ISAE Embedded systems master

Evaluation – Exercices and questions with corrections

28 january 2016

Computer security

Advice to students and supervisors : course documents (either furnished by the school or hand-written during oral courses by the student himself) are allowed during the examination, a standard calculator too (for calculation purposes only) and blank paper sheets for draft.

All other documents or media access are not allowed, unless direct explicit authorization from the session supervisor.

Advices :

- Do not hurry, you should have all the time needed to prepare your answers. Use a draft and be on topic.
- Be careful with multiple choice questions:the notation penalizes bad answers and some are not so straightforward.
- Do not simply research the answers in the course documents.

Please, write your answers on the document itself in the place reserved.

Student name and surname :

Part I (10 pts)

This first part consists of ten questions (1 pt per question) with multiple answers proposed among which you must select the appropriate one. Unless explicitly indicated, only one answer is the right one.

Attention, the following notation system will be used :

Right answer : 1 point added

False answer : 0,25 point *removed*

No answer : 0 point

Q1 What is the distinct advantage of positioning the computer security officer of a company inside its computing department ?

- ☐ He will explain the existing vulnerabilities to top level management under the wise supervision of the IT head .

- ☐ He will have his hands busy with actual software security updates deployment and be confronted to real issues.
- ☐ He will be easily available to provide technical advices to the various software projets managed by the IT division.
- ☐ He will have administrator-level credentials and be able to access all the files in the company under the control of the IT division.

Q2 In a buffer overflow exploitation code, why is it important to exit cleanly after taking control of the CPU execution path :

- ☐ To prevent detection of the attack.
- ☐ Because a multiple steps attack will not work if some of the intermediate steps lead to faults catched by the OS.
- ☐ Because the hackers coding standard requires it.
- ☐ Because we may freeze the whole computer if we do not.

Q3 What is the security mechanism needed to reach the upper half of the evaluation levels in normalized evaluation criteria :

- ☐ A mandatory security policy.
- ☐ A trusted execution path (SysRq).
- ☐ A discretionay security policy.
- ☐ Lots of documentation.

Q4 A `"%s"` format should always be passed to `printf()` calls because :

- ☐ it will display a better formatted user-level message ;
- ☐ it makes the job of quality control people easier ;
- ☐ it will prevent the program from crashing ;
- ☐ it may prevent the program from revealing internal data and memory layout.

Q5 At which step of the application development phase is it best to identify the needed security *mechanisms* :

- ☐ At the beginning of the development phase, when global requirements are declined into detailed specifications.
- ☐ During negociations with sub-contractors implementing them.

- ☐ At the integration phase when the development of the main software body is completed.
- ☐ At the end of the system life, so users do not get too annoyed by security constraints.

Q6 Because *floating-point* numbers represent real numbers, it is often mistakenly assumed that they can represent any simple fraction exactly. Floating-point numbers are subject to representational limitations just as integers are, and binary floating-point numbers cannot represent all real numbers exactly, even if they can be represented in a small number of decimal digits. Noting that the decimal number 0.1 is a repeating fraction in binary and cannot be exactly represented as a binary floating-point number, consider the following code fragment.

```
void func(void) {
  for (float x = 0.1f; x <= 1.0f; x += 0.1f) {
    /*Some loop body */
  }
}
```

How many iterations will the above program fragment perform at execution ?

- ☐ an unpredictable number
- ☐ 10 iterations
- ☐ either 9 or 10 times, depending on the implementation
- ☐ 9 iterations

Q7 What is the information provided daily by a CERT (Computer Emergency Response Team) ?

- ☐ Information on computer software vulnerabilites.
- ☐ Information on the most agressive computer hacking teams.
- ☐ Emergency information in case of a general Internet failure due to attacks.
- ☐ Cyber-security awareness raising documents for the general public.

Q8 Given the vulnerabilities identified by cryptanalysts on the MD5 hash function, what would be the adequate advice to give to the developpers of the git source code management system which uses MD5 sums as identifiers of source files successive versions ?

- ☐ Replace MD5 by SHA3 for all future versions as a more secure identifier
- ☐ Replace MD5 by SHA3 and also implement mechanisms allowing to update past data and migrate it also to the more secure version
- ☐ Replace MD5 by full RSA signatures

- ☐ Stay as-is. MD5 is good for content based adressing and fast checking of different text files when so specific security property is needed.

Q9 Intelligence agencies analysts frequently gather information coming from several sources in order to obtain secret information. For example, they may find the destination of some navy ship starting with the fuel bay capacity, ship speed and bought food supplies. When doing so, analysts use :

- ☐ interference
- ☐ inference
- ☐ covert channels
- ☐ psychology

Q10 Before the test date, questions, exercices and correction guidelines are only accessible by professors and supervisors. This rule is related to :

- ☐ confidentiality
- ☐ integrity
- ☐ availability
- ☐ or *survivability*

Q11 On a computer system implementing the Bell La-Padula multilevel mandatory security policy, is it possible that there exists access rights and a owner associated to the files of the filesystem ?

- ☐ YES
- ☐ NO

Bonus question What is the *worst* option from the security point of view :

- ☐ An application where all users have different identifiers but everyone uses a blank password in order to allow automatic access from another portal application.
- ☐ An application where all users share the same identifier and the same password (changed every year).
- ☐ An application with a hidden password that allows access for maintenance.
- ☐ An application without any authentication at all.

Hint : All options are obviously suboptimal with respect to potential security requirements. However, the third case involves explicitly hiding to end users a critical vulnerability of the software they. Unfortunately, many systems, including security-oriented ones (home alarm

systems, password management software for example), have been shown to exhibit such a characteristic.

Part II (10 pts)

This part is composed of five open questions (2 pts each). Please write down your answer *on this document* in the appropriate space.

Question 1

Give an original exemple of a computer security objective (i.e. a requirement) corresponding to an organization and the associated security rule (i.e. a mechanism) that can be proposed in order to help reach the desired objective.

Give another pair of examples corresponding to a piece of software.

NB : Do not simply reuse as is the examples proposed as illustration in the course. If needed for comprehension, provide a few hints of the kind of organization or software you envisaged (bank, hospital, army, phone chat software, RDBMS, word processor, etc.).

Two answer examples proposed (only one was requested)

Context hint for the organization (*optional*) *Bank*

Objective 1 (organization) :

A) Dual validation should be done on all significant stock market operation

B) Regular low-level employees should not be associated to cash flow management.

Rule 1 (organization) :

A) After being entered in the system by employees according to customer needs, each operation is sent by the information system to a validation phase (by a manager who checks the operation).

B) ATMs (Automatic Teller Machines) are resupplied by external fund transport personnel and the bank agencies employees are not allowed to access these funds.

Context hint for the software (*optional*) *Encryption mail software*

Objective A (software) :

A) No administrator should be centrally responsible for the key signing infrastructure.

B) All private key storage areas should be controlled and explictily managed.

Rule A (software) :

A) Each user is allowed to certify another user identity by signing keys. The software should easily allow to check the existence of a validation chain between the main identity and those of all mail recipients.

B) Private keys should only be stored inside smalloc()-ed areas. All such areas should be superfree()-d after use. An automatic check of source code should be done to enforce this rule.

Question 2

Give 4 examples of malicious faults, accidental faults or intentional (but non-malicious) faults (at least 1 one of each class).

(Additional examples proposed)

Malicious faults :

- *Burglary*
- *Car hijacking*
- *Backdoor password*

Accidental faults :

- *Storm damage*
- *Fire*
- *Stepping on someone's toes*
- *Forgetting one's handbag (nb : can be intentional, but usually not in a professional context)*

Intentional but non-malicious faults :

- *Program bug*
- *Maintenance password*
- *Hiding the program source code to prevent vulnerabilities identification (and correction)*
- *Install your-favorite-game on your professional smart phone*

Question 3

What are the advantages *and* drawbacks of using all the currently most commonly available algorithms of cryptography (i.e. : RSA for asymetric encryption, AES for symetric encryption and SHA3 as a secure hash function) and only them ?

Pros :

- *Reliable, proven existing algorithms and implementations*
- *State of the art cryptography*
- *Standardized (all 3)*
- *Hardware implementations available*

Cons :

- *Single line of defense, no diversity*
- *Not always suited to embedded/specific systems*

Question 4

The CERT C Coding Standard documentation provides the following information and *non*-compliant code example with respect to the usage of the `system()` function, as well as an example of secure usage inside a POSIX environment.

« […] *The C Standard `system()` function executes a specified command by invoking an implementation-defined command processor, such as a UNIX shell or `CMD.EXE` in Microsoft Windows.* [...removed for brevity...].

Use of the `system()` function can result in exploitable vulnerabilities, in the worst case allowing execution of arbitrary system commands. [...removed for brevity...]

Noncompliant Code Example

In this noncompliant code example, the `system()` function is used to execute `any_cmd` in the host environment. Invocation of a command processor is not required.

```
#include <string.h>
#include <stdlib.h>
enum { BUFFERSIZE = 512 };
void func(const char *input) {
  char cmdbuf[BUFFERSIZE];
  int len_wanted = snprintf(cmdbuf, BUFFERSIZE, "any_cmd '%s'",
input);
  if (len_wanted >= BUFFERSIZE) {
    /* Handle error */
  } else if (len_wanted < 0) {
    /* Handle error */
  } else if (system(cmdbuf) == -1) {
    /* Handle error */
  }
}
```

[…removed for exam. purpose...]

Compliant Solution (POSIX)

In [the] compliant solution, the call to `system()` is replaced with a call to `execve()`. The `exec` family of functions do not use a full shell interpreter, so they are not vulnerable to command-injection attacks, such as the one illustrated in the noncompliant code example. [...]»

Explain how the above non-compliant code could be used to run a privileged command (like creating a new user account with somthing like « `useradd caroline` ») if it is compiled

and run with elevated privileges on a POSIX system in a context where a potential attacker can pass it an arbitrary string.

If possible, provide (possible) examples of the kind of input data an attacker could try to use to perform such an attack.

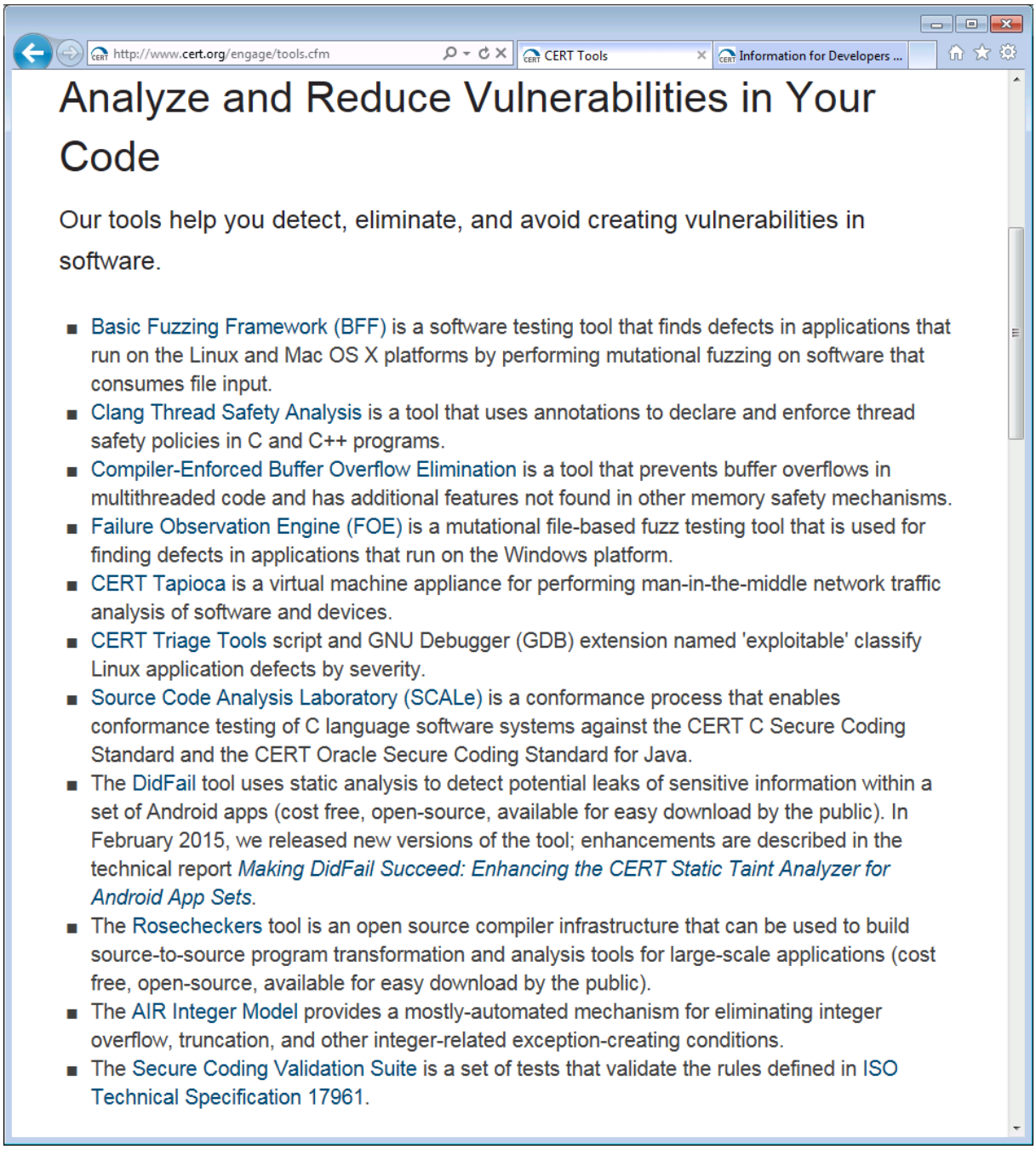
Question 4 answer :

NB : Check CERT C Secure Coding Standard rule ENV33-C « Do not call system() », available at : https://www.securecoding.cert.org/confluence/pages/viewpage.action?pageId=2130132

The following input could be malicious : `blah' ; useradd 'caroline` *or* `blah' ; useradd caroline ; echo 'empty` *to use strictly the given command string.*

The problem is you cannot easily and securely use the shell unless you yourself emulate the shell parsing (which cannot be called simple and also defeats the interest of calling it in the first place) or only allow basic arguments (in which case you can directly call execve() or other functions which bypass the shell entirely). Hence... the CERT recommendation.

Question 5



The above figure presents a page from CMU CERT website (currently available at http://www.cert.org/engage/tools.cfm) which lists several tools they recommend for software developpers. Among those, are there some that you would you be interested in using for software developed in the C language for an automotive industry device ? Justify and explain your selection criteria.

No need to enumerate all listed tools in details, but justify your choices.

Question 5 answer :

Your mileage may vary but all explained answers will be honoured to their respective merits.

Personnally, I know I would reluctantly be obliged to use some the black box testing tools because the subcontractors never want to give us the source code and the damn buying department always « forgets » to add that clause to the final signed version of the contract.

In the rare case where I have the source code, I would certainly rush to some other tool allowing to analyse it.

Probable low on the list here would be the programs focussing on environments too far from those found in the automotive industry : most probably Tapioca (network-oriented attacks) or Clang Thread Safety Analysis (multithreading may not be available at all).

Similarly, depending on the use of Android or not in our specific case, DidFail may be at the top or the bottom of the list.

Note again how source code checkers are more useful in long term. (OK, I stop.)