ROMA MMXVI

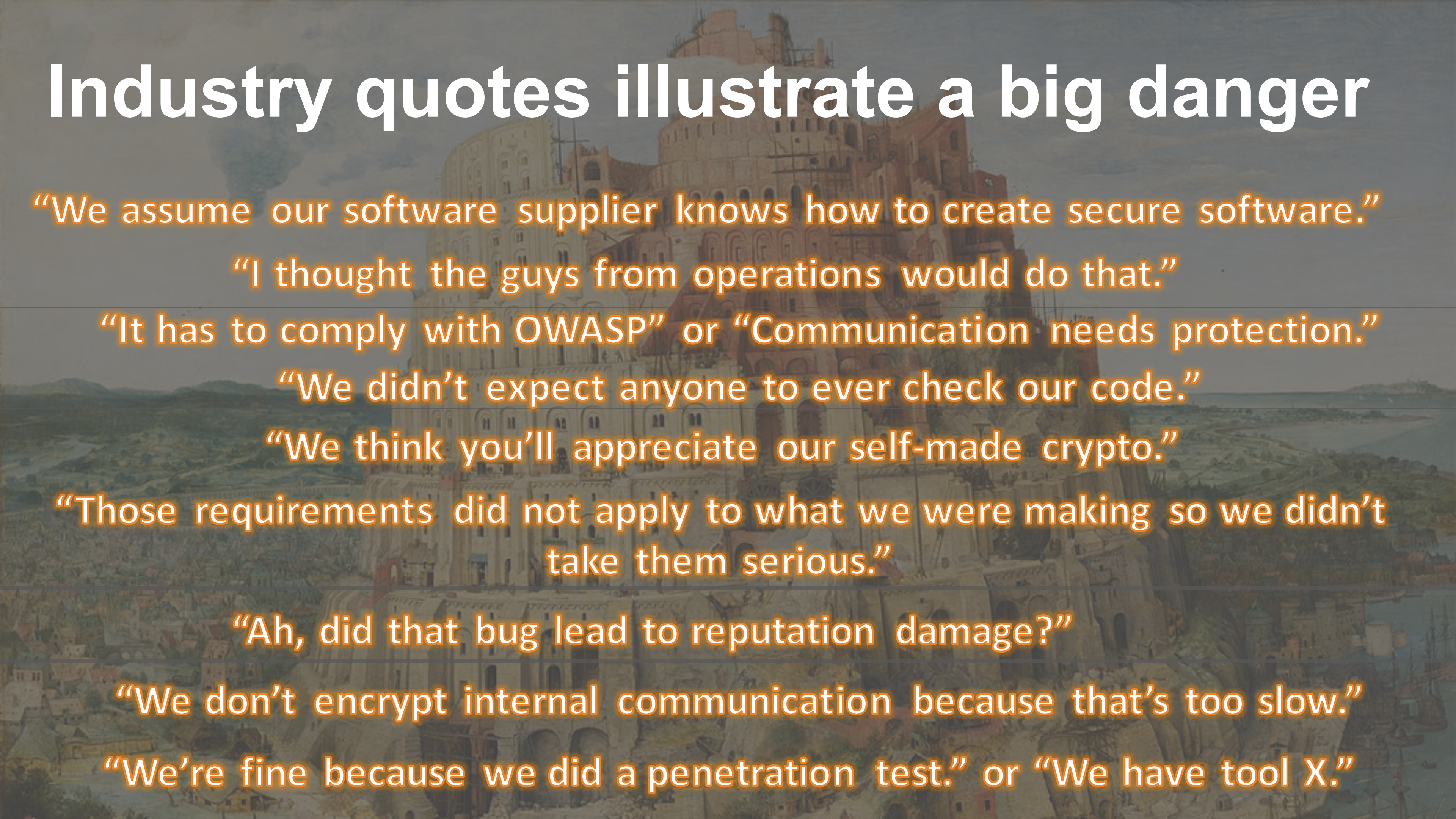
APPSEC EUROPE

Grip on Secure Software Development

How a Dutch government standard conquered the danger of haste

Rob van der Veer July 1st 2016

Industry quotes illustrate a big danger



“We assume our software supplier knows how to create secure software.”

“I thought the guys from operations would do that.”

“It has to comply with OWASP” or “Communication needs protection.”

“We didn’t expect anyone to ever check our code.”

“We think you’ll appreciate our self-made crypto.”

“Those requirements did not apply to what we were making so we didn’t take them serious.”

“Ah, did that bug lead to reputation damage?”

“We don’t encrypt internal communication because that’s too slow.”

“We’re fine because we did a penetration test.” or “We have tool X.”

Introducing

Rob van der Veer
Principal consultant
Software Improvement Group

r.vanderveer@sig.eu
@robvanderveer
+31 6 20437187

www.sig.eu/security

Agenda

1. The danger of haste
2. The ‘Grip on Secure Software Development’ initiative
3. How the method works
4. Lessons learned
5. Future work

ROMA
MMXVI

APPSEC
EUROPE

So what is happening?

Clients and suppliers[1] don't take time to arrange secure software:

- Requirements are lacking or vague and unspecific
- Who does what is unclear
- No security dialogue
- Proven technology is ignored
- Developers not informed on risks
- No risk management
- Tools and pentesting regarded as panacea
- Looking at code is avoided

[1] Either internal development department or external vendor/system integrator

For some mysterious reason, people are trying really hard to avoid looking at code. Yet, there are all kinds of expectations about the quality of code. It needs to be maintainable. Security, reliability and performance need to be built in.

The danger of haste

- Quality becomes an afterthought: no security by design
- Test & fix at the end:
  - Time pressure only allows for quick fixes
  - Effort x 100 (1)
  - Tests miss weaknesses
- Some risks are wrongfully ignored
- Result:
  - lower security and higher cost
  - Also: disturbed relations between parties involved

(1) B. Boehm and V. R. Basili. Software defect reduction top 10 list. IEEE Computer, 34:135–137, 2001.

What is the cause?

- Clients want visible results quick
- You can’t control what you don’t measure
- Clients have little experience with security
- Suppliers love to implement their own ideas
- <u>There was no shared idea on how to start and what to do</u>

The *Grip on SSD* initiative

- Standard to arrange software security without intervening with development
- Developed by the Dutch government organisation ‘Center for Information security and Privacy protection’ (CIP) and many others
- Main points addressed:
  - Risk management
  - Security requirement dialogue
  - Verification strategies
- A free common standard to guide both clients and suppliers

Handover of the method by CIP to the Dutch central government CIO, Dion Kotteman

Grip on SSD practitioner community

- 30 organizations (government, system integrators, experts)
- Share experience and grow the standard
- Newsletter, regular meetings, working groups
- Links with OWASP, Dutch CERT, ENISA

Ministerie van Binnenlandse Zaken en Koninkrijksrelaties

Ministerie van Volksgezondheid, Welzijn en Sport

Ministerie van Financiën

Dienst ICT Uitvoering
Ministerie van Economische Zaken

Nationaal Cyber Security Centrum
Ministerie van Veiligheid en Justitie

Agentschap NL
Ministerie van Economische Zaken, Landbouw en Innovatie

Belastingdienst

SVB

UWV

IBM

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

SIG Software Improvement Group

ORDINA
CONSULTING | ICT | OUTSOURCING

SOGETI

centric

CGI

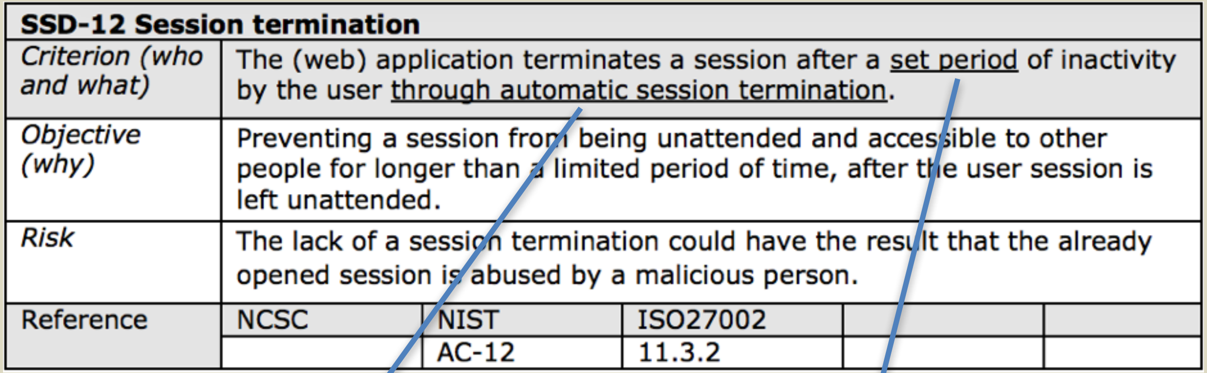
pijler 3 – SSD-processen
CTP
NOOIT GEDACHT
DAT BIJ DEVOPS
We are Centric. We believe in people. We are convinced
when the right people come together, they can reach for the
highest. That is why we invite professionals, partners and clients
t and combine their talents, knowledge and ideas.
nect.eng e.succe

Products at www.griponssd.org

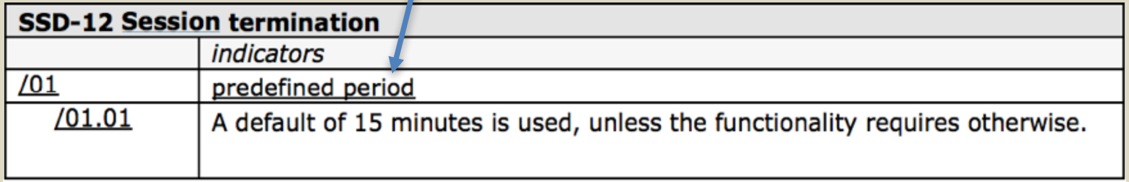
- Method handbook
- Baseline requirements
- Verification guide
- Training slides for testers
- Contract templates

Grip op
Secure Software Development
Het proces
CIP
Dit is een publicatie van het CIP:
Centrum Informatiebeveiliging en Privacybescherming

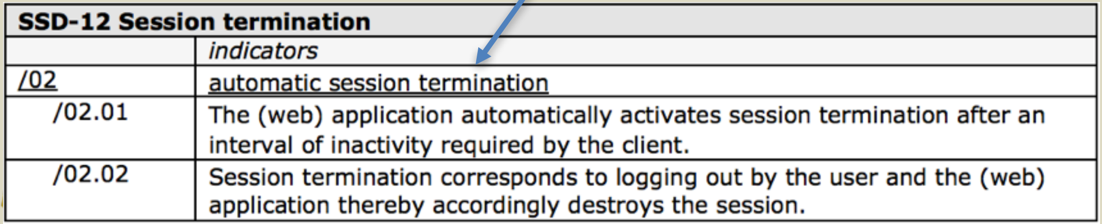
SIVA notation of requirements



SSD-12 Session termination					
Criterion (who and what)	The (web) application terminates a session after a set period of inactivity by the user through automatic session termination.				
Objective (why)	Preventing a session from being unattended and accessible to other people for longer than a limited period of time, after the user session is left unattended.				
Risk	The lack of a session termination could have the result that the already opened session is abused by a malicious person.				
Reference	NCSC	NIST	ISO27002		
		AC-12	11.3.2		



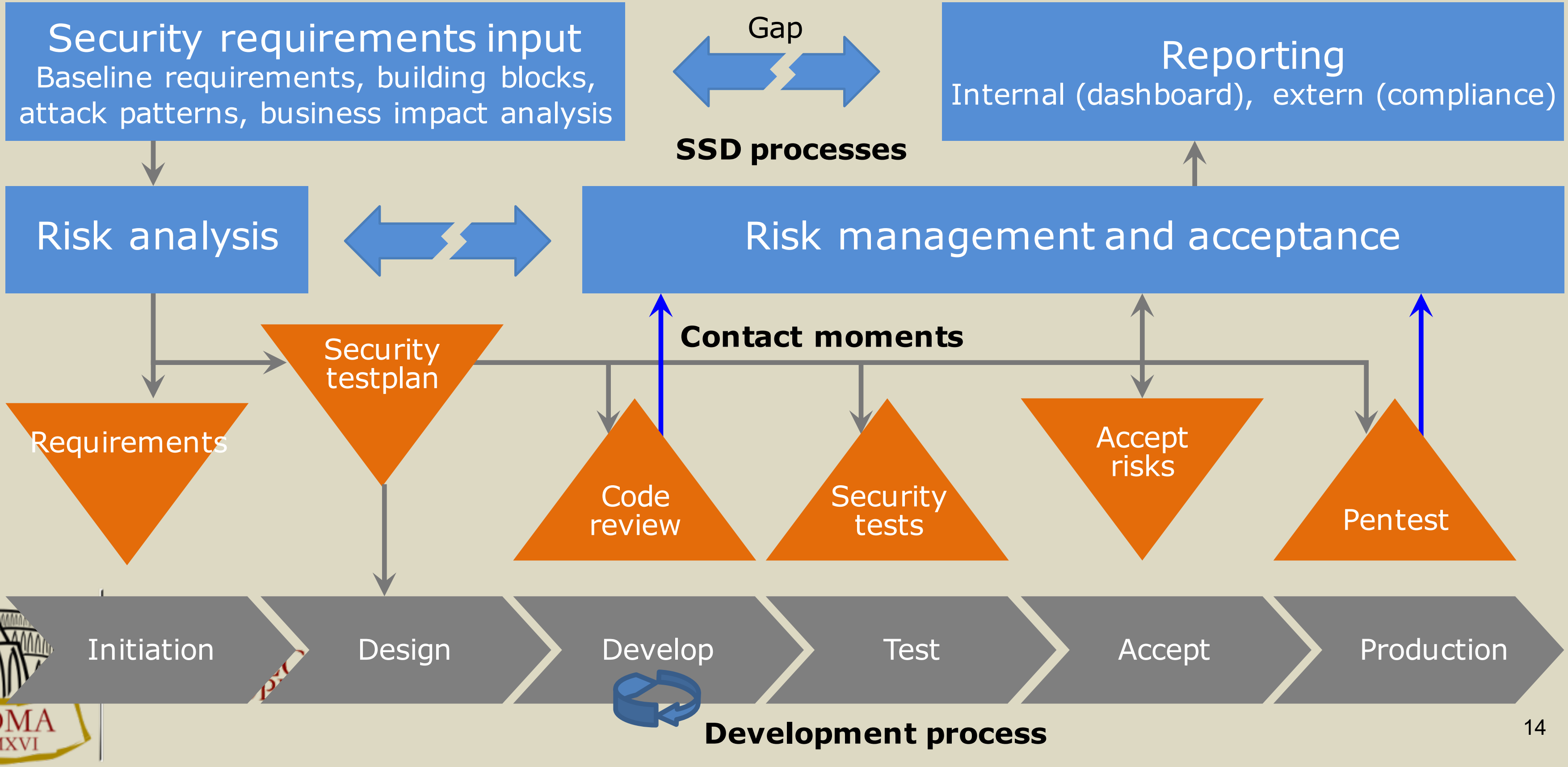
SSD-12 Session termination	
	indicators
/01	predefined period
/01.01	A default of 15 minutes is used, unless the functionality requires otherwise.



SSD-12 Session termination	
	indicators
/02	automatic session termination
/02.01	The (web) application automatically activates session termination after an interval of inactivity required by the client.
/02.02	Session termination corresponds to logging out by the user and the (web) application thereby accordingly destroys the session.

See “Siva” by Wikram Tewarie, VU University Press, 2014

The method



Security requirements input
Baseline requirements, building blocks, attack patterns, business impact analysis

Gap

SSD processes

Reporting
Internal (dashboard), extern (compliance)

Risk analysis

Risk management and acceptance

Contact moments

Requirements

Security testplan

Code review

Security tests

Accept risks

Pentest

Initiation

Design

Develop

Test

Accept

Production

Development process

Key best practices

- Have standard requirements
- Risk analysis for every project and tailor the requirements
- Be clear, but do not try to be complete
- Comply or explain
- Keep track of (accepted) risks
- Perform penetration tests
- Agree with supplier on early independent code reviews

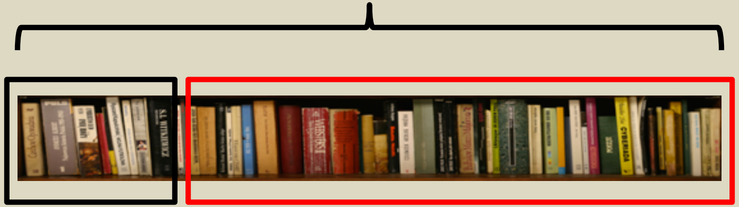
Lessons – implementing Grip on SSD

- First: minimum baseline, dashboard and mandatory risk analysis for every IT project
- Acquire/contract skills for the above
- Next, extend supplier contracts
- Manage expectations and increase maturity step by step, following the included maturity model

Lessons – setting requirements

- Use the requirements to start the conversation.
- You cannot cover all security specifics in requirements
  - Too much to cover: vast area and many variations
  - Too dynamic, constantly evolving

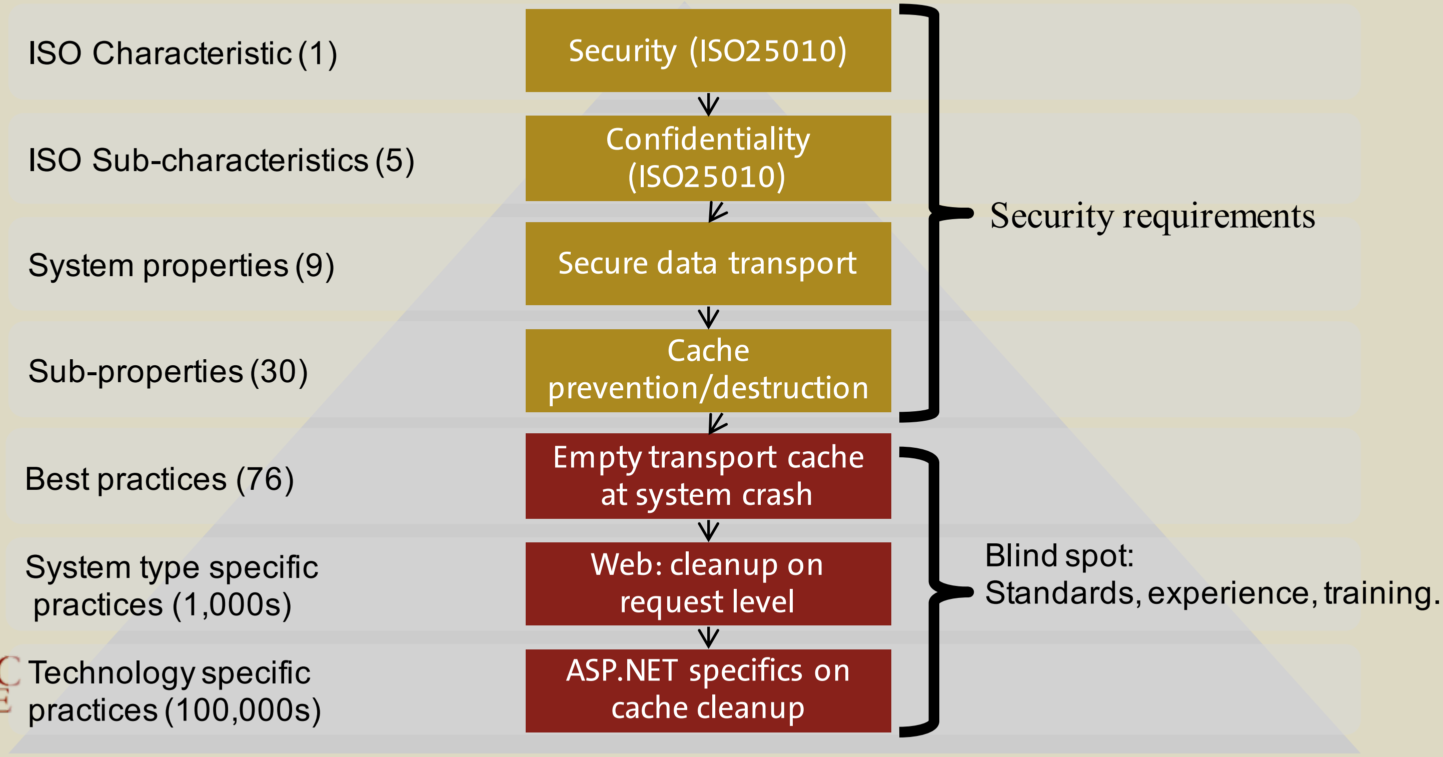
How to write secure software



Security requirements

Gap: not required, but expected or suggested in the requirements

The security requirements catch 22



ISO Characteristic (1)	Security (ISO25010)
ISO Sub-characteristics (5)	Confidentiality (ISO25010)
System properties (9)	Secure data transport
Sub-properties (30)	Cache prevention/destruction
Best practices (76)	Empty transport cache at system crash
System type specific practices (1,000s)	Web: cleanup on request level
Technology specific practices (100,000s)	ASP.NET specifics on cache cleanup

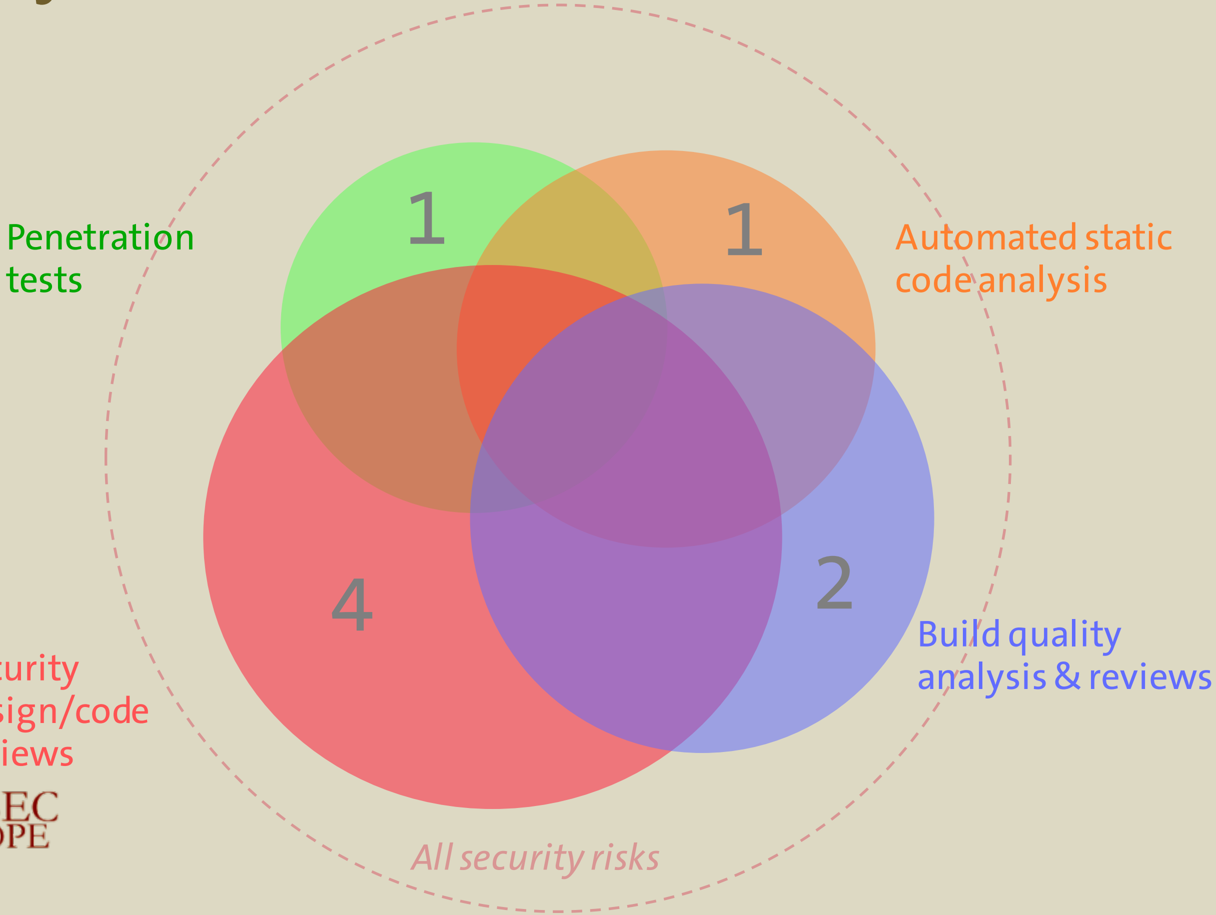
Security requirements

Blind spot:
Standards, experience, training.

Lessons - verification

- Verify the product, do not fix the process and then hope
- Don’t rely on pentesting, or scan tools only
- Do not limit verification to the set requirements
- Review code by experts
- Don’t limit code review to security flaws
    - Privacy
    - Maintainability

Security verification *effectiveness*


Penetration tests
Automated static code analysis
1
1
4
2
Build quality analysis & reviews
Security design/code reviews
All security risks

In summary, Grip on SSD provides

- Security by design
- More security for the money
  -> less incidents, less impact -> less damages
- Insight in risks
- A shared way of working in the industry
- Better relations between parties involved

Future work

- More publications
  - Working on ‘Privacy by design’
  - Adding code review guidelines
  - Adding Grip on SSD maturity self-assessment
- Internationalization
  - Translations done with IBM, Centric and Sogeti
  - Increase collaboration through OWASP