Security guarantees for the execution infrastructure of software applications

Frank Piessens, Dominique Devriese, Jan Tobias Muehlberg, Raoul Strackx

iMinds-DistriNet, KU Leuven

IEEE SecDev 2016

Introduction

- Consider Heartbleed
  - Buffer over-read in OpenSSL
  - Leaving a vast amount of web applications vulnerable
  - Who is to blame?
    - OpenSSL developers for the buffer over-read bug itself?
    - Apache (or other web servers) developers for using OpenSSL?
    - The compiler for not checking array bounds automatically?
    - The OS/HW for storing network buffers in the same protection domain as crypto keys?
    - …
- Application security benefits from adequate security countermeasures in the infrastructure

Introduction

- Software applications rely on *infrastructure*
  - = collection of HW/SW required to execute that application
  - I.e. Compilers, operating systems, virtual machines, databases, network protocol stacks, API implementations, …
- A significant fraction of attacks against SW applications rely on, or exploit implementation details of the underlying infrastructure
  - Exploitation of memory safety errors
  - Network sniffing or man-in-the-middle
  - Memory scanning malware
  - Fingerprinting
  - …
- It is NOT the case that more defenses are always better!
  - Depends on the threat model – e.g. memory safety
  - Cost/benefit issues

Introduction

- Key question: what security guarantees should the infrastructure offer?
  - Under what attacker model?
  - And at what cost?
- Once we understand the answer to this question, we can develop a systematic approach to:
  - Hardening existing infrastructure
  - Designing new infrastructure in a better way
    - This is where the IoT might present great opportunities!

DistriNet iMinds KU LEUVEN

Defining infrastructure security

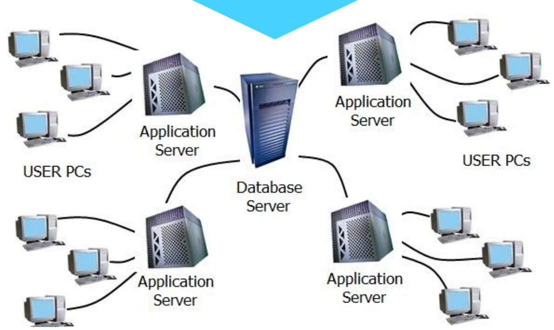
Source code

```
public static final Parcelable.Creator<FragmentState> CREATOR
        = new Parcelable.Creator<FragmentState>() {
    public FragmentState createFromParcel(Parcel in) {
        return new FragmentState(in);
    }

    public FragmentState[] newArray(int size) {
        return new FragmentState[size];
    }
};
```

Compilation and deployment

Runtime system



Defining infrastructure security

Source code

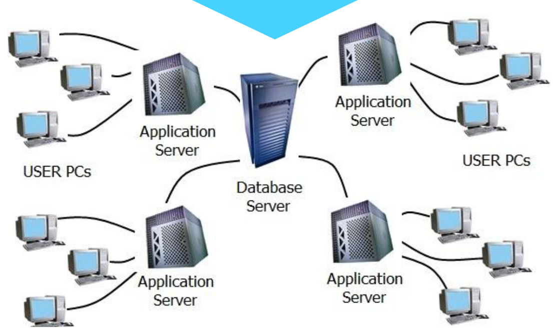
```
public static final Parcelable.Creator<FragmentState> CREATOR
        = new Parcelable.Creator<FragmentState>() {
    public FragmentState createFromParcel(Parcel in) {
        return new FragmentState(in);
    }

    public FragmentState[] newArray(int size) {
        return new FragmentState[size];
    }
};
```

This is the abstraction level at which the application is developed / analyzed / debugged

Compilation and deployment

Runtime system



DistriNet iMinds KU LEUVEN

Defining infrastructure security

Source code

```
public static final Parcelable.Creator<FragmentState> CREATOR
        = new Parcelable.Creator<FragmentState>() {
    public FragmentState createFromParcel(Parcel in) {
        return new FragmentState(in);
    }

    public FragmentState[] newArray(int size) {
        return new FragmentState[size];
    }
};
```

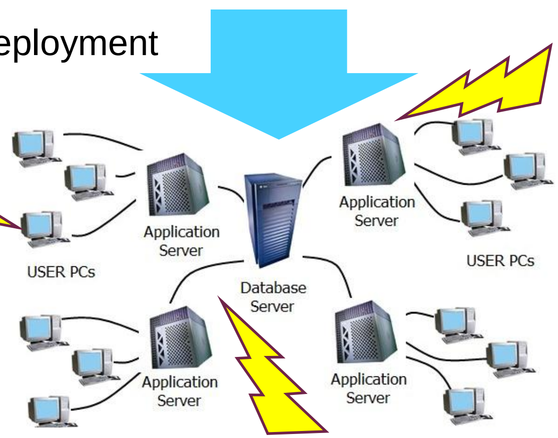
This is the abstraction level at which the application is developed / analyzed / debugged

Compilation and deployment

code injection attacks

tampering attacks

Runtime system



network attacks

This is the abstraction level at which the system is attacked

Defining infrastructure security

Source code

```
public static final Parcelable.Creator<FragmentState> CREATOR
        = new Parcelable.Creator<FragmentState>() {
    public FragmentState createFromParcel(Parcel in) {
        return new FragmentState(in);
    }

    public FragmentState[] newArray(int size) {
        return new FragmentState[size];
    }
};
```

This is the abstraction level at which the application is developed / analyzed / debugged

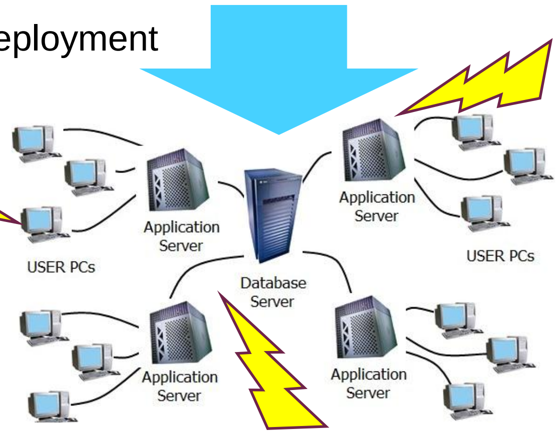
The "developer" view: infrastructure executes my source code and implements the APIs I use

Compilation and deployment

tampering attacks

code injection attacks

Runtime system



network attacks

This is the abstraction level at which the system is attacked

The "attacker" view of the system

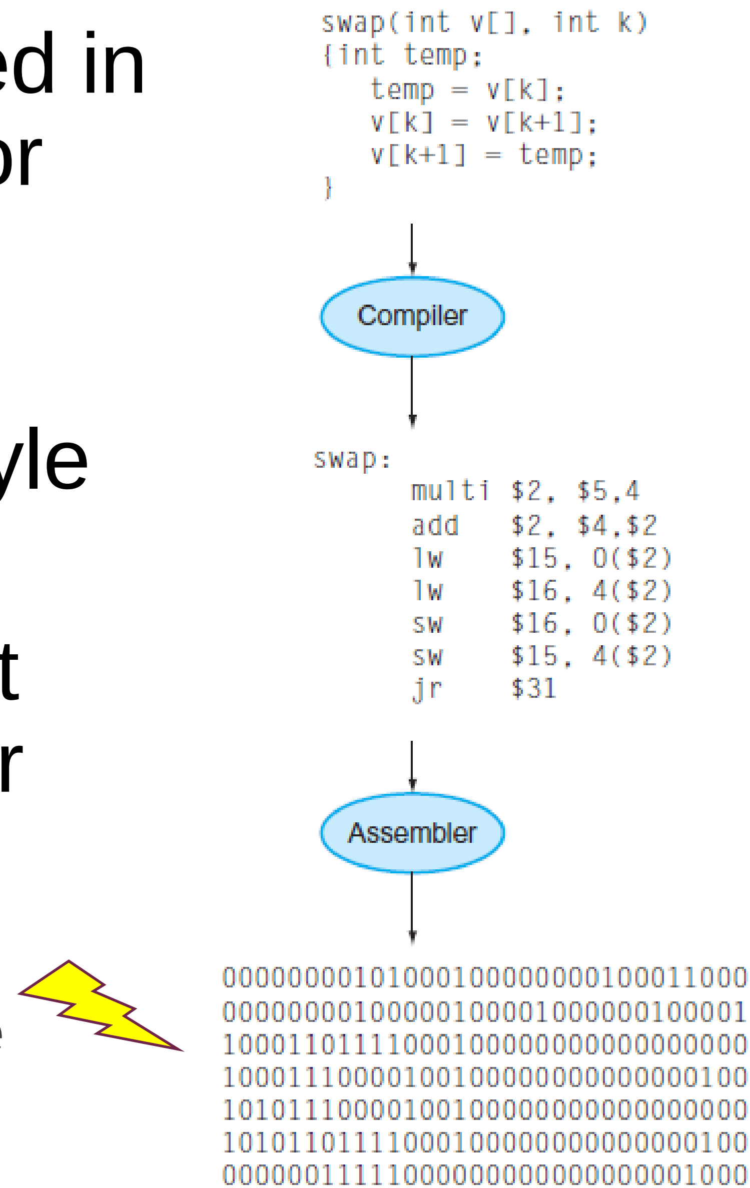
Defining infrastructure security

- Infrastructure should securely “bridge” these two levels
  - I.e. the infrastructure should implement the developer view “securely”
- Two fundamental properties
  - **Safety** of the source level language (Milner)
    - “No undefined behavior”
  - **Full abstraction** of the translation to the target (Abadi)
    - “No additional attacks because of the translation”
    - “Principle of source-based reasoning” (Gordon)

DistriNet iMinds KU LEUVEN

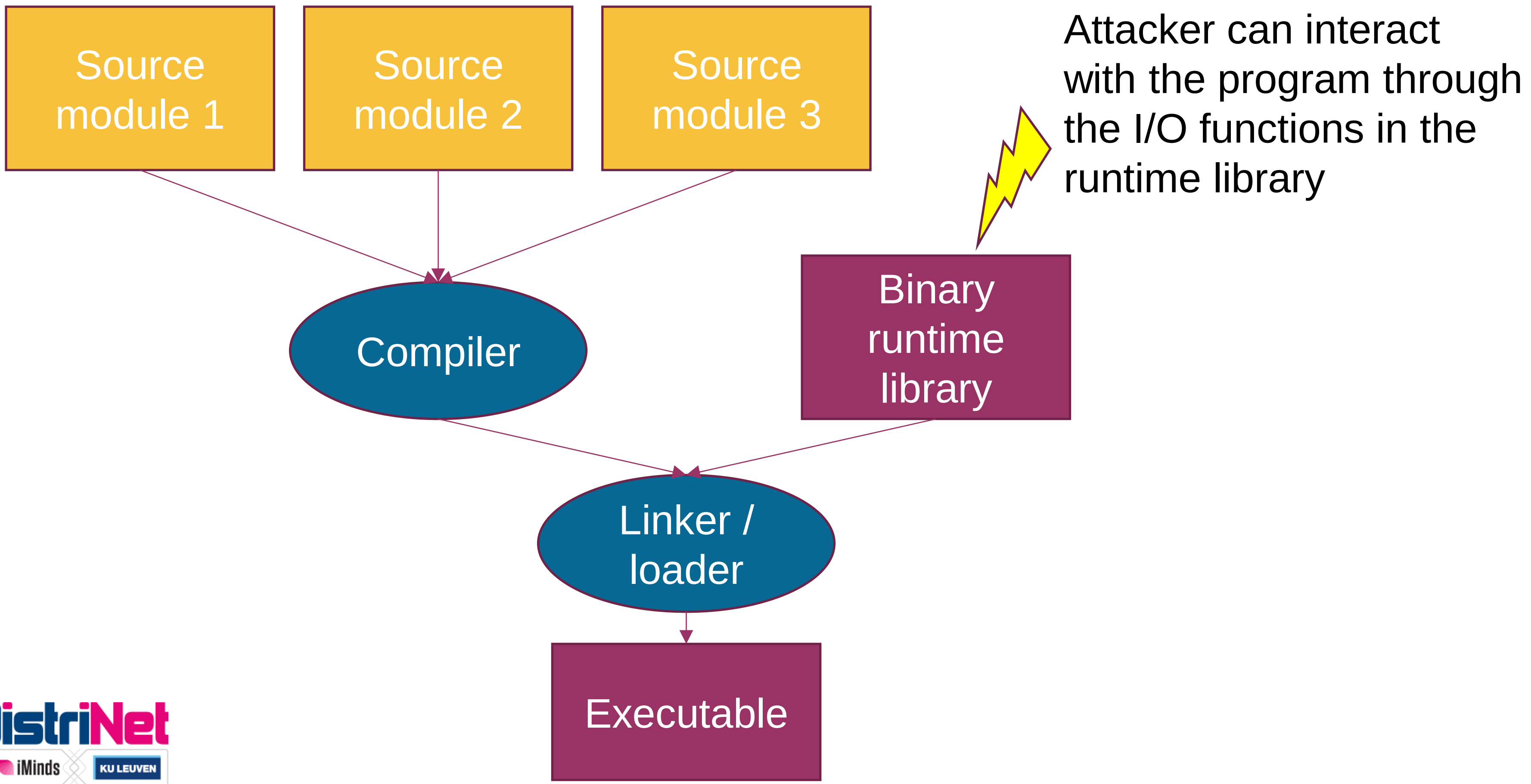
Example: secure compilation to machine code

- Applications are developed in source languages like C or Java
- But executed as machine code on von Neumann style micro-processors
- And attacks often occur at layers of abstraction lower than source code
  - E.g. code injection attacks
  - E.g. memory scanning malware



Safety

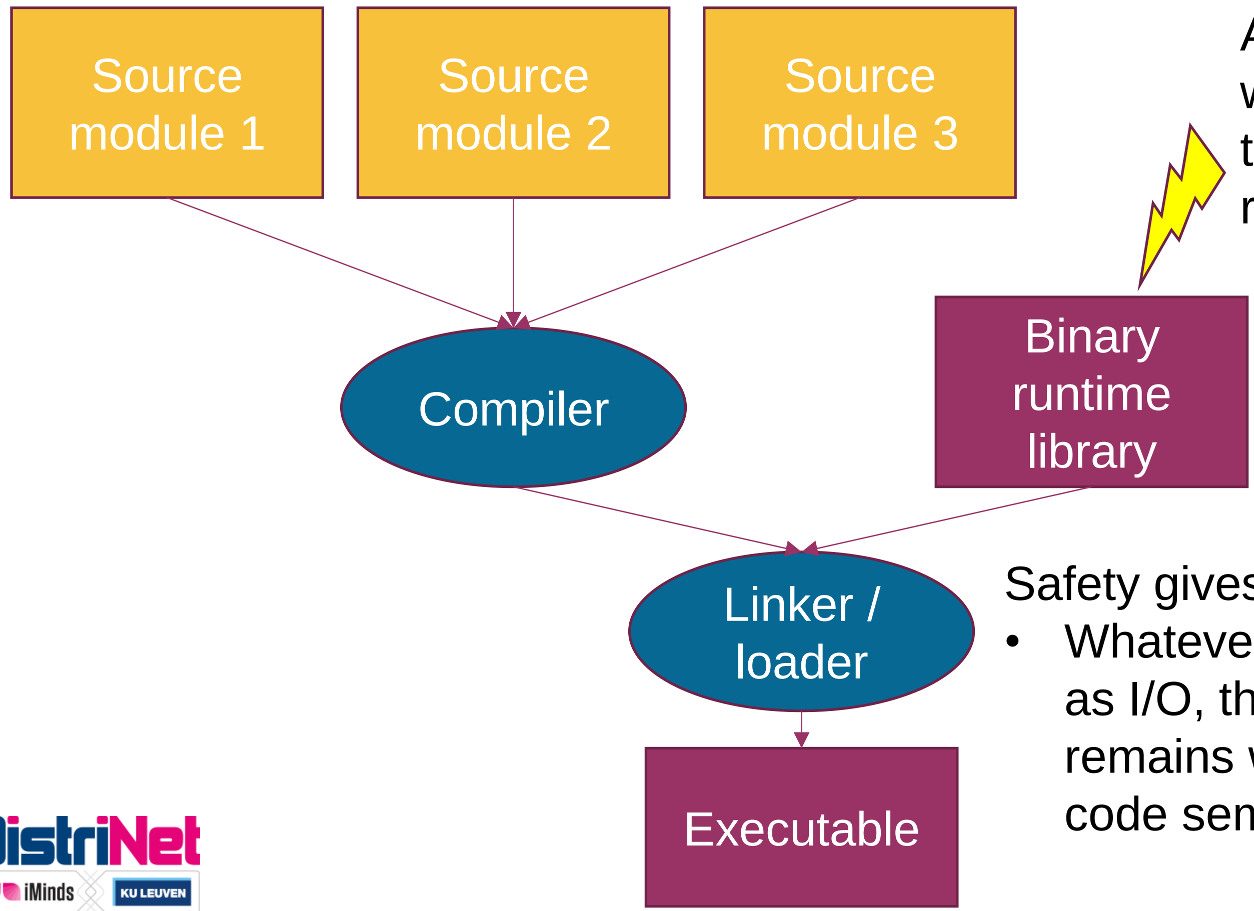
- Safe languages (or safe compilers for unsafe languages)



DistriNet iMinds KU LEUVEN

Safety

- Safe languages (or safe compilers for unsafe languages)



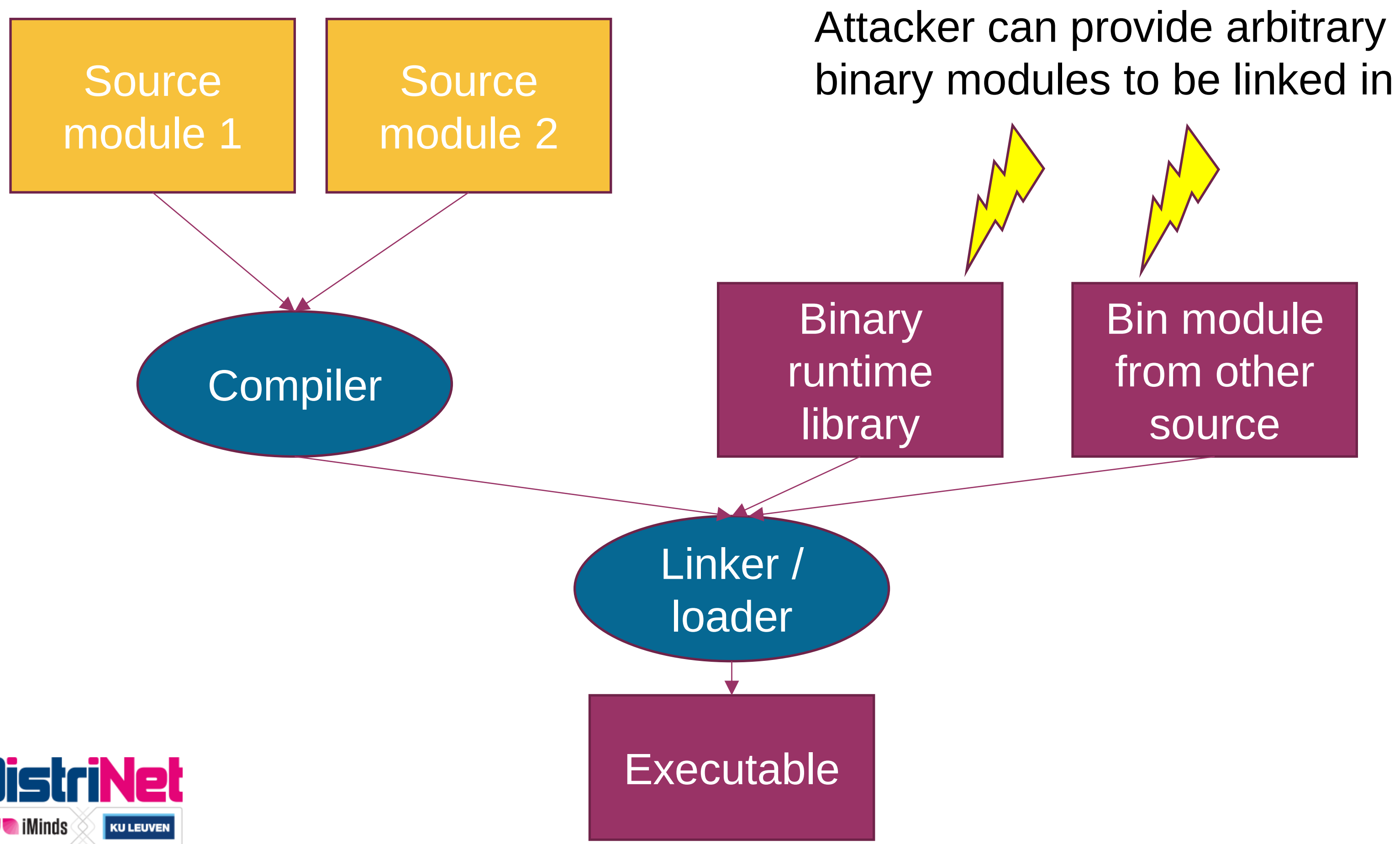
Attacker can interact with the program through the I/O functions in the runtime library

Safety gives us:

- Whatever the attacker provides as I/O, the behavior of the program remains well-defined by the source code semantics

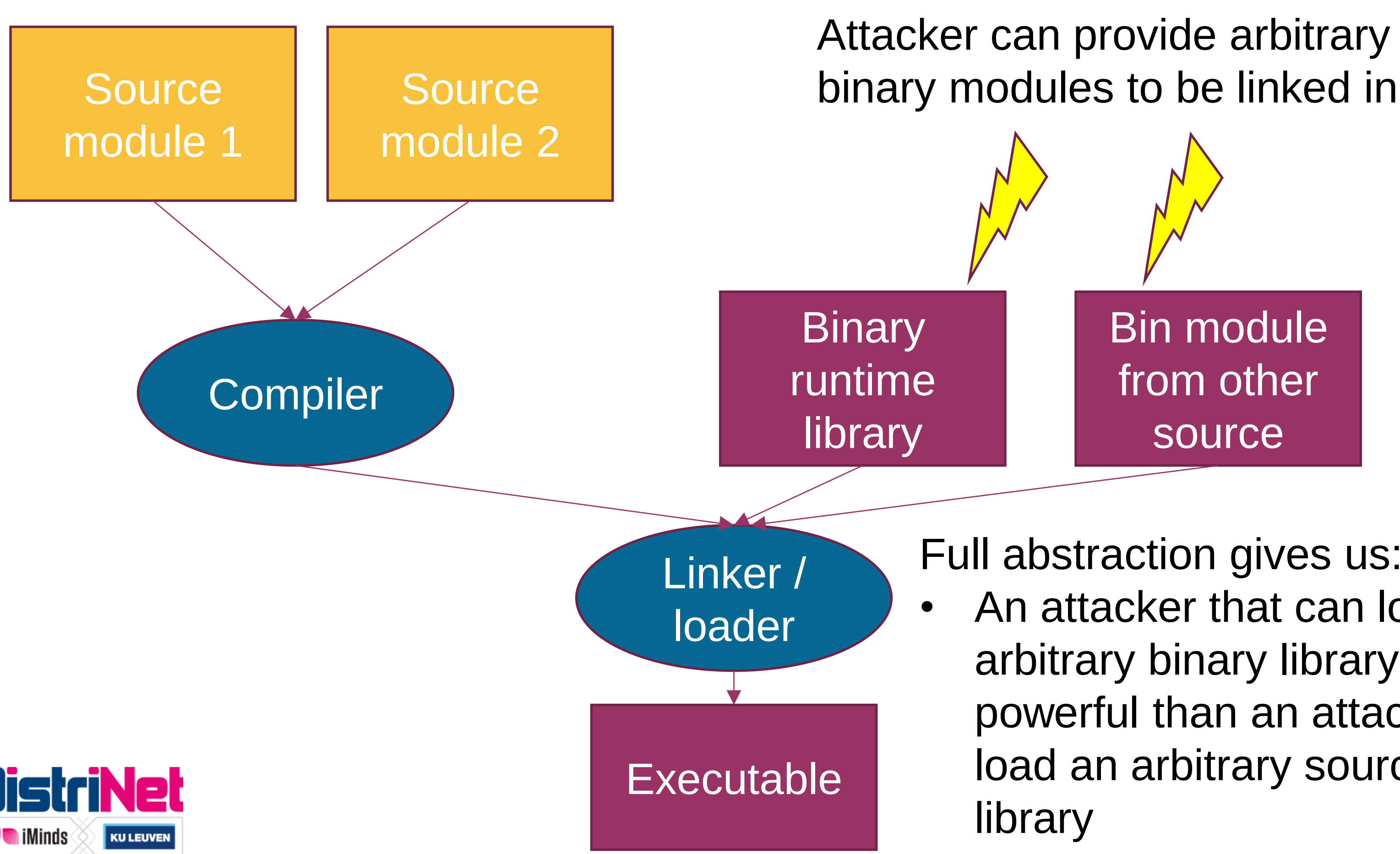
Full abstraction

- Fully abstract compilation:



Full abstraction

- Fully abstract compilation:



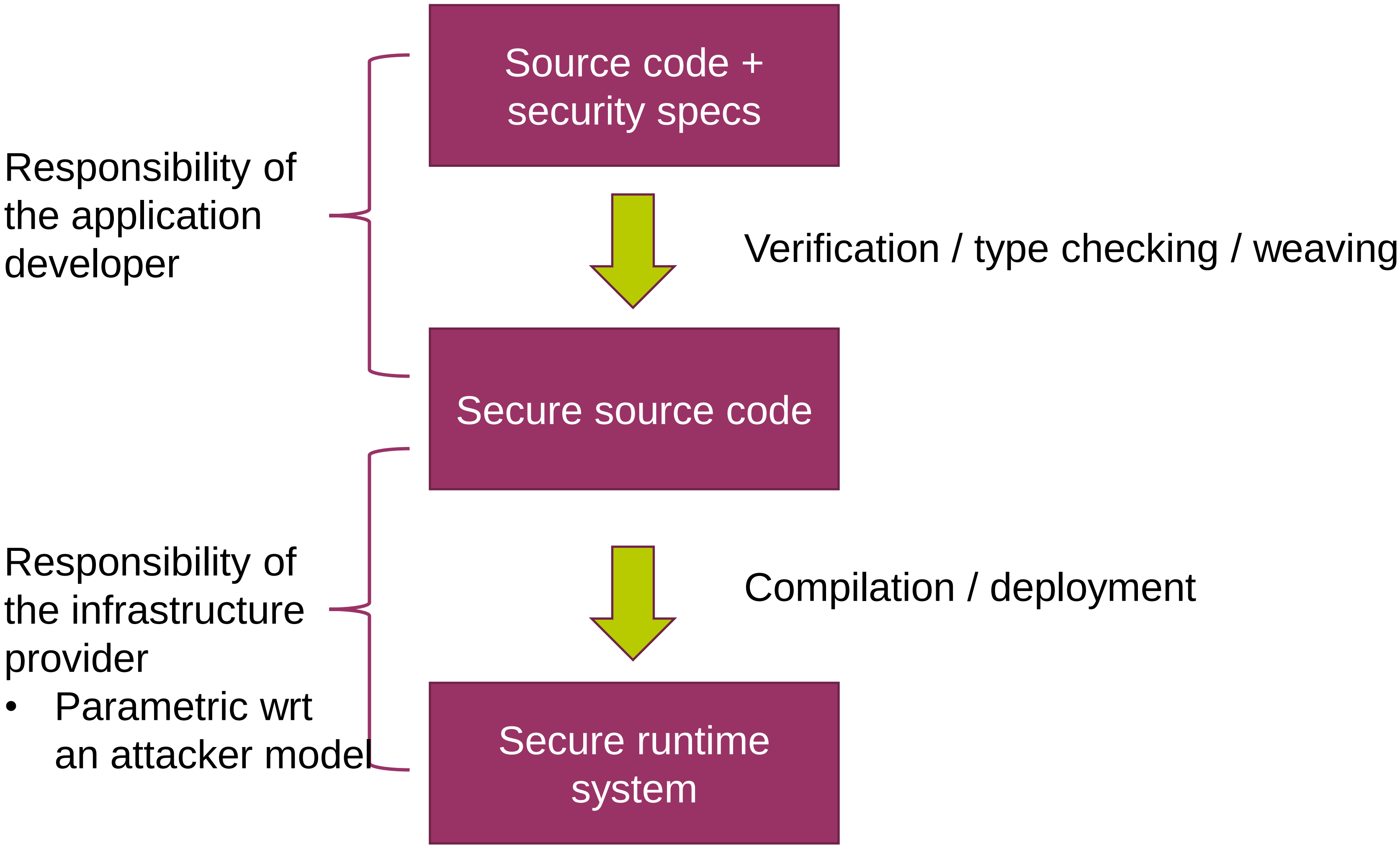
Full abstraction gives us:

- An attacker that can load an arbitrary binary library is no more powerful than an attacker that can load an arbitrary source code library

Conclusions

- ICT security is a cross-layer concern
- It is useful to have a clear division of responsibilities between
  - The infrastructure provider
  - The application developer
- Such a division can lead to precise security requirements for the infrastructure provider
  - Safety: “integrity of execution”
  - Full abstraction: “integrity and confidentiality of execution”
- One man’s infrastructure is another man’s application

Summary



DistriNet
iMinds KU LEUVEN