MathWorks
AUTOMOTIVE
CONFERENCE 2019

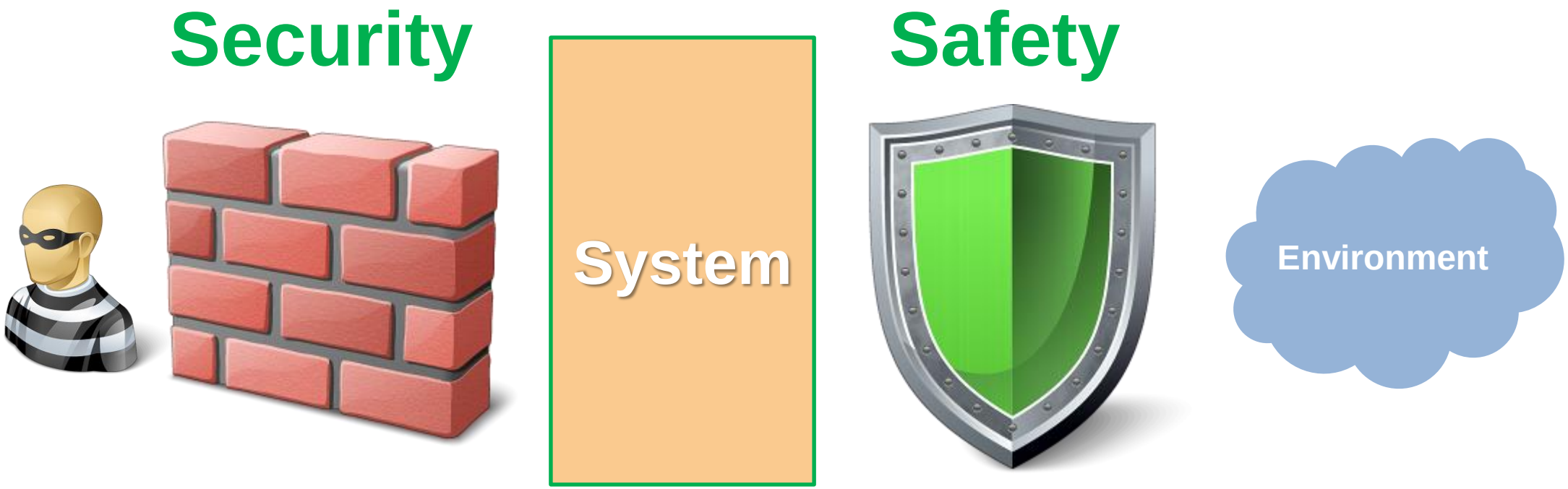
Secure Coding Guidelines를 위한 Polyspace 제품군

유용출

Agenda

- Safety vs Security
- Why Check Secure Coding Guidelines?
- How to Apply Secure Coding Guidelines?

Safety vs. Security



Note: Security issues may cause safety issues

5W1H - Secure Coding Guidelines

Who	You or Your colleagues (Developers, QA, QE)
Where	at workplace
What	Most of your software running on target
When	Every day, week, month

5W1H - Secure Coding Guidelines

Why

1. Security issues may lead catastrophe
2. Required by your customers

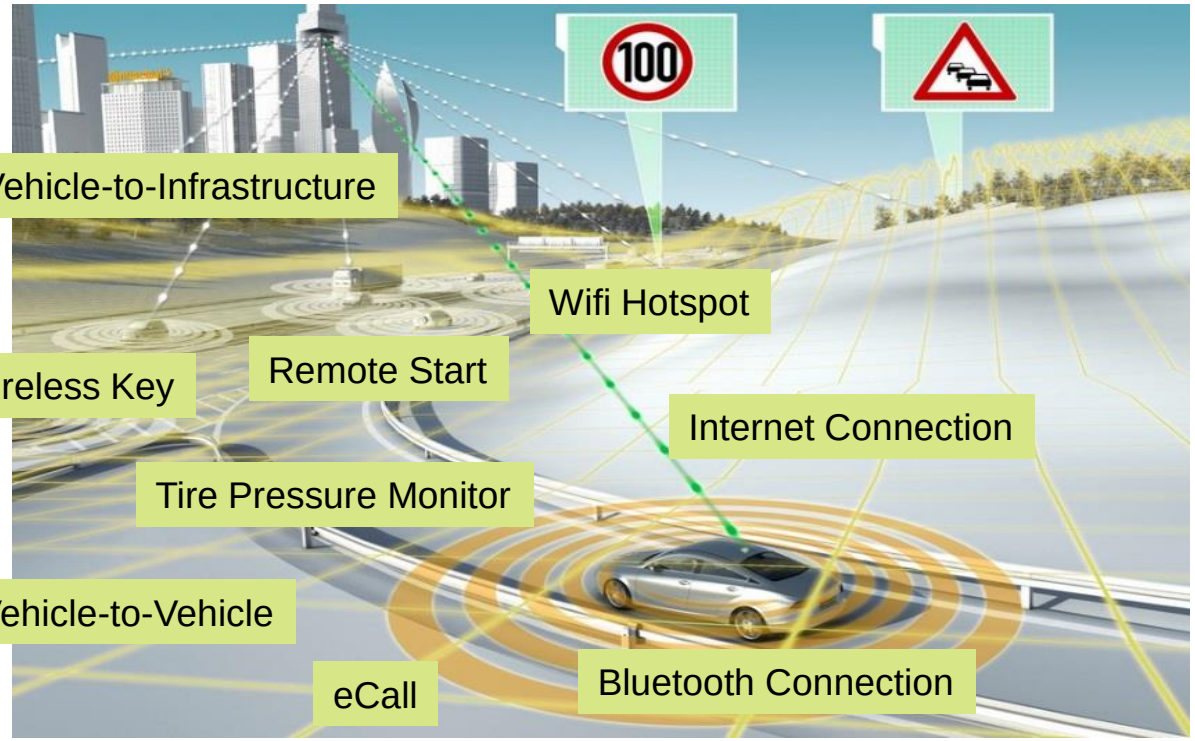
How

1. Use static analysis tool for security
2. Make analysis jobs automated

Why Check Secure Coding Guidelines?

Cybersecurity – Emerging Topic in the Auto Industry

On-Board & V2X Communication

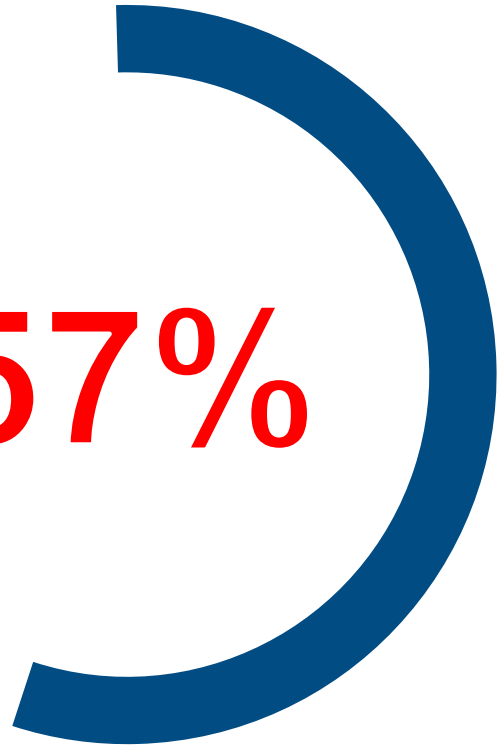


- Growing communication of on-board systems, sensors and external sites
- Car becomes another node of IoT
- Security of automotive embedded systems increasingly important (possible cyber attacks)

FCA recalls 1.4 Million cars after Jeep hack

https://youtu.be/MK0SrxBC1xs

Security in Consumers' Mind



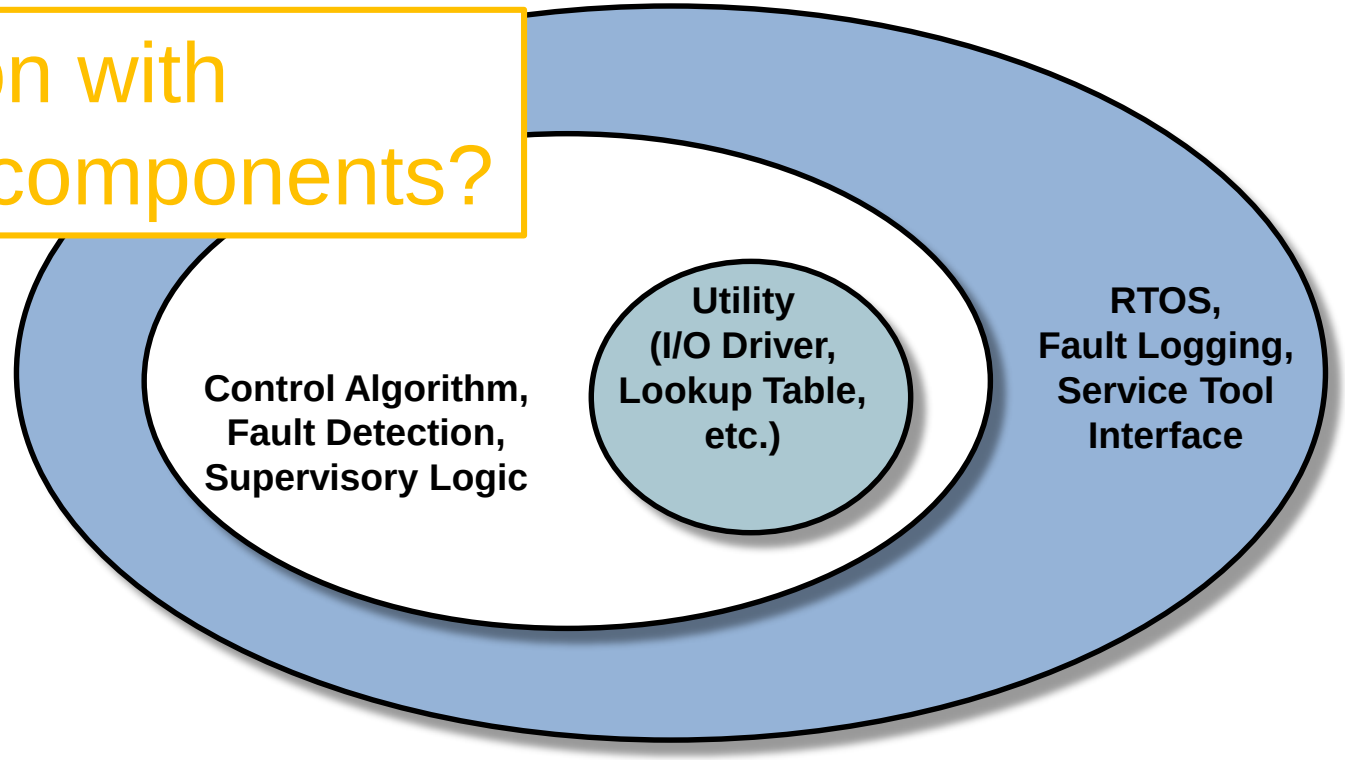
… of customers think automakers and suppliers are responsible for protecting data

…of customers would never buy or wary buying from automakers were hacked

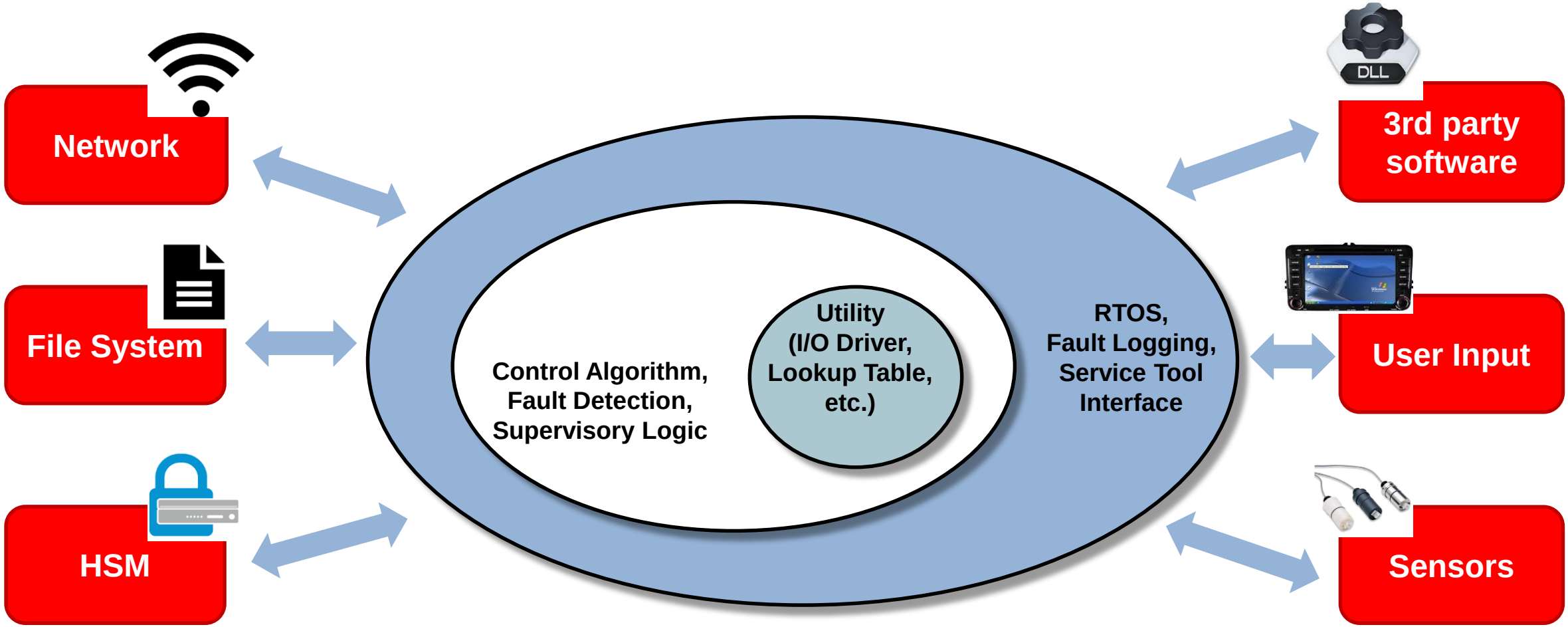
according to 2019 KPMG Consumer Loss Barometer study

Typical Embedded Software Architecture

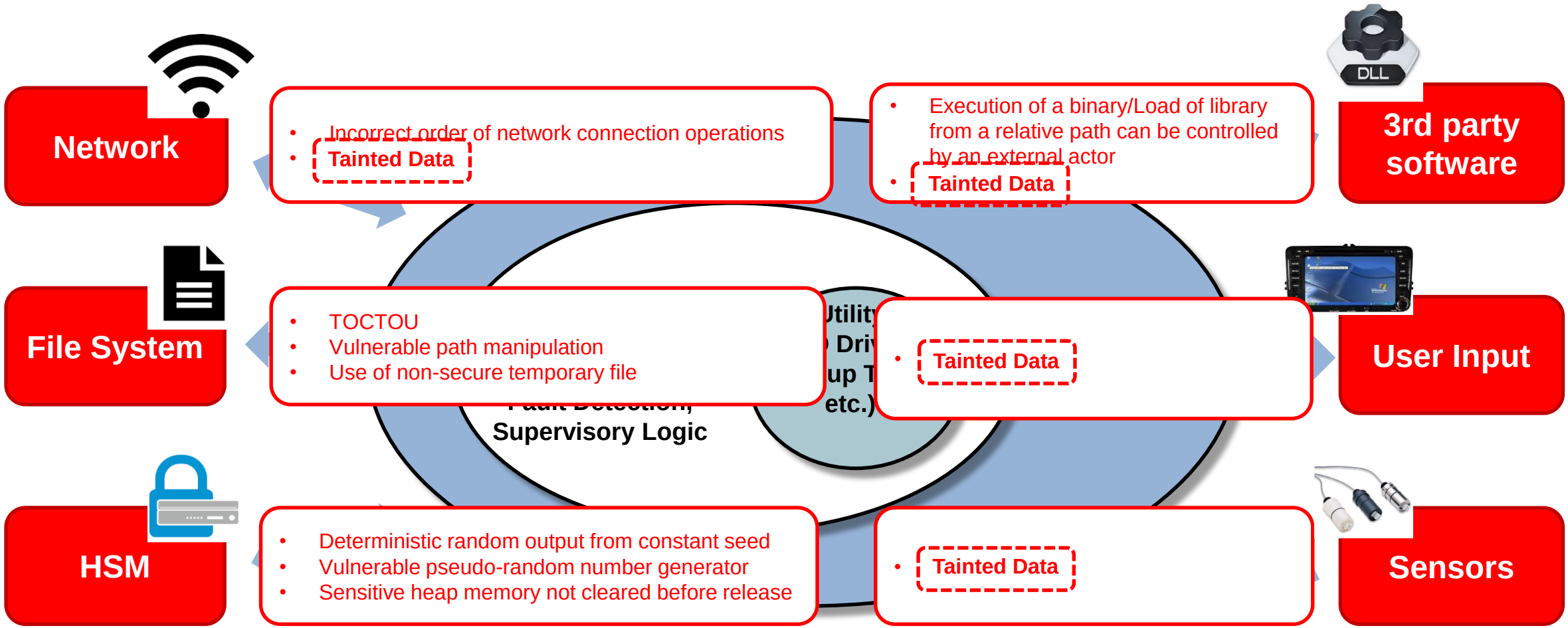
Interaction with external components?



Embedded Software External Interactions



Embedded Software Security Concerns

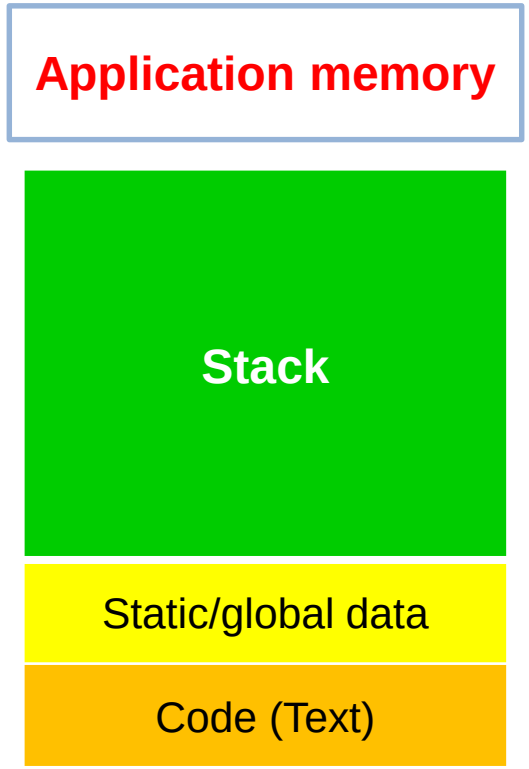


Let's look at an example - Tainted Data

```
1   #include <stdio.h>
2   #define ADCMAXSIZE 256
3
4   typedef signed int sint32;
5   typedef unsigned char uint8;
6
7   extern sint32 getLengthRxData (void);
8   extern sint32 readByte (void);
9
10  void receiveData(void)
11  {
12      sint32 i, length;
13      sint32 ADCdata[ADCMAXSIZE];
14
15      length = getLengthRxData();
16
17      for (i = 0; i < length; i++)
18      {
19          ADCdata[i] = readByte();
20      }
21  }
```

Let's look at an example - Tainted Data

- One of the most exploite **ffer overflow!**




```c
#include <stdio.h>
#define ADCMAXSIZE 256

typedef signed int sint32;
typedef unsigned char uint8;

extern sint32 getLengthRxData (void);
extern sint32 readByte (void);

void receiveData(void)
{
    sint32 i, length;
    sint32 ADCdata[ADCMAXSIZE];

    length = getLengthRxData();

    for (i = 0; i < length; i++)
    {
        ADCdata[i] = readByte();
    }
}
```

Stack is corrupted after array overflow.

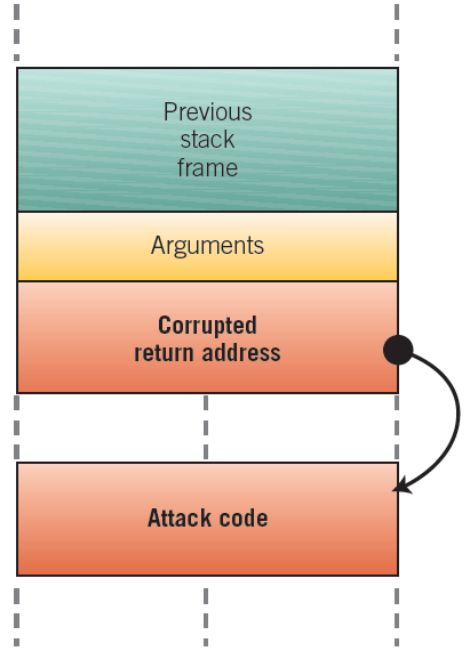
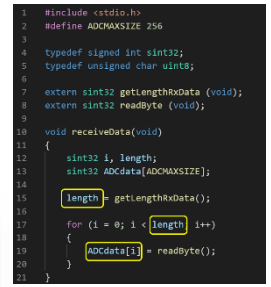


Figure 2

http://www.embedded.com/design/safety-and-security/4304104/Security-fundamentals-for-embedded-software

Polyspace helps you find those vulnerabilities



Polyspace Bug Finder

Source

StackAttack.c

```
1  #include <stdio.h>
2  #define ADCMAXSIZE 256
3  typedef signed int sint32;
4  typedef unsigned char uint8;
5
6  extern sint32 getLengthRxData (void);
7  extern sint32 readByte (void);
8
9  void receiveData(void)
10 {
11     sint32 i, length;
12     sint32 ADCdata[ADCMAXSIZE];
13
14     length = getLengthRxData();
15
16     for (i = 0; i < length;
17     {
18         ADCdata[i] = readByte();
19     }
20 }
```

Select one or more results to review:

- **Loop bounded with tainted value** (Impact: Medium)
- **SEI CERT C MSC21-C** (Recommendation)
- **SEI CERT C INT04-C** (Recommendation)
- **ISO/IEC TS 17961 taintformatio**

Loop bounded with tainted value (Impact: Medium)
Loop is controlled by a value from an unsecure source.
Loop may be infinite.

	Event	File	Scope	Line
1	Return of external getLengthRxData	StackAttack.c	receiveData()	14
2	Assignment to local variable 'length'	StackAttack.c	receiveData()	14
3	Loop bounded with tainted value	StackAttack.c	receiveData()	16

Select one or more results to review:

- **Loop bounded with tainted value** (Impact: Medium)
- **SEI CERT C MSC21-C** (Recommendation)
- **SEI CERT C INT04-C** (Recommendation)
- **ISO/IEC TS 17961 taintformatio**

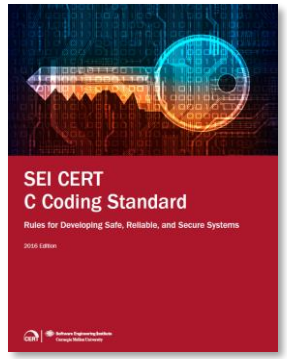
SEI CERT C INT04-C (Recommendation)
Enforce limits on integer values originating from tainted sources
Loop is controlled by a value from an unsecure source.
Loop may be infinite.

	Event	File	Scope	Line
1	Return of external getLengthRxData	StackAttack.c	receiveData()	14
2	Assignment to local variable 'length'	StackAttack.c	receiveData()	14
3	INT04-C Enforce limits on integer	StackAttack.c	receiveData()	16

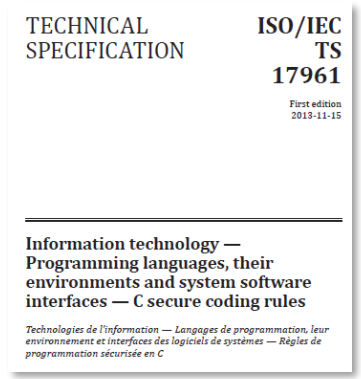
How to Apply Secure Coding Guidelines?

Most Frequently Heard Secure Coding or Security Standards

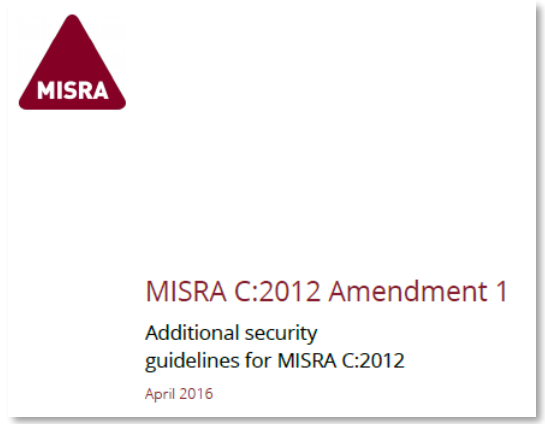
- **CERT C/C++**



- **ISO/IEC 17961**



- **MISRA-C:2012 Amendment 1, Addendum 2/3**



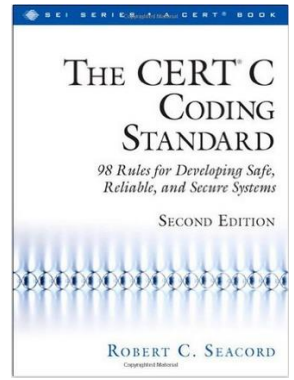
- **CWE** (Common weakness enumeration)

Embedded Safety and Security Coding Standards Overview

Coding Standard	C Standard	Security Standard	Safety Standard	International Standard
MISRA C:2004	C89	No	**Yes**	No
MISRA C:2012	C90/99	**Yes***	**Yes**	No
CERT C99	C99	**Yes**	No	No
CERT C11	C11	**Yes**	**Yes**	No
ISO/IEC TS 17961	C11	**Yes**	No	**Yes**
CWE	None/all	**Yes**	No	No

** Additional security guidelines for MISRA-C:2012 Amendment 1*

Source: Table is based on the book:



How does *Polyspace* help you with embedded software security?

- Detecting security vulnerabilities and underlying defects early
- Provides Exhaustive Documentation and recommendation for security fix
- Proving absence of certain critical vulnerabilities
- Complying with industry standards such as CERT C/C++ and ISO 17961

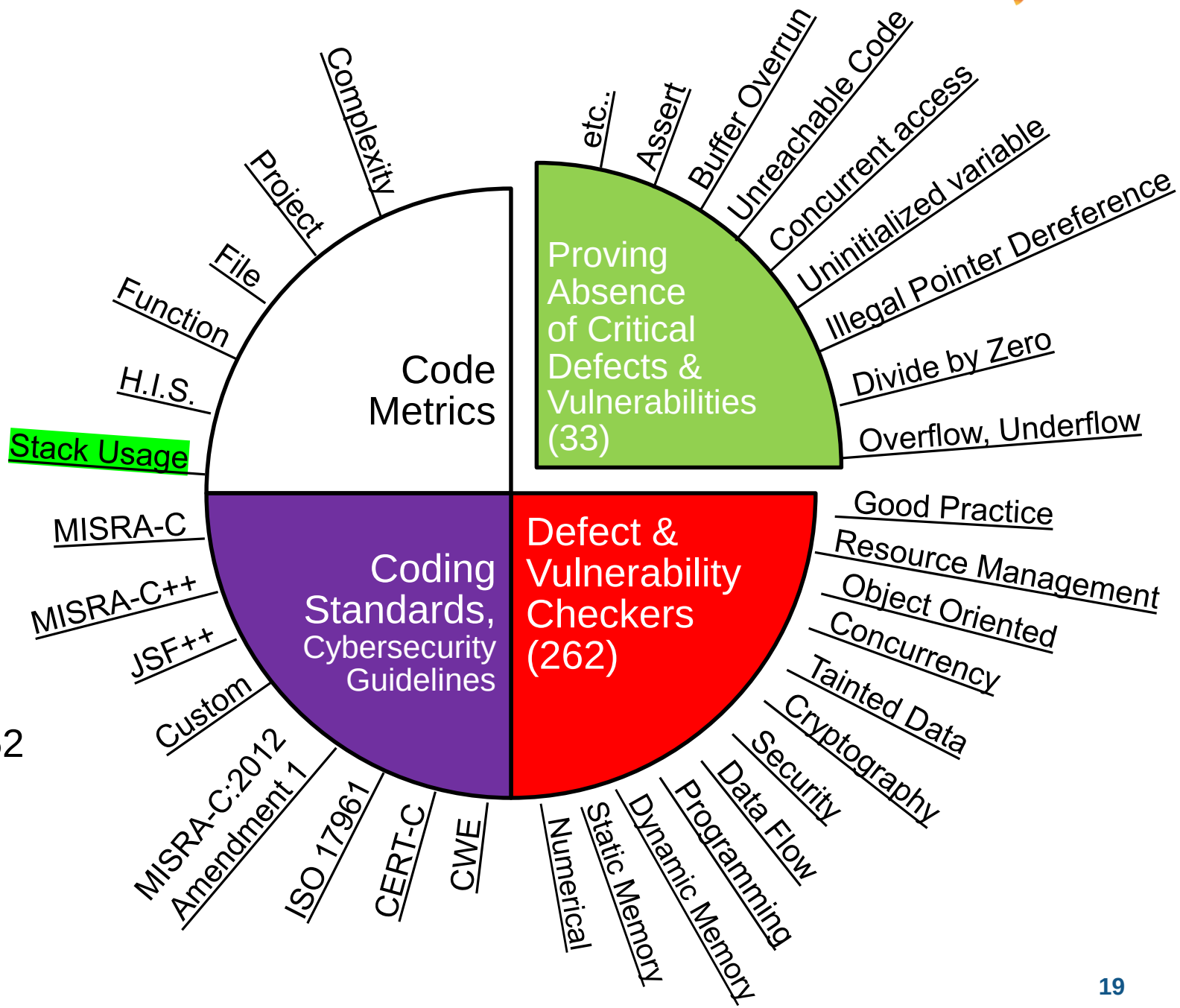
Polyspace Tools

Bug Finder

- Produce code metrics
- Check coding standards
- Find defects and vulnerabilities

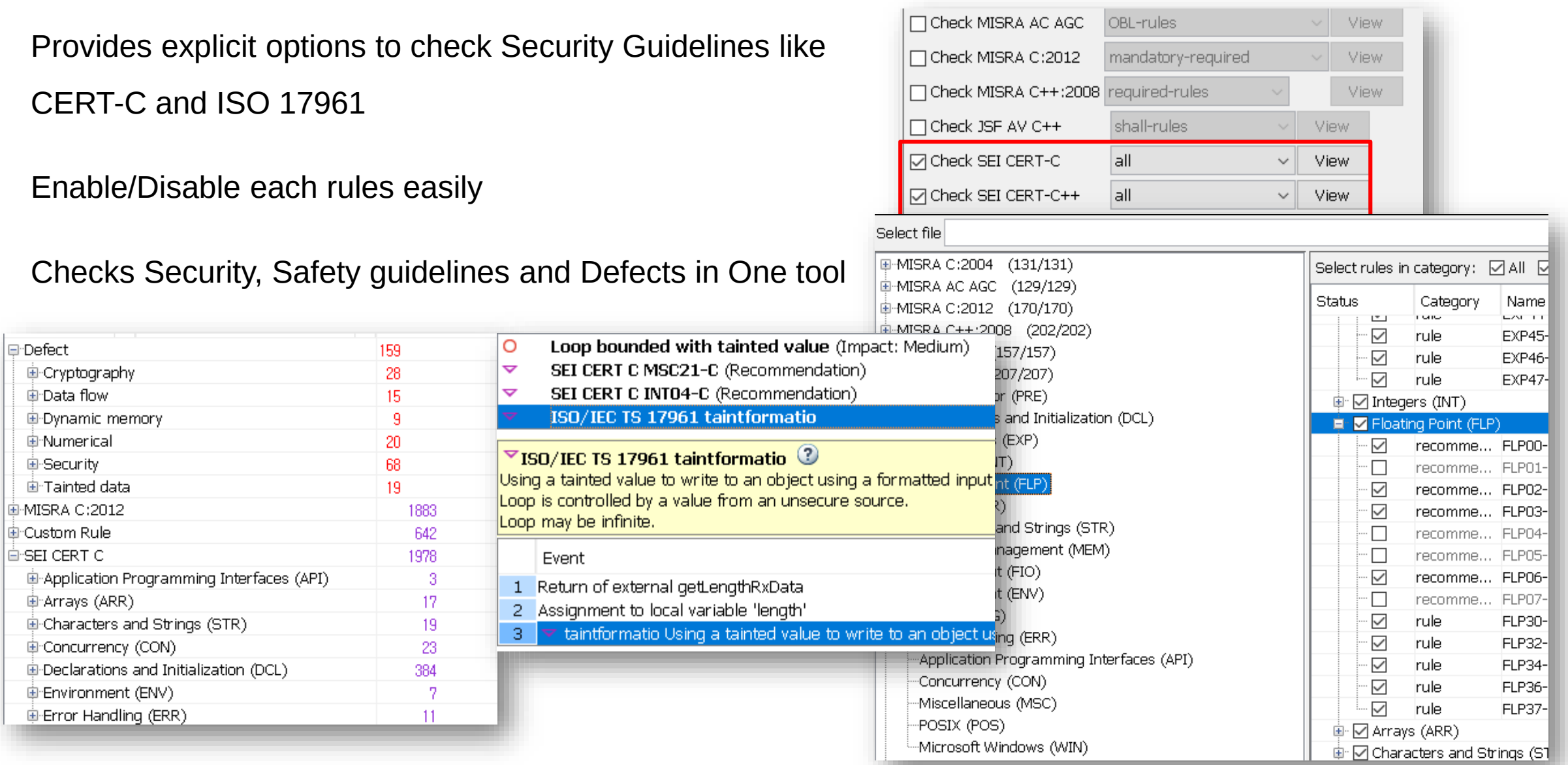
Code Prover

- Proves code Safe and Secure
- 33 most critical run-time checks
- Supports DO-178 and ISO 26262

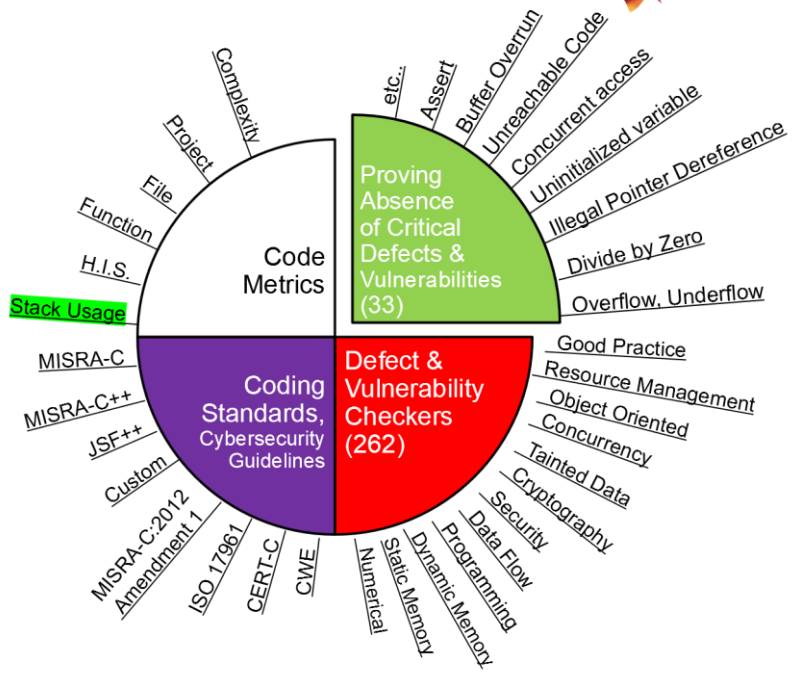


Easy to Configure and Review

- Provides explicit options to check Security Guidelines like CERT-C and ISO 17961
- Enable/Disable each rules easily
- Checks Security, Safety guidelines and Defects in One tool



Important to close all the windows!



- False negatives - missed vulnerabilities
- All malicious attackers want is one loop hole
- Testing is not exhaustive, almost all static analysis tools are not exhaustive
- Polyspace Code Prover – proving absence of specific vulnerabilities
- For critical defects such as buffer overflows, illegal pointer dereferencing ..

Let's look at an example - Tainted Data

```c
#include <stdio.h>
#define ADCMAXSIZE 256

typedef signed int sint32;
typedef unsigned char uint8;

extern sint32 getLengthRxData (void);
extern sint32 readByte (void);

void receiveData(void)
{
    sint32 i, length;
    sint32 ADCdata[ADCMAXSIZE];

    length = getLengthRxData();

    for (i = 0; i < length; i++)
    {
        ADCdata[i] = readByte();
    }
}
```

Polyspace helps you find those vulnerabilities

Polyspace Bug Finder

Source

StackAttack.c

```
1  #include <stdio.h>
2  #define ADCMAXSIZE 256
3  typedef signed int sint32;
4  typedef unsigned char uint8;
5
6  extern sint32 getLengthRxData (void);
7  extern sint32 readByte (void);
8
9  void receiveData(void)
10 {
11     sint32 i, length;
12     sint32 ADCdata[ADCMAXSIZE];
13
14     length = getLengthRxData();
15
16     for (i = 0; i < length;
17     {
18         ADCdata[i] = readByte();
19     }
20 }
```

Polyspace Code Prover

```
9  void receiveData(void)
10 {
11     sint32 i, length;
12     sint32 ADCdata[ADCMAXSIZE];
13
14     length = getLengthRxData();
15
16     for (i = 0; i < length; i++)
17     {
18         ADCdata[i] = readByte();
19     }
20 }
```

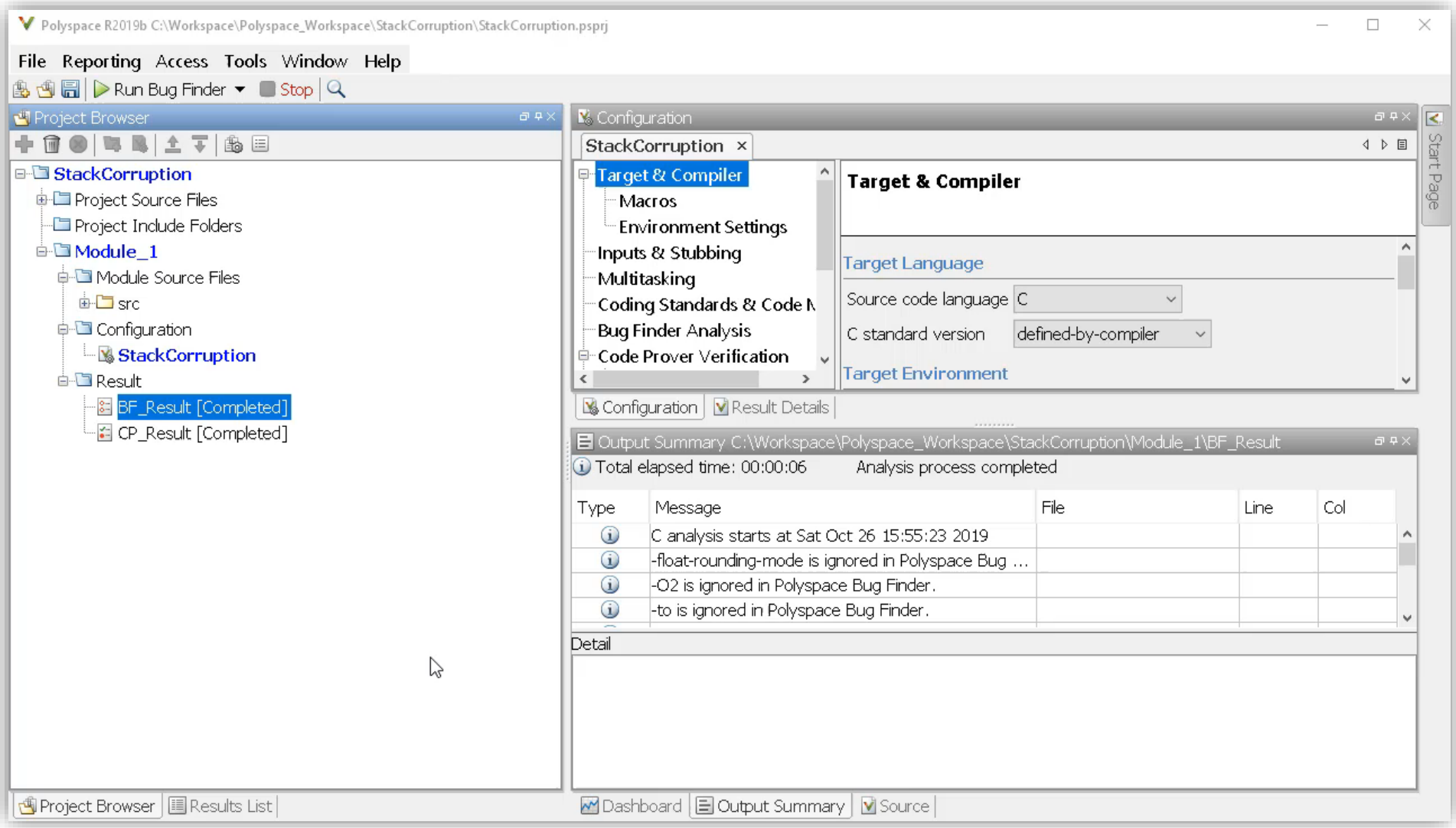
Out of bounds array index

Warning: array index may be outside bounds : [0..255]
array size: 256
array index value: [0 .. 256]

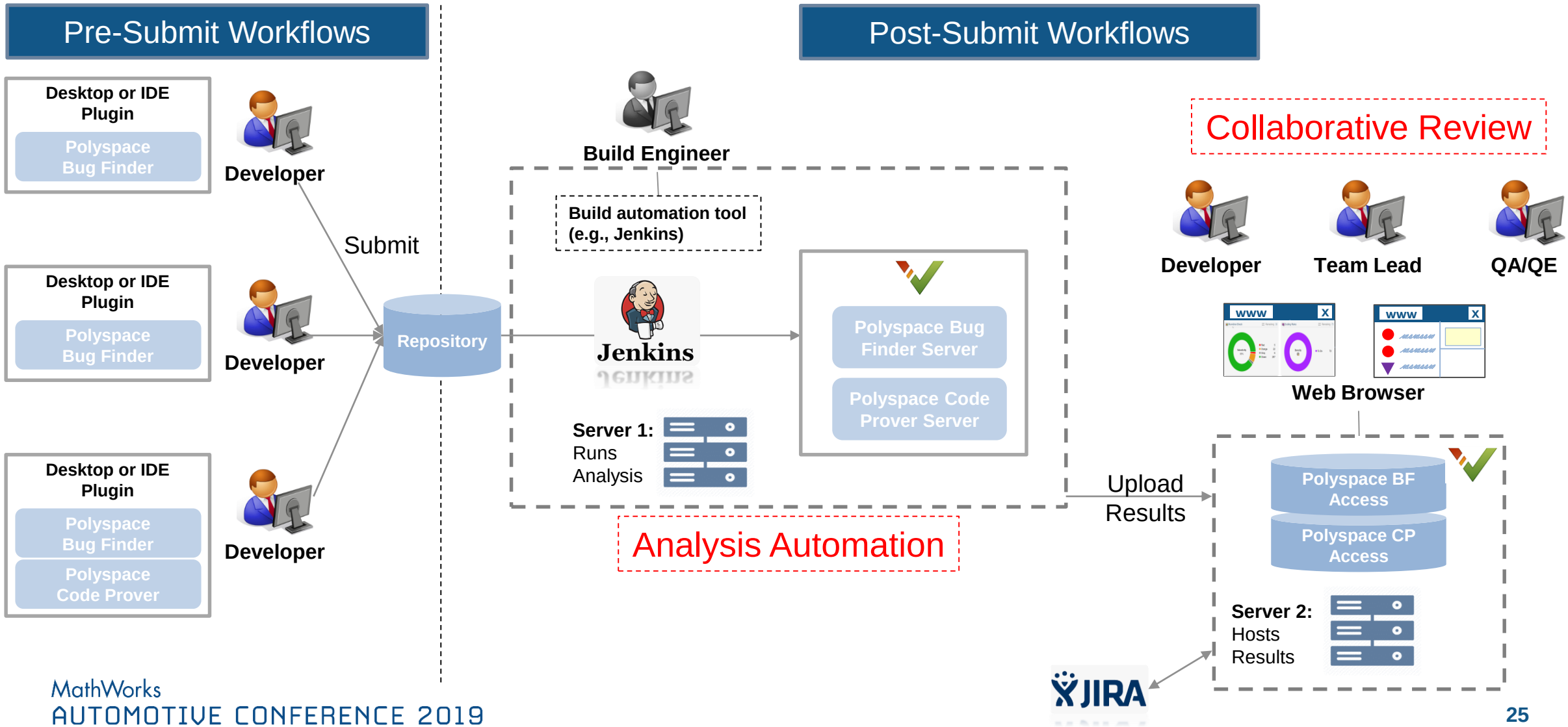
Polyspace helps you find those vulnerabilities

Polyspace
Bug Finder

Polyspace
Code Prover

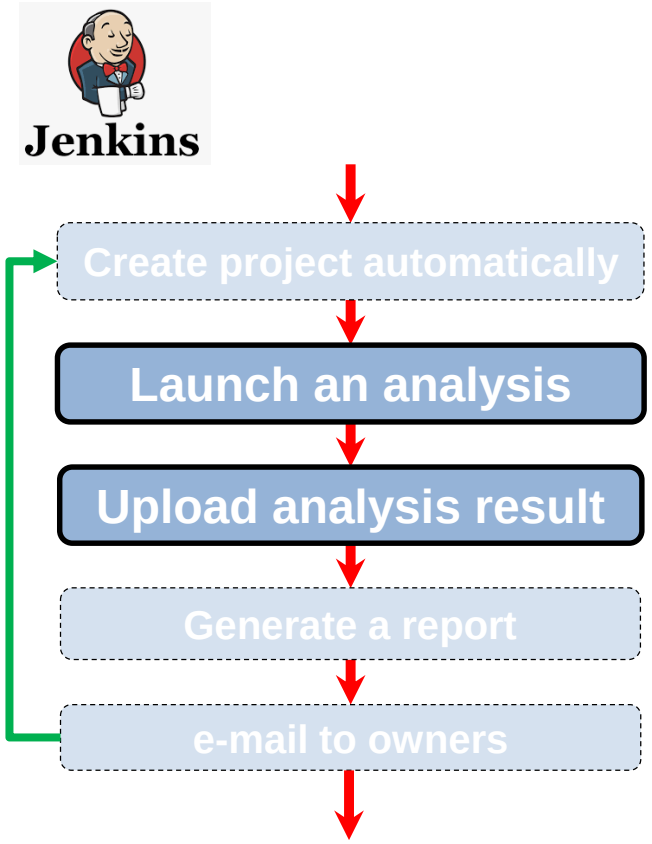


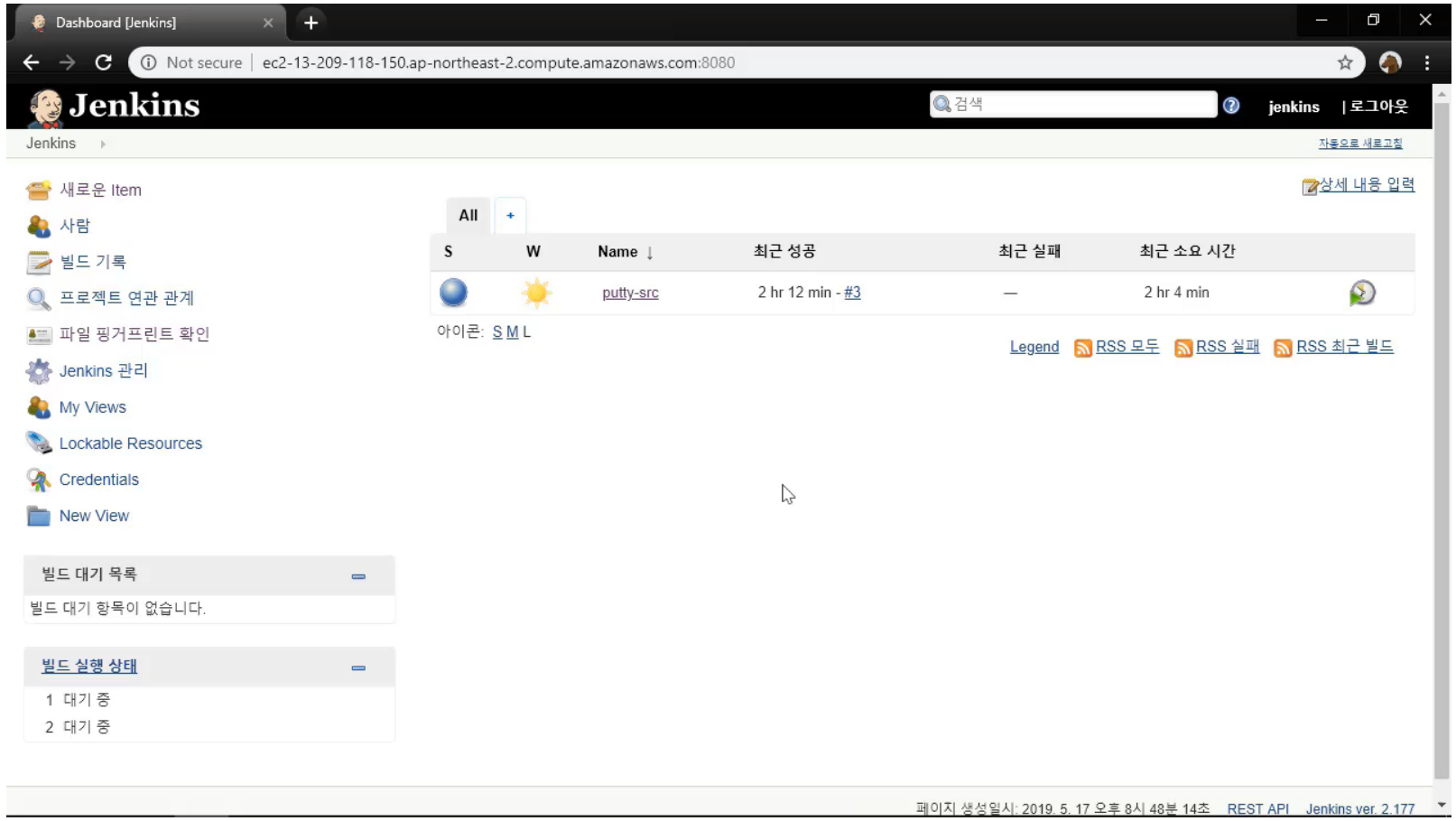
Recommended Workflow



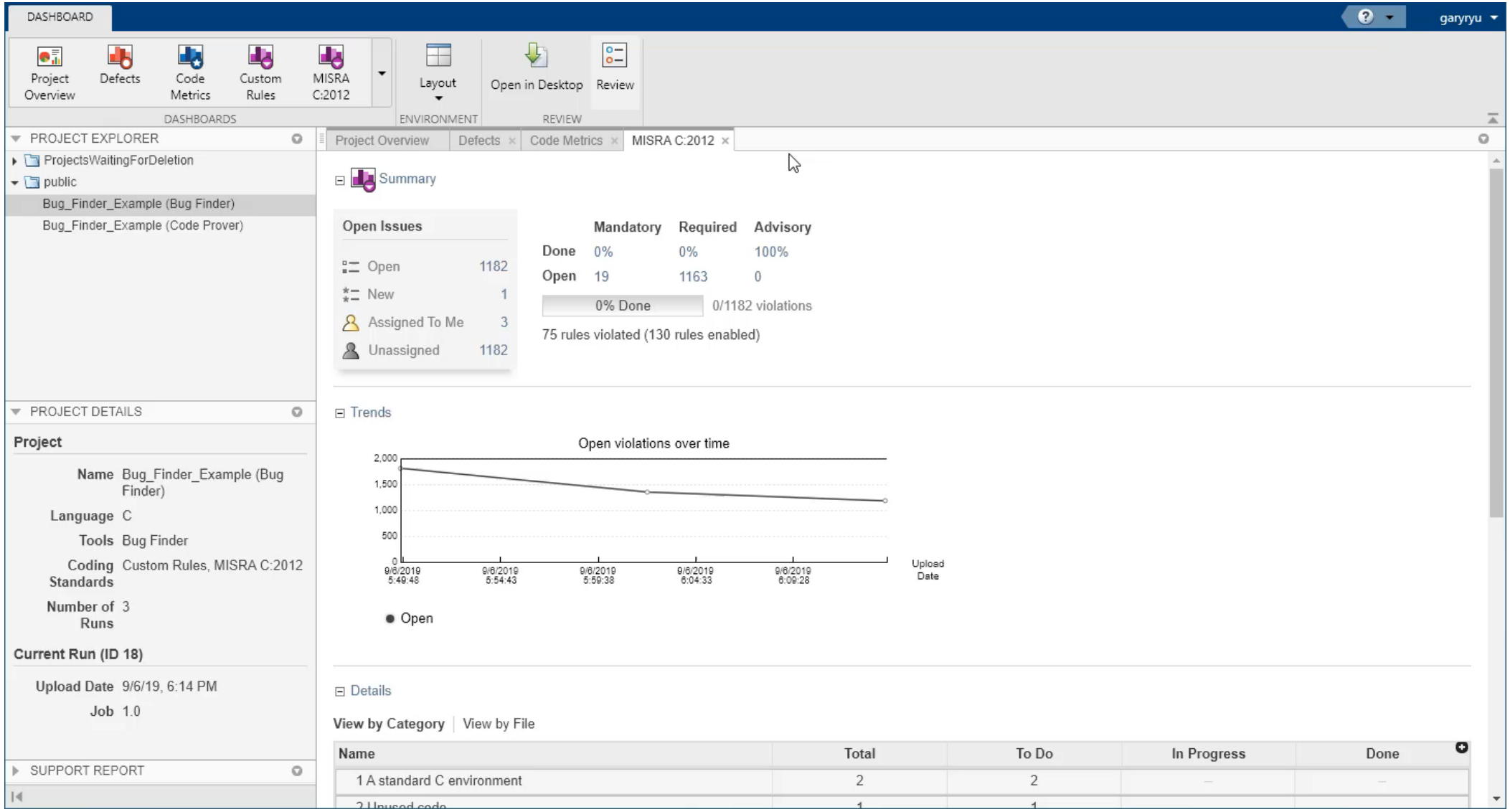
Automate Polyspace Analysis by Jenkins plug-in

- Set env variables for Polyspace Access and Web Metrics
- Ease analysis automation and configuration for a standard CI workflow

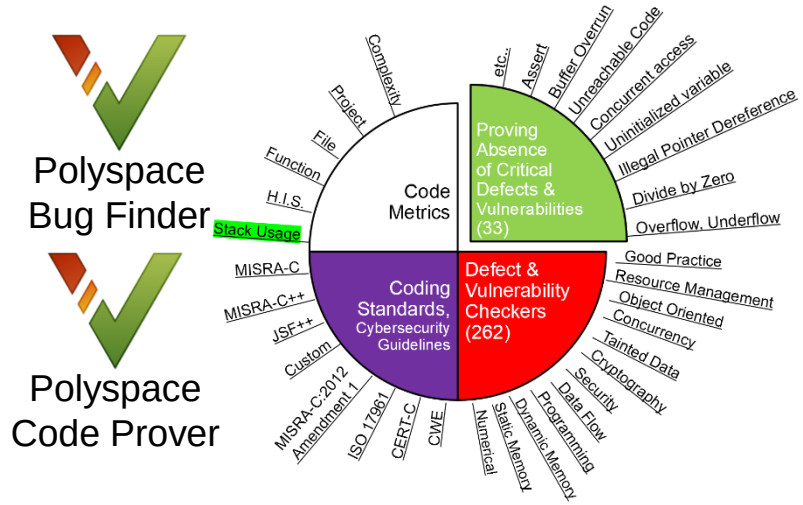


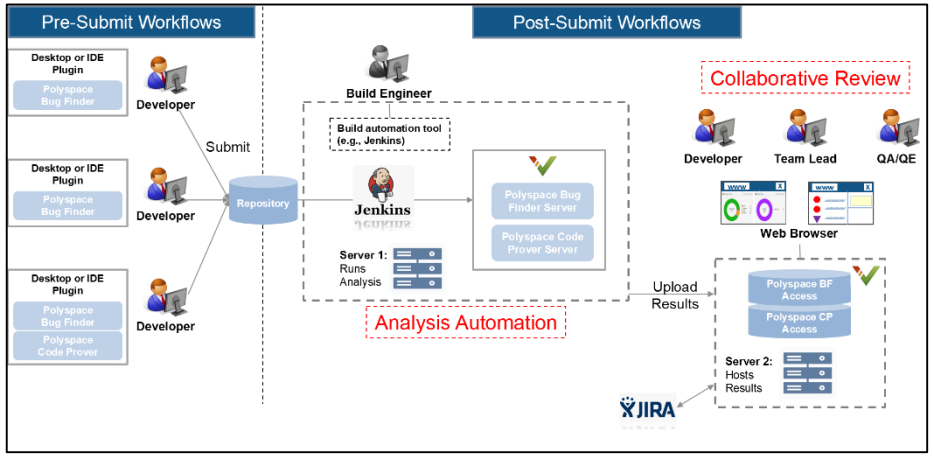


Polyspace Access - Review Results in Collaborative Environment



Key takeaways





1.

Achieve the Goals for Security and Safety by One Tool, Polyspace

2.

Improve your workflow by Analysis Automation & Collaborative Review Environment

Thank you!