# Assembling secure OS images

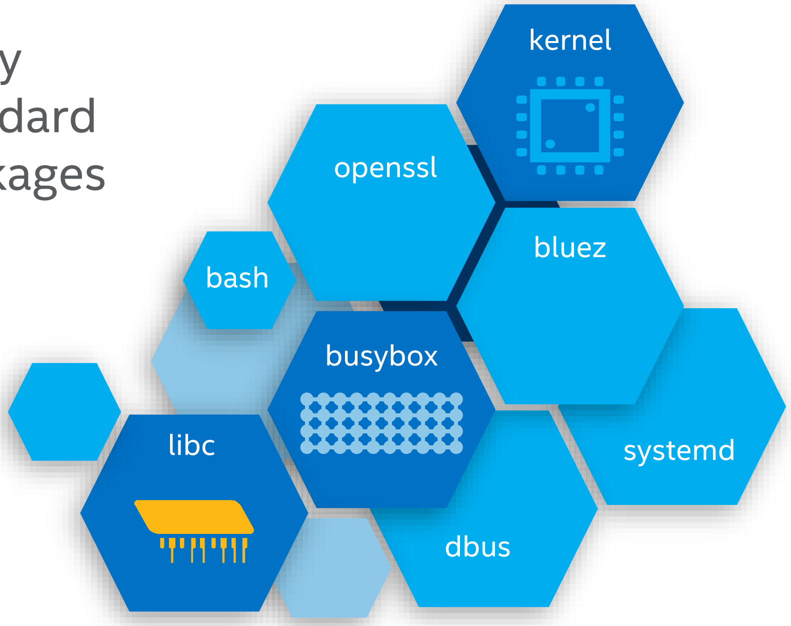
Elena Reshetova, Intel Open Source Technology Center

Linux Security Summit, August, 20, 2015

# Motivation

## Modern Linux-based OS image

Many standard packages



Configuration scripts

## OS image producers

- Companies, big, small and tiny
  - Especially true in embedded world

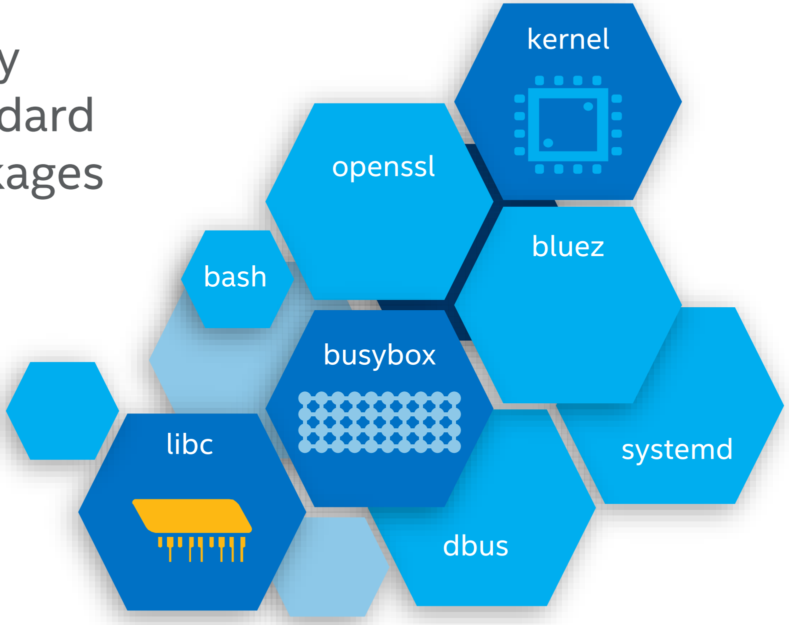
## Tools

- Automated build systems
  - Proprietary & Public

# Motivation

**Modern Linux-based OS image**

Many standard packages



Configuration scripts

**What can we say about OS security without manual or run-time analysis**

# System requirements

## Functional

Asset/Analyze security during various stages of build process

Provide informative & prioritized issue report

Extensible architecture supporting independent plugins

## Non-functional

Build system agnostic and easily integratable

Reasonable performance impact

## Nice to have

Work on image diffs

Suggest fixes/hardening options

# Basic analysis

General

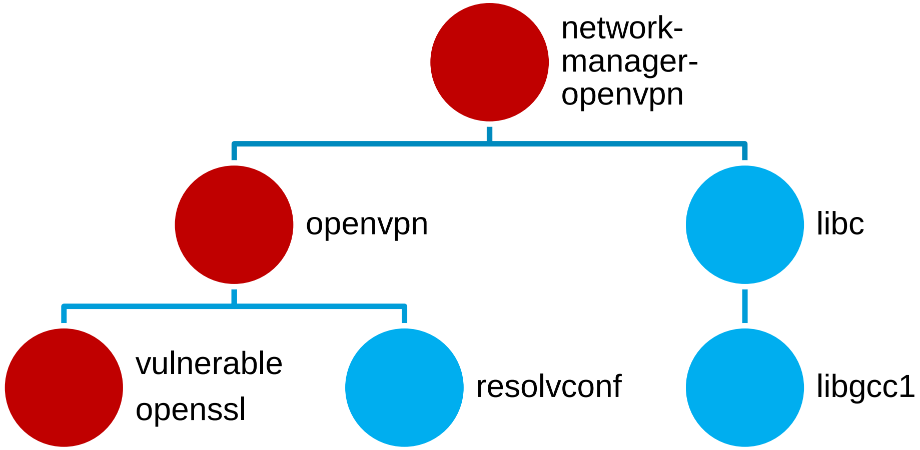
- Kernel config settings
- Filesystem permissions
- Filesystem mount options
- Security-related compile flags
- Log and auidit settings
- ....

Per package

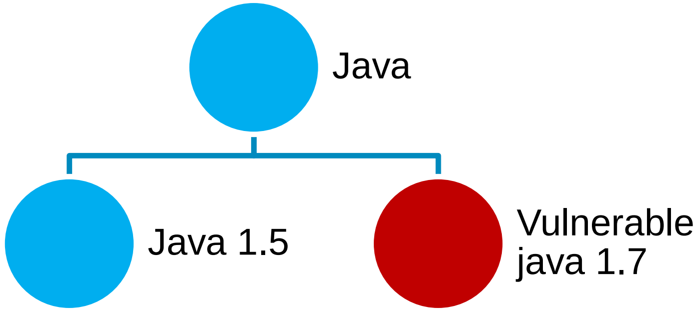
- Presense or absence
- Known unsecure legacy services
- CVEs
- Package-specific configurations and settings
- ….

# Dependencies analysis

## Show potentially affected areas in the stack

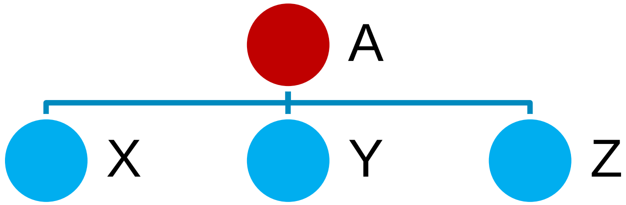


## Suggest more secure alternatives

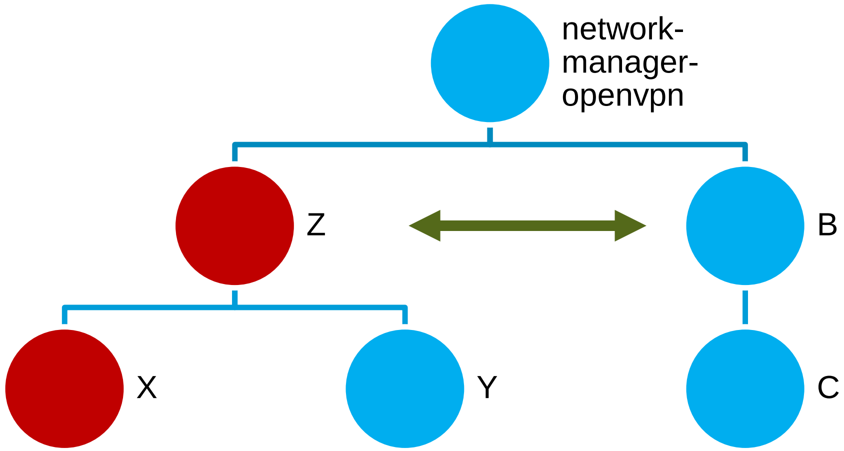


# Potential analysis

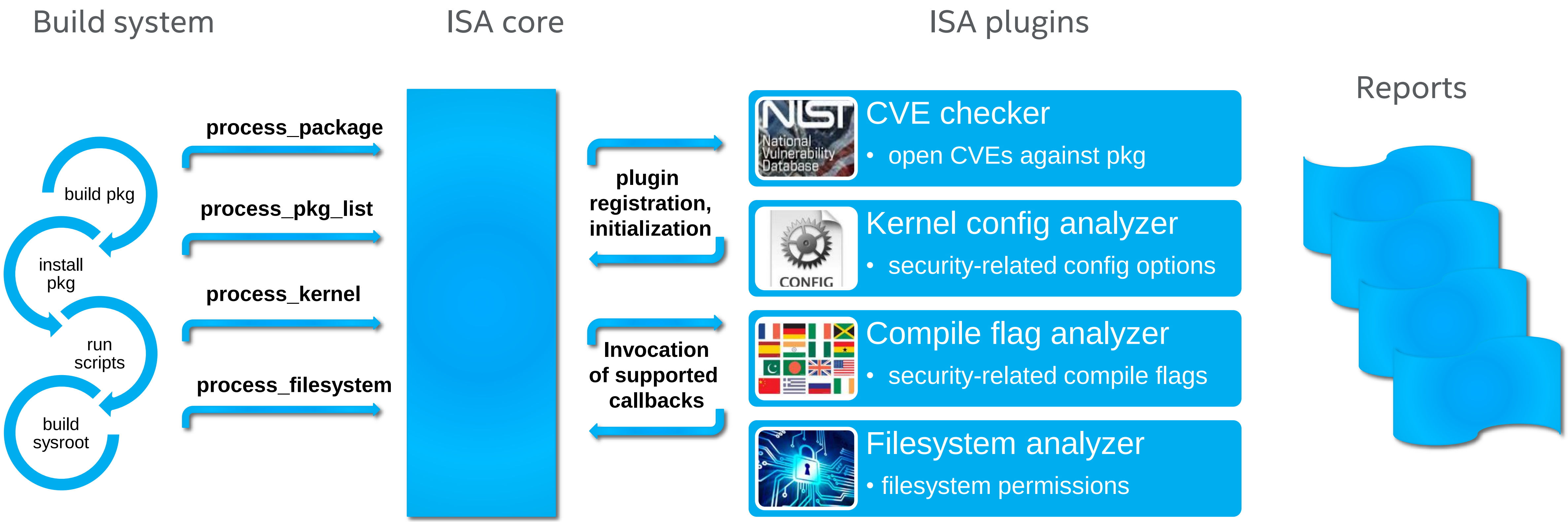
Can a set of "ok" packages lead towards a less secure system?



Can one package cancel the bad effect of less secure package?



# Architecture



Build system

ISA core

ISA plugins

Reports

build pkg

install pkg

run scripts

build sysroot

**process_package**

**process_pkg_list**

**process_kernel**

**process_filesystem**

**plugin registration, initialization**

**Invocation of supported callbacks**

CVE checker
- open CVEs against pkg

Kernel config analyzer
- security-related config options

Compile flag analyzer
- security-related compile flags

Filesystem analyzer
- filesystem permissions

# Implementation & Build System integration

- Prototype implementation in Python

  https://github.com/otcshare/isafw

- Integrated into a Yocto layer as a .bbclass
  - Checks packages, kernel config and filesystem

- Coming very soon: Open Embeeded layer

# Discussion

- Do you see a value in the proposed concept/tool?
- Would you be interested for the project to cover particular things?
- Do you want to see integration to different build system?
- What are the things to do differently?
- What is the general direction to develop this further?

https://github.com/otcshare/isafw