TOWARDS FORENSIC-READINESS BY DESIGN

Antonio Maña
Proteus Research, Universidad de Málaga
Safe Society Labs

City University, December, 2015
Introduction
Part 1: Introduction
- Digital Forensics. Situation and challenges.
- "by design". The big picture
- Security in Systems Engineering; Overview, current approaches to SSE, and problems

Part 2: forensic-readiness by-design
- What is forensic-readiness by-design
- An example of an infrastructure for forensic-readiness by design
Digital Forensics

- **Forensic:**
  - Origin: Latin *forens(is)* of, belonging to the forum, public
  - Meaning: the art or study of argumentation and formal debate.

- **Forensic Science:**
  - The application of science to those criminal and civil laws that are enforced by police agencies in a criminal justice system (Saferstein, 2004)

- **Digital Forensics (aka digital forensic science):**
  - a branch of forensic science encompassing the recovery and investigation of material found in digital devices, often in relation to computer crime. (Wikipedia)
  - The use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations. (Digital Forensic Research Workshop – DFRWS)
Digital Forensics

- Digital Evidence:
  - Any relevant information being subject to human intervention or not, that can be extracted from a digital device.
  - Must be possible to transform it in human-readable format or capable of being interpreted by a person with expertise in the subject.

- Examples
  - Recovering thousands of deleted emails
  - Performing investigation of past-employees hostile activities
  - Recovering evidence of a re-formatted hard drive
  - Analysing digital images for manipulation
  - Analysing digital video for determining its origin
  - Performing investigation after multiple users had taken over the system
  - ...
Do we really need forensics?

☐ In a perfect world...
  - everyone is good
  - applications and security mechanisms are correctly designed to fulfil the user needs and flawlessly implemented.
  - Forensics in such case makes little or no sense.

☐ But in the real world...
  - bad guys do exist
  - the design of applications and security mechanisms is based on wrong assumptions, oversimplified facts, vague and incomplete requirements, etc. which are then poorly implemented...
  - In this setting, forensics does not only make sense but becomes essential in order to achieve high levels of assurance.
The role of forensics is essentially to **learn what went wrong** with the security elements included in a system (in the wider sense: HW, OS, platform/middleware, applications...)

- Revising **assumptions**
  - Upon which the security of the system is built (e.g. whether we can assume that the different VMs running in a platform are isolated)

- Finding the **weaknesses** to improve the system security

- But can also help ensuring that bad behaviour is identified and the responsible entities are made **accountable** for it → dissuasive measure
Why Digital Forensics are needed

- Wide range of computer crimes and misuses
  - Regular crimes
    - Homicide
    - Forgery
    - Perjury
    - Theft of trade secrets
    - Fraud
    - Extortion
    - ...
  - Digital crimes
    - Industrial espionage
    - Pornography
    - SPAM investigations
    - Virus/Trojan distribution
    - Intellectual property breaches
    - Unauthorized use of personal information
    - ...
    - Cyberwar
Computer Forensics is closely related and often part of the **Incident Response** (IR) capability.

IR is an integral part of **Information Assurance** (IA).

IR adopts forensic “friendly” procedures & processes.

- More later

Proper evidence management and handling.
Incident Response and Digital Forensics

- Two of the least practiced, most stressful, highly scrutinized areas of Information Security.
- Every incident is unique and can incorporate many different areas of the affected organization, involve many technologies and involve huge amounts of data.
- Incident analysts must be able to think quickly, remain calm and consider all possibilities.
- Knowledge, tools and processes are only valid as part of an overall case-specific strategy.
6 Steps of the “Traditional” Incident Handler Methodology

- Preparation
- Identification
- Containment
- Eradication
- Recovery
- Lessons Learned
The basic elements of the forensic methodology are known as the 3 As:

- **Acquire** the evidence without altering or damaging the original
  - Normally, this is done by obtaining a perfect copy of the evidence that we’ll call “image”

- **Authenticate** the image
  - To ensure the validity of the analyses made on it

- **Analyse** the evidences without modifying them
  - Present the results in a way that is both sound and convincing
More in detail, digital forensics activities commonly include:

- the **secure collection** of devices and digital data (evidences)
- the **identification** of evidences
- the **authentication/preservation** of evidences
- the **examination** of evidences to determine details such as origin and content
- the **presentation** of computer-based information to courts of law
- the **application** of a country's laws to computer practice.
Legal and ethical principles
Investigations
Forensic science
Digital forensics
Application forensics
Hybrid and emerging technologies
Digital forensics is not yet a mature science

Different from other forensic sciences as the media that is examined and the tools/techniques for the examiner are products of a fast-evolving market-driven private sector

No real basic theoretical background upon which to conduct empirical hypothesis testing

Professional designations still not well established

Proper training not available

Still more of a “folk art” than a true science
Legal Challenges

- Status as scientific evidence
- Criteria for admissibility of novel scientific evidence (Daubert v. Merrell)
  - Whether the theory or technique has been reliably tested;
  - Whether the theory or technique has been subject to peer review and publication;
  - What is the known or potential rate of error of the method used; and
  - Whether the theory or method has been generally accepted by the scientific community.
- Kumho Tire case was relevant as it extended the criteria to technical knowledge (as opposed to “scientific”)
How to link technical/scientific results with real world

- In other words, how to prove technically and scientifically facts in the physical world?

How to present a highly-complex, extremely elaborated, computer-supported evidence so that it is convincing in a court?

- Results and processes are most of the times too complex to present them “raw”

How to be more convincing than the other party when the judge/jury does not have enough IT knowledge?

- Pseudo-forensics are a real problem
Specific Challenges

- What constitutes evidence?
- What are we looking for?
- Where is the “crime scene”? 
- What is the applicable law?
- ...

Towards Forensic-readiness by design
Specific Challenges

- No International definitions of Computer Crime
- No International agreements on extraditions
- Multitude of OS platforms and file systems
- Incredibly large storage capacity
  - 100 Gig Plus quite common
  - Terabytes not rare
- Virtualization and Cloud systems becoming popular
- Mobile devices frequently involved
- And all in constant evolution...
Many of the tools used currently for forensic analysis are:

- Difficult to find
- Difficult to set up
  - Poorly documented
  - Not thoroughly tested
  - Have lots of dependencies
  - Not updated, supported, not reliable
- Incompatible / not standardized
- Provide insufficient levels of security, rigor, trust
- Based on proprietary / secret algorithms
- ... do I need to continue?
Anti-Forensics

- Software and techniques to eliminate, limit and/or corrupt evidence that could be collected by an investigator
- Performs data hiding, encryption and destruction
  - Disk wiping, onion routers,...
- Exploits limitations of known and widely used ICT systems and forensic tools
- Works on all OS, even Internet
- Can be in place prior to or post incident and pre/post system acquisition
- Can simply focus on nullifying the legal validity of evidences
Why we need a new approach

- Attacks and cases that require forensic investigation are on the rise, and this is only the beginning
- Cases are increasingly “international” and require exchange and collaboration of different investigators
- Tools quickly become obsolete
- Trust in tools does not have a solid transparent basis
- Commercial tools exist, but they still have the problems mentioned in previous slides
- Forensic procedures and knowledge are not shared
  - It can be seen as valuable know-how, but on the other hand, at some point this knowledge is going to become crucial for the protection of our society
- Only community-backed initiatives with a wide participation can succeed
Past Computing Scenarios

1st generation
- Batch processing
- One user
- Specific Software

2nd generation
- Multiuser
- Real time
- Distributed systems

3rd generation
- AI
- OO technologies
- Parallelism

4th generation
- Expert systems
- PCs + internet

5th generation
- Global Computers
- Ubiquitous/Pervasive/AI
- SOC / Cloud Computing
- CPS, IoT, …
Future Scenarios

- Open systems
  - Distributed, heterogeneous, evolutive and large-scale

- Centralized control and possession are gone

- End of the concepts of system and application

- Increase in the needs for security and resilience

- Increased adaptability and context-awareness

- Self-*, Smart-*

5th generation
- Global Computers
- Ubiquitous/Pervasive/AmI
- SOC / Cloud Computing
- CPS, IoT, Smart-*
- Social media
- Big Data, …
Current situation

- Forensic analysis of current and future systems is becoming increasingly complex
- Anti-forensics are steadily advancing
- More and more court cases involve digital forensics
- Forensic capabilities can be an effective complement to security engineering

- Only way to overcome this situation is to make systems forensic-ready by-design
- This is related to another paradigm: accountability by-design
“By Design” Approaches
What is “by-design”

- This term refers to approaches in which some goal is promoted to become the focus of the design process.
- Essentially, it is about designing a system so that it is not possible not meeting that goal.
- One of the most well-known “by-designs” is privacy by-design.
  - Focuses on the design of systems that respect privacy.
  - Extreme example: A system that deliberately does not process private information.
  - Example 1: Deliberately using a low resolution video-camera in a surveillance system.
  - Example 2: A database of medical data that has no primitive for retrieving individual values and provides only aggregated, anonymised statistics.
There are many possible ways of applying the privacy related information to a system design. We choose to be practical and provide what is achievable to help designers.

**Good is enemy of best**

Our goal is that the proposed modelling will help achieving the compliance with a given concern, but we’re not pretending to guarantee it.

Sometimes we can get pretty close to guaranteeing compliance.

Most times the information is so vague or wide that we cannot ensure its compliance. In this cases, the modelling proposal will at least provide mechanisms to help an external actor to carry out the compliance check.

Our approach is to impose restrictions on the modelling, which allow us to validate the design. It is not possible to validate arbitrary system design models.
The model of a surveillance system under development (SUD) is a set of UML artefacts representing the SUD components and their relationships.

We include a special metaclass `SUDDescription` to provide general information, such as the purposes of the system, its limitations, etc. that applies to the whole SUD.
Each SUD has a description

- Descriptions include the system purposes, which are represented as stereotypes.
The purposes of the system are represented in a hierarchy according to the following metamodel.
Purpose limitation principle (EU Directive 95/46):
- The purpose of a surveillance system has to be specified, explicit and legitimate.

Implementation
- Engineers mark elements representing private data with the stereotype «privacy_sensitive». In some cases (e.g. Spanish data privacy law - LOPD) an optional attribute «sensitivity_level» (basic, medium, high) can be used.
- Engineers also mark elements that process private data with the stereotype «privacy_processing» with an attribute «purposes».

Validation
- All elements with the stereotype «privacy_processing» should have a non empty attribute «purposes». The declared purposes must have an associated description, and will be included in the documentation (produced by the GenerateDocumentation() method of the SUDDescription class)
Use limitation principle (EU Directive 95/46):
- Private data should not be used for purposes other than those specified.

Implementation
- Recall that engineers associate the SUDDescription class with a set of purposes using the attribute «purposes».
- Engineers mark elements representing private data with the stereotype «privacy_sensitive». In some cases (e.g. Spanish data privacy law - LOPD) an optional attribute «sensitivity_level» (basic, medium, high) can be used.
- Engineers also mark elements that process private data with the stereotype «privacy_processing» with an attribute «purposes».

Validation
- All values of the «purposes» attribute of all elements having the «privacy_processing» stereotype are included in the set of purposes associated to the SUDDescription class.
What do the examples show?

- The inclusion of design constraints can help defining automated validation rules for privacy by-design concerns.
- By-design approaches are useful only if you can validate such design.
- By-design approaches are useful as input to certification, compliance, and assessment.
- This approach can be used for other types of concerns.
Towards Forensic-readiness by design
Background

Traditionally
- Security
- Safety
- Reliability
- Privacy
- Accountability
- Quality
- ...

Today
- Non-functional aspects (call it Quality or whatever you like)
  - IT Security
  - Safety
  - Reliability
  - Privacy
  - Accountability
  - Quality
  - ...

City University, December, 2015
Towards Forensic-readiness by design
Security engineering today

“It has become clear that, generally, engineers have not had sufficient training nor been encouraged to have a mind-set that considers how an adversary might thwart their system”

“Now is the time not only for better defensive measures but also for cybersecurity standards and best practices that consider the entire technology life cycle”

IEEE Senior Member Greg Shannon, chair of the IEEE Cybersecurity Initiative.
We’re in a daunting situation:

- Future IT Systems
  - will be composite, dynamic, extremely complex and highly interconnected,
  - will be immersed in every aspect of our lives, and
  - will be the target of powerful, highly-motivated attackers,
- but still we do not have rigorous expertise-based and computer-assisted engineering approaches to help system developers in the formidable task of designing and maintaining those systems reasonably secure.
Security solutions are not obvious, stable, simple, eternal, ...

What worked well in some context or for some time will not work forever (or violate essential security)

Context
- level of security
- security requirements
- technologies
- stability
- dynamism
- control
New complex systems ▶ new and complex security problems

Secure systems must prevent not only technical, but also social engineering attacks (e.g. phishing, pretexting, etc.)

Software Security vs. Information Security vs. System Security

- Passive and/or Active Assets vs. only Passive Assets vs. Process Security
Classifying approaches for providing security

- Depending on the **moment:**
  - A priori (aka “by-design”)
  - A posteriori (aka “penetrate and patch”)

- Depending on **coverage:**
  - Total (aka “Paranoid”)
  - Selective (aka “Best effort”)

- Depending on the **conceptualization:**
  - Perimeter (“The walled fortress”)
  - Integral (“The open metropolis”)

- Depending on the **strategy:**
  - Preventive (e.g. based on properties)
  - Reactive (e.g. based on threats)

- Depending on adversary models, their capabilities, models, technologies, parties, activities, etc…
The term “Security engineering” groups together a wide set of techniques and approaches including:

- Threat-based initiatives
- Risk-based initiatives
- Formal methods-based initiatives
- Pattern-based initiatives
- Model-based initiatives
- Other techniques and approaches (language-based approaches, knowledge engineering...)
- Processes / Preventing Software Errors / Testing / Education / etc.
Conclusions

- Security engineering is not yet engineering
- Many open issues currently and more appearing everyday
- Even with perfect security engineering, there are many sources for insecurity in IT systems
- Attacks and incidents will continue to happen
- Sometimes knowing you’ll be made accountable for bad behaviour is more effective than preventing it.
An example of a forensic-by-design approach: Monitoring Capable Code
Monitoring choices

- Monitoring variants
  - Internal vs. external event generation
  - Internal vs. external monitoring
    - Internal monitors run in the same trust domain as applications
    - External monitors run in a different trust domain
  - Observe vs. act
  - Reactive vs. predictive...

- Monitoring schemes
  - Self-monitoring applications vs.
  - Monitoring unaware applications vs.
  - Instrumented applications.
Applications contain self-behaviour checking code.

Advantages
- Transparent for the platform
- Less likely to introduce vulnerabilities on the platform

Drawbacks
- Requires extra programming effort
- Expensive to change when the system is in operation
- Some checks are not possible (e.g. interactions)
- Some capabilities are limited (e.g. reaction)
- The platform does not have control over the application or the checks
Requires generic monitoring specifications

Advantages
- More flexible when operational environments change and applications change,
- Can be applied with all applications (e.g. legacy apps.)

Drawbacks
- Less efficient
- More error-prone
- Responsibility / Accountability issues
Instrumented applications

- Applications contain monitoring-specific code
  - additional event generators, monitoring rules, etc.

- Advantages
  - Increased fault/attack tolerance
  - Monitoring can be fine-tuned to the application
    - Application-specific rules can be used

- Drawbacks
  - More expensive
  - Limited adaptability
  - Event genuineness may be an issue
Instrumented applications

- Necessary in certain circumstances
  - Different instances of the same application running in different VEs and in different physical environments
  - Only choice for monitoring internal events (e.g. an application changes from state1 to state2)

- Schemes
  - Source / executable code
  - Automatic / Assisted / Manual
**Static** bytecode verification before execution

- A central element of the Java security architecture (together with stack inspection).
- Limited set of verifiable properties
- Mature theoretical framework (in fact 2 of them: type-based PCC and logic-based PCC)
- Has already been applied to Virtual Machines

*Source: MOBIUS Project. Deliverable D4.1*
New tools: Monitoring Capable Code (MCC)

- (Bytecode) dynamic verification during execution
  - Wide set of verifiable properties
  - No mature theory available (PASSIVE defined it)
  - Can be combined with PCC and Protected Computing

Diagram:
- Source program
- Monitoring specification
- Compiler
- Monitoring-instrumented code (event capturers)
- Monitoring rules
- Execution
- Monitor
- Code consumer (virtualized environment)
Applications must be designed for forensic readiness

Software has attached a set of rules similar to the PCC approach

- App code + data
- App Behaviour Security Model (ABSM)

This part is relevant for the runtime framework (e.g., in a VM)

This part is relevant for the Monitor

Towards Forensic-readiness by design

City University, December, 2015
Application security model

- Includes monitoring rules (such as event E1 is generated when X happens)

**App behaviour security model**

**Event declaration**

- Ev1 { Application specific event, VMM event}
- Ev2 { Application specific event, VMM event}
- ...

**Monitoring rules**

- R1:: Seen(Ev1, t) → Seen(Ev2, t+100)

**Monitoring Reactions**

- R1: App.Halt
Application-specific events

- A mechanism for communicating app-specific events is needed (e.g. applications write on a specific file such as monitoring log)
- Monitoring-unaware applications must be supported
- Event genuineness must be determined
- Event privacy issues
  - Internal application events can be shared
Application interaction security model

To model the monitoring of problems derived from the interaction of this application with other applications running in the same platforms

This specification can come from external parties (e.g. Cloud Provider)
AISM rules

- These rules monitor the use of shared resources
- Rules are mostly
  - based on probabilities, generalization and abduction
    - \( A \land B \) then probably \( A \Rightarrow B \)
    - \( A \Rightarrow B \land B \) ... then “probably” \( A \)
  - focused on interaction and
  - not application-specific
- Example: if more than 60\% of the times, after App1 fails, we observe App2 failing, we can conclude that App1 and App2 have an unexpected interaction of the form
  - \( \text{Fail(App1)} \Rightarrow \text{Fail(App2)} \)
App1 uses key K1 from keystore KS1 for encryption.
- We have proofs that none KS1 or App1 does ever leak K1

App2 also uses K1 from KS1, but in this case we know it is vulnerable to an attack that can extract K1

In this case App1 becomes vulnerable too and should be warned
Monitoring Forwarding control

- App code + data
- App Behaviour Security Model (ABSM)
- Application Interaction Security Model (AISM)
- Monitoring Forwarding Control

Editable
Parameterizable
Configurable
(by user / admin)
Monitoring forwarding control

- In the case that we do not want to record some details
- Privacy must be maintained (e.g. A specific parameter of a violated rule reflects that the access to the file gambling.avi is not granted)
- Monitoring forwarding control rules can be edited / parameterized / configured by administrators or users
Rule R1 is violated by App1 and 1 sec after, rule R2 is violated 3 times by App2
As already mentioned, this indicates that a correlation exists
Next step is to notify someone that “R1 is always violated after R3 is violated three times”
This has to be done in a privacy-respectful and based on a need-to-know policy
For performance, privacy and efficiency issues, it is necessary to send only the useful information
Multi-layer monitoring architecture

- A1
- LAS
- IPS
- GAS (A1)
- App Behaviour Security Model (ABSM)
- App Behaviour Security Model (ABSM)
- Application Interaction Security Model (AISM)
- Monitoring Forwarding Control

- App code + data

Towards Forensic-readiness by design

City University, December, 2015
Monitoring Architecture Components

- Local Application Surveillance (LAS)
  - Monitors rules that are **specific** for the monitored application and designed to detect **unexpected behaviour** of the application resulting from **implementation flaws**. Rule violation can be communicated to IPS. (Application Behaviour Security Model)

- Intra-Platform Surveillance (IPS)
  - A.k.a. Horizontal monitor. Rules are **general** and try to detect **unexpected interactions** between applications. Rule violation can be communicated to GAS. (Application Interaction Security Model and Monitoring Forwarding Control)

- Global Application Surveillance (GAS)
  - A.k.a. Vertical monitor. Executes monitoring rules that are **specific** for the monitored application and try to detect **design flaws** of the application. (Special type of Application Behaviour Security Model)

- Trusted, authenticated LAS, IPS and GAS registers can be extremely useful for forensic analysis!
NOVA VM Architecture
NOVA architecture

Towards Forensic-readiness by design

City University, December, 2015
Uses of the approach

- Ensure application is forensic-ready
- Detect misbehaviour of applications
  - React to misbehaviour (halt, reset or change something – resources, priority, permissions... – )
- Support the evolution of applications
  - Detect errors in the application model
  - Detect errors in application implementation
  - Detect errors due to unexpected interactions
  - Report errors to application providers
    - Under control of users
Conclusions

- Forensic analysis of IT systems will become a real necessity
- Future systems will increase in complexity, distribution, lack of centralized ownership and control, etc.
  - This will make forensic analysis much harder or even impossible
- Only way to overcome this situation is to make systems forensic-ready by-design
THANK YOU FOR YOUR ATTENTION!

QUESTIONS?

City University, December, 2015
Towards Forensic-readiness by design