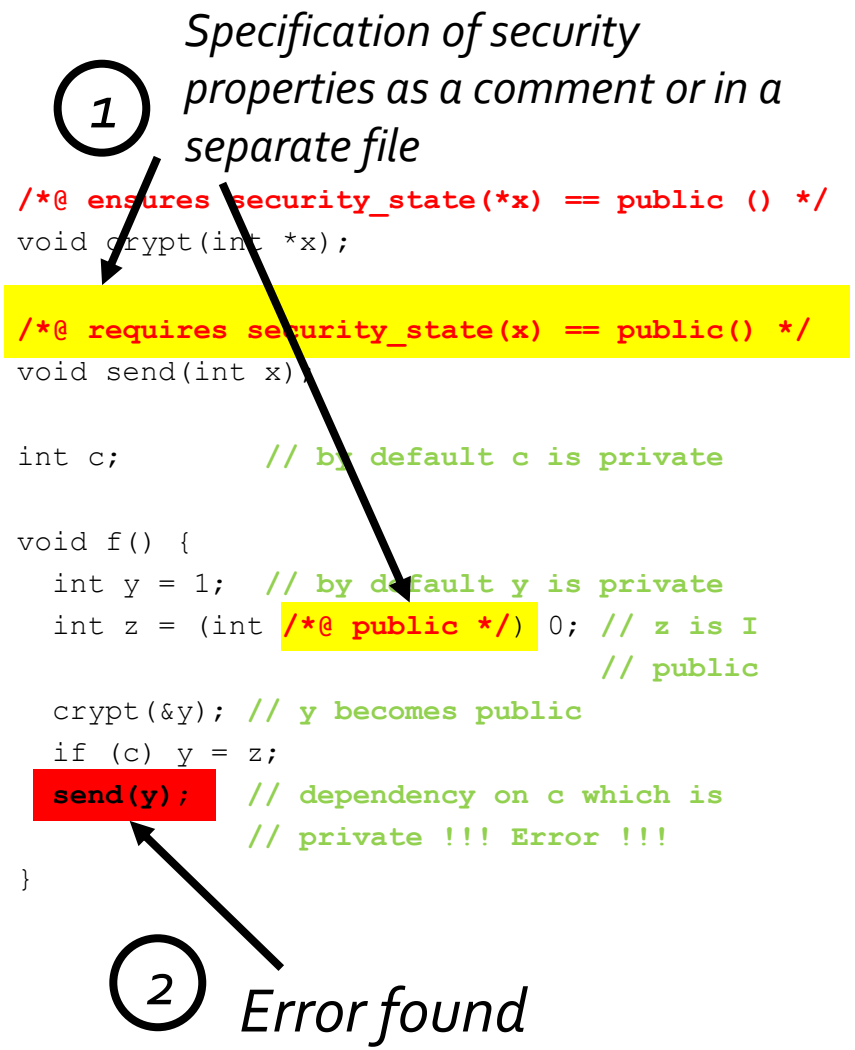
- CWE-078: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
- CWE-306: Missing Authentication for Critical Function
- CWE-798: Use of Hard-coded Credentials CWE-311: Missing Encryption of Sensitive Data CWE-807: Reliance on Untrusted Inputs in a Security Decision
- CWE-250: Execution with Unnecessary Privileges CWE-022: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
- CWE-863: Incorrect Authorization
- CWE-676: Use of Potentially Dangerous Function
- CWE-732: Incorrect Permission Assignment for Critical Resource
- CWE-327: Use of a Broken or Risky Cryptographic Algorithm
- CWE-307: Improper Restriction of Excessive Authentication Attempts
- CWE-134: Uncontrolled Format String
- CWE-759: Use of a One-Way Hash without a Salt CWE-770: Allocation of Resources Without Limits or Throttling
- CWE-754: Improper Check for Unusual or Exceptional Conditions
- CWE-838: Inappropriate Encoding for Output Context
- CWE-362: Concurrent Execution using Shared Resource with Improper Synchronization ('Race Condition')
- CWE-841: Improper Enforcement of Behavioral Workflow
- CWE-772: Missing Release of Resource after Effective Lifetime
- CWE-209: Information Exposure Through an Error Message
- …

TRUST IN SOFT

Advanced security properties

Validation of security properties

- TrustInSoft can specify properties on software.
  An example is shown on a toy example for cryptography where only public data shall be sent on a network
- This methodology can be adapted to match your security requirements



TRUST IN SOFT

Contact info

Fabrice Derepas

06 51 70 36 77

derepas@trust-in-soft.com