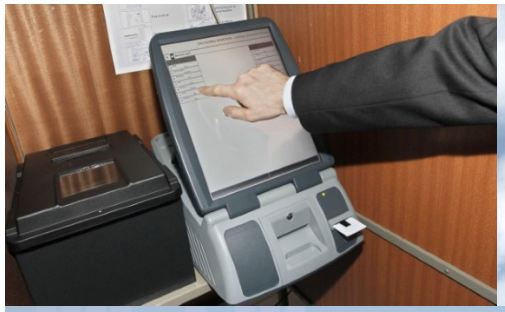
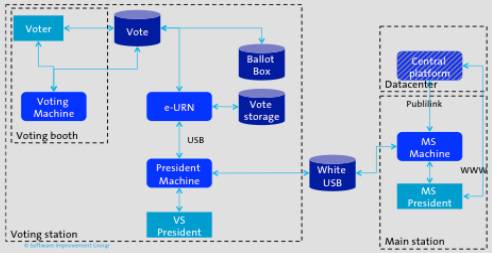


```
int tls1_process_heartbeat(SSL *s) {
   /* Controlled by user */
   unsigned char *p = &s->s3->rrec.data[0], *pl;
   unsigned int payload;
   unsigned int padding = 16; /* Use minimum padding */
...
   /* The first two bytes of p represent the length of the
   * payload and is put in variable payload
   */
   n2s(p, payload);
   pl = p;
...
   unsigned char *buffer, *bp;
   buffer = OPENSSL_malloc(1 + 2 + payload + padding);
   bp = buffer;
...
   /* Enter response type, length and copy payload */
   bp++ = TLS1_HB_RESPONSE;
   /* Puts payload length in two bytes of bp. */
   s2n(payload, bp);
   /* Copies payload length of pl to bp */
   memcpy(bp, pl, payload);
```



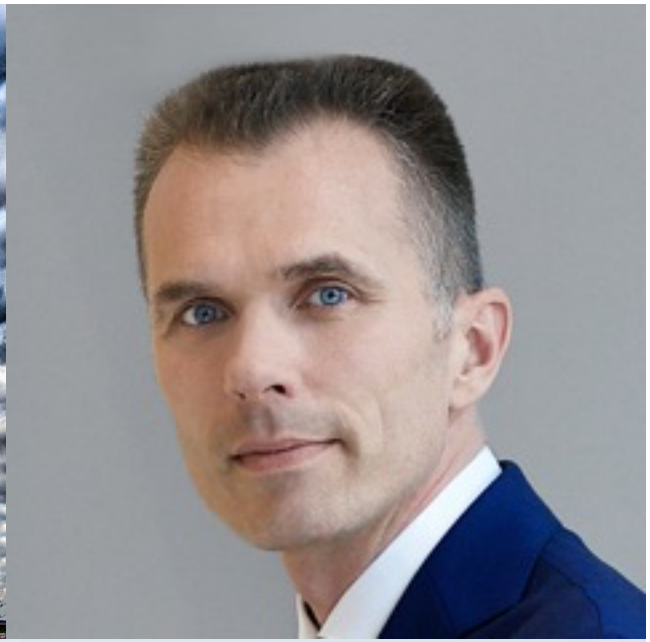
Application security of the Belgium electronic voting system

Rob van der Veer - SIG

OWASP AppSecEU 15

Amsterdam, The Netherlands

Introduction



Rob van der Veer
Principal consultant
Software Improvement Group

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

www.sig.eu

Software Improvement Group

OWASP AppSecEU 15
Amsterdam, The Netherlands

Agenda

- Background
- Publishing source code
- How to analyze source code
- Our approach
- Key take-aways

vodafone NL 4G 18:06 45%

Terug datanews

13.06.2014 - 16:31

NEDERLANDERS ANALYSEREN BELGISCHE VERKIEZINGSSOFTWARE

SIG onderzoekt de broncode van de verkiezingssoftware die de FOD Binnenlandse Zaken tijdens de verkiezingen van 25 mei heeft gebruikt bij de stemcomputers en telsystemen.

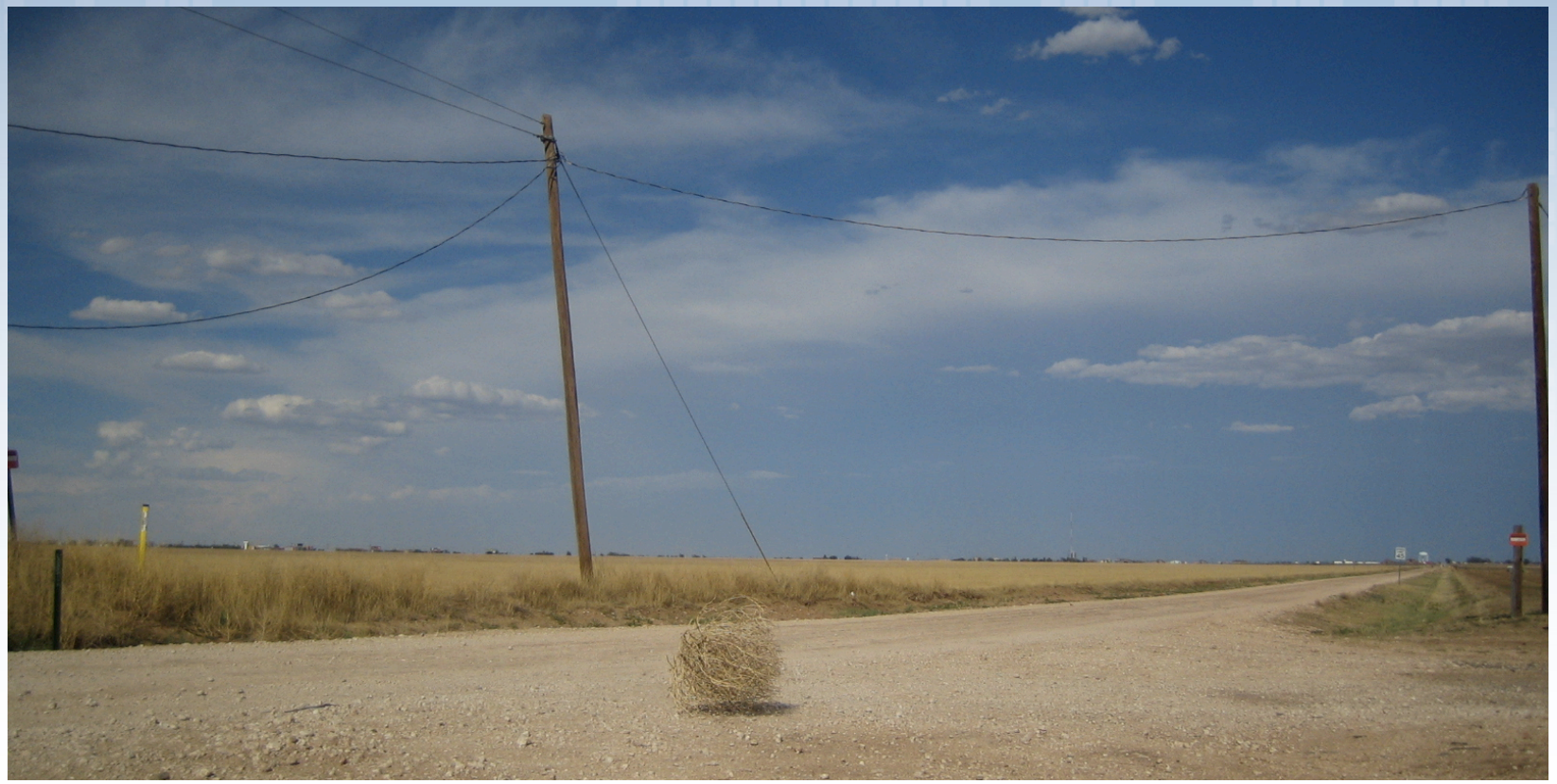
SIG (Software Improvement Group) pleit al langer voor een doordacht beleid rond technische kwaliteit bij de ontwikkeling van software. Het Nederlandse bedrijf stelt dat kwaliteit het risico op mislukte ict-projecten vermindert. Aandacht voor softwarekwaliteit kan op die manier heel wat geld opleveren. Op 2 juni

T- T+

Background

- May 2014: publishing of e-voting source
- European, federal & regional elections
- General e-voting sensitivities
- Sensitivities for Belgium
  - Reliability incidents
  - Usability incidents

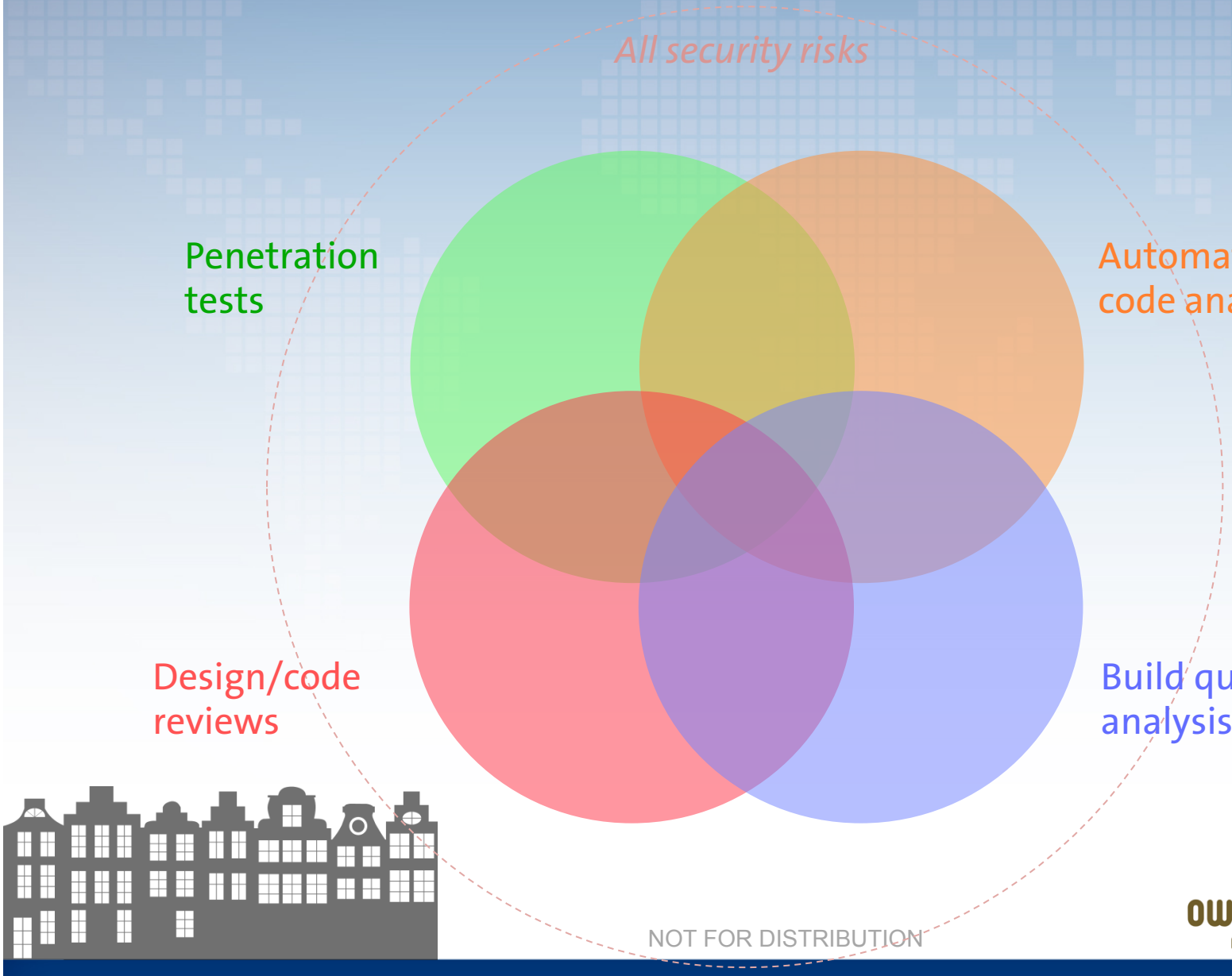
What happens when you publish source code?



What happens when you publish source code?

- It radiates confidence
- People make very little effort to look at it (Heartbleed, anyone?)
- Setting up test environment is hard
- Good code review is hard, especially without interviews
- Code reviewing is sticking your neck out

How to analyze software security

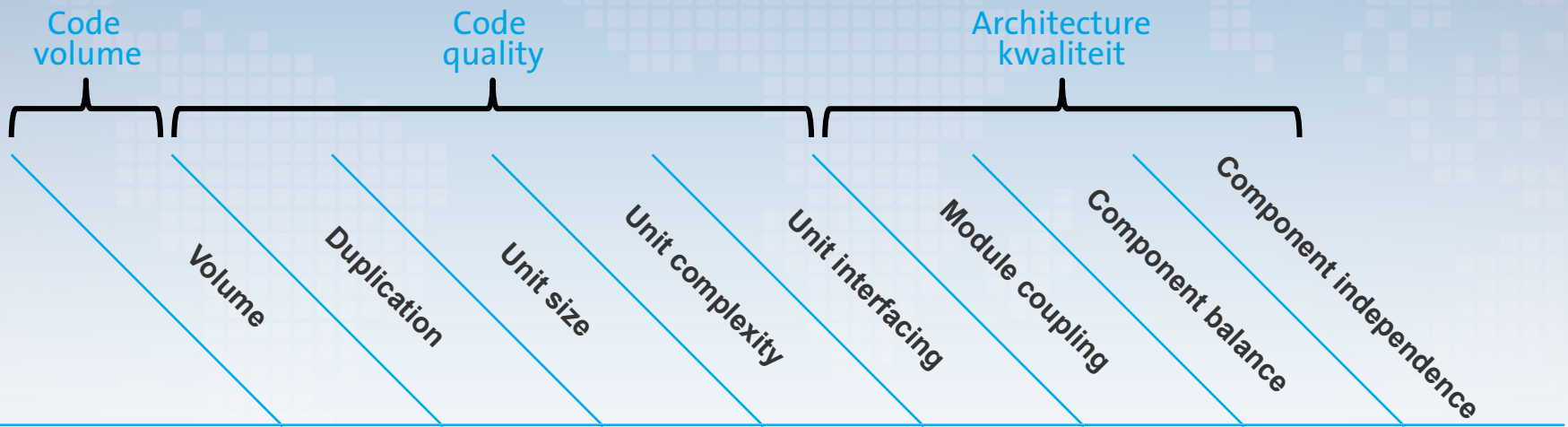


Review system properties to measure ISO25010 security

	Secure data transport	Identification strength	Access management strength	Session management strength	Authorized access	Input and output verification	Secure data storage	Evidence strength	Secure user management	
	★★☆☆☆	★☆☆☆☆	★★☆☆☆	★★★★☆	★☆☆☆☆	★★☆☆☆	★★★★☆	★★☆☆☆	★★☆☆☆	
Confidentiality & Integrity	X				X	X	X			★★☆☆☆
Non-repudiation & Accountability		X						X		★★☆☆☆
Authenticity			X	X					X	★☆☆☆☆

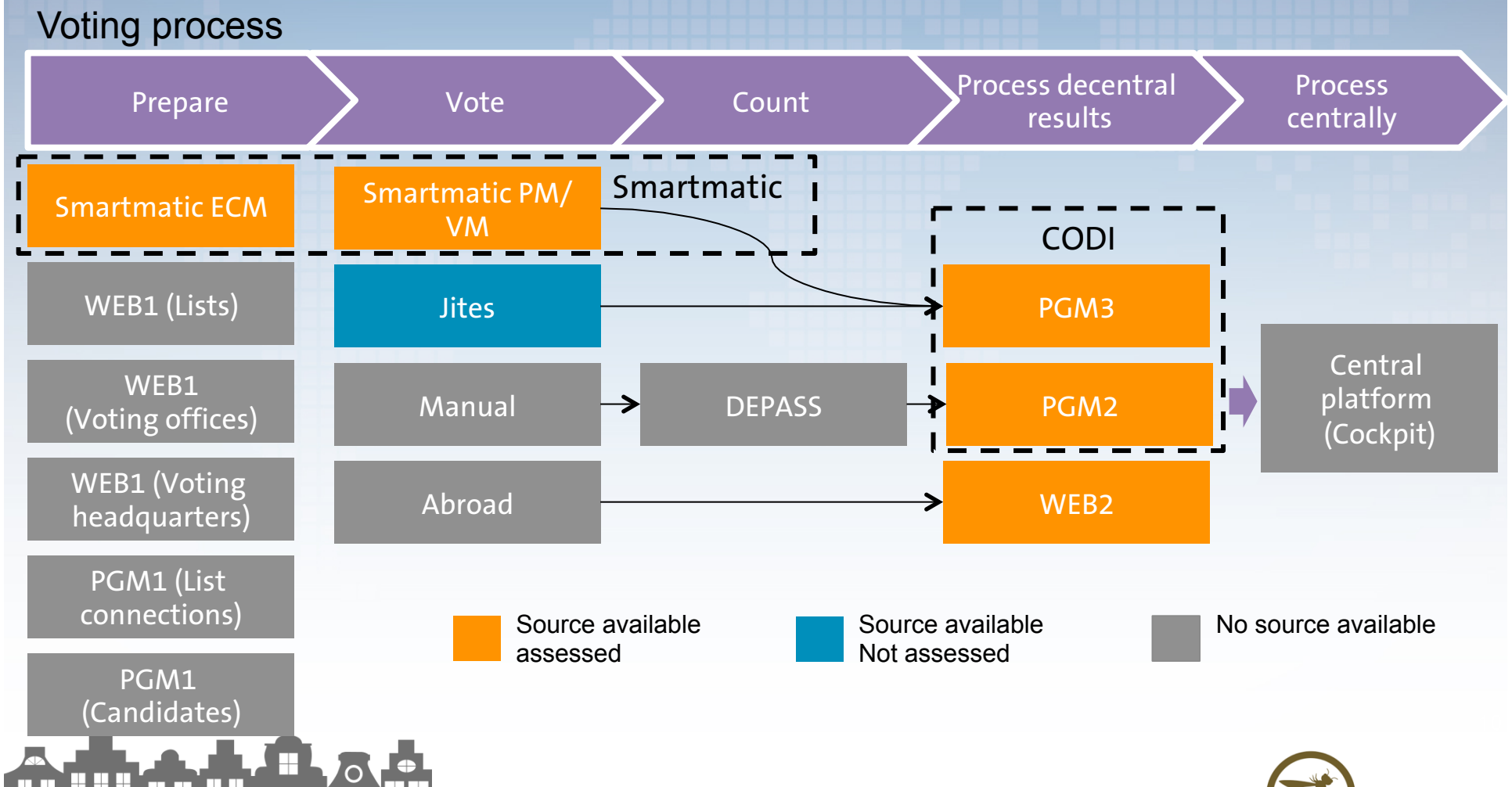
Final result: ★★☆☆☆

Measure ISO25010 maintainability

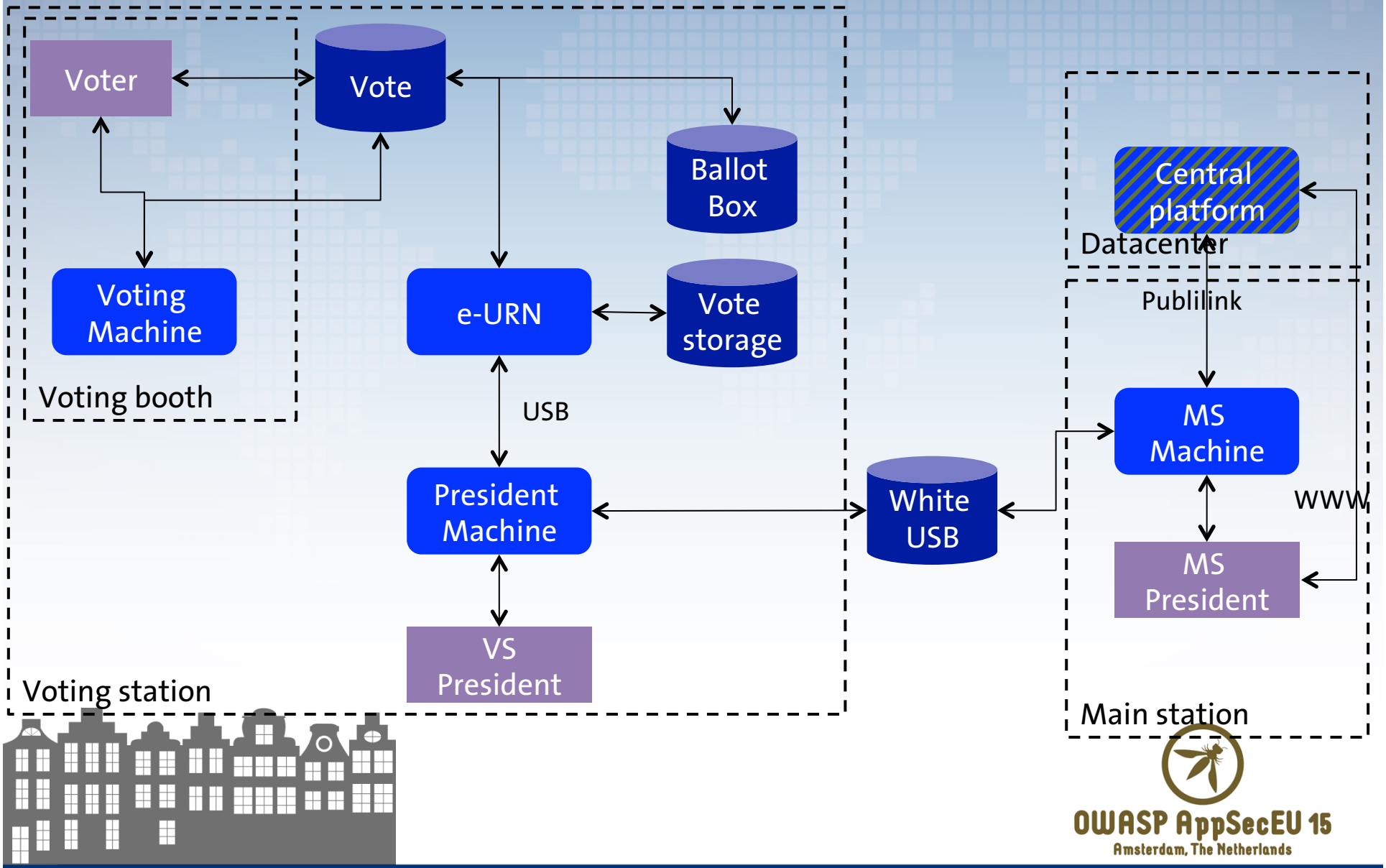


Analysability	X	X	X				X	
Modifiability		X		X		X		
Testability	X			X				X
Modularity						X	X	X
Reusability			X		X			

E-voting software overview



Threat modeling



Set SMART non-functional requirements from the start, including who does what

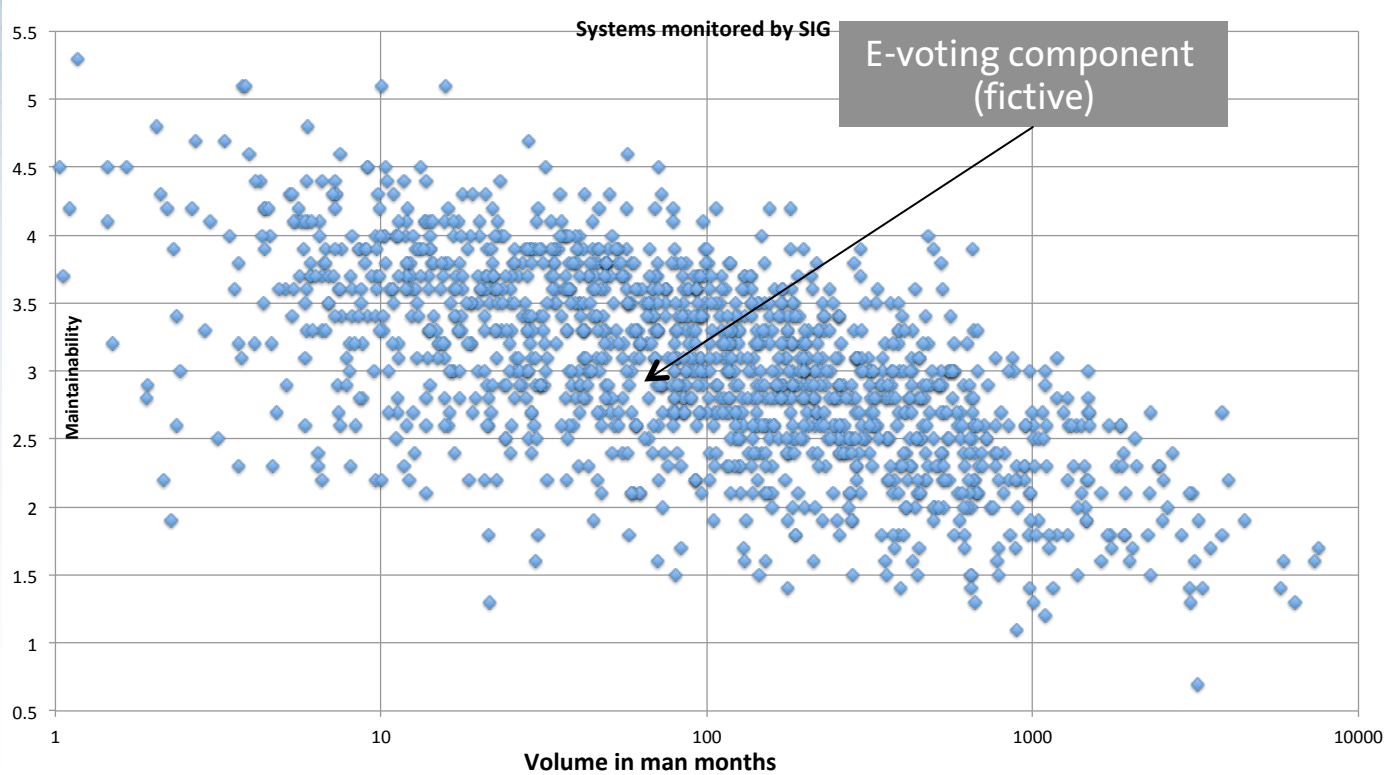
E-Voting requirements

22 functional requirements.

No nonfunctional requirements on building security in

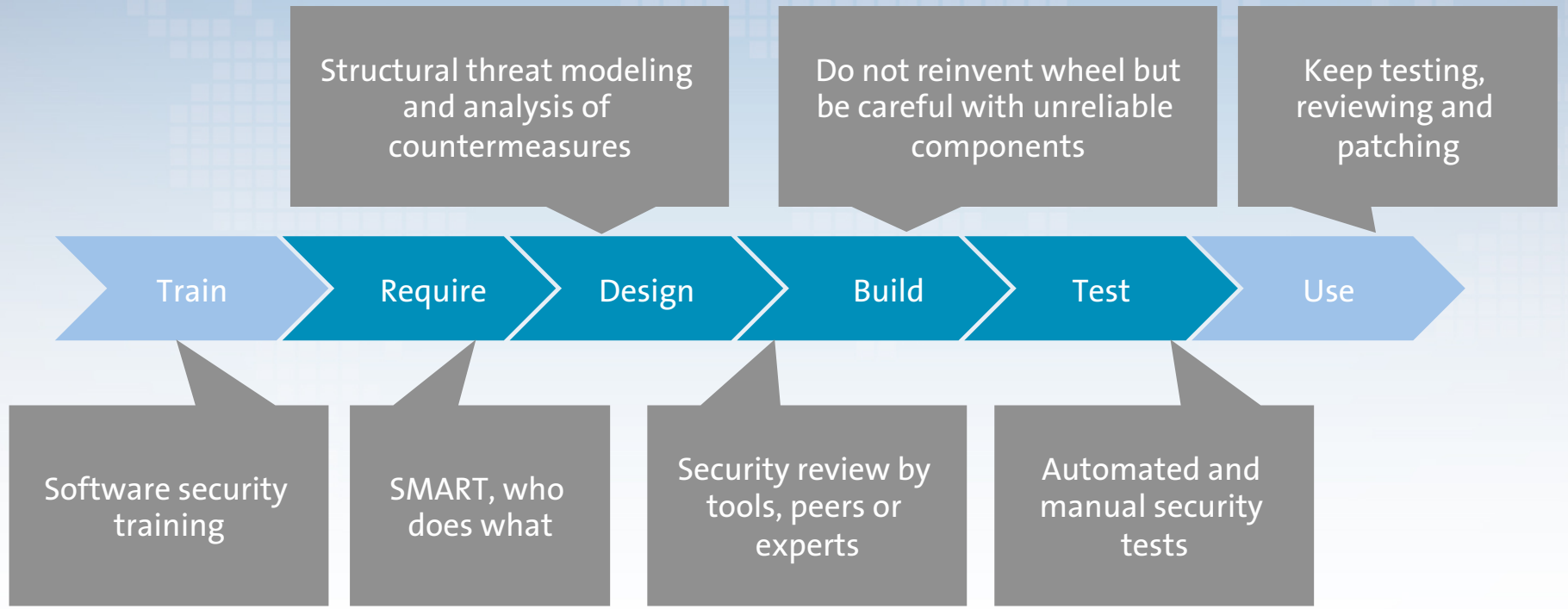
No requirements for other applications in chain

Also require maintainable software as technical debt fuels security incidents



OWASP AppSecEU 15
Amsterdam, The Netherlands

Mature the secure SDLC



OWASP AppSecEU 15
Amsterdam, The Netherlands

Manage your risks