SOFTWARE ASSURANCE MARKETPLACE

A NATIONAL CYBERSECURITY RESOURCE

The Tool Developer and the SWAMP

James A. Kupsch

kupsch@cs.wisc.edu

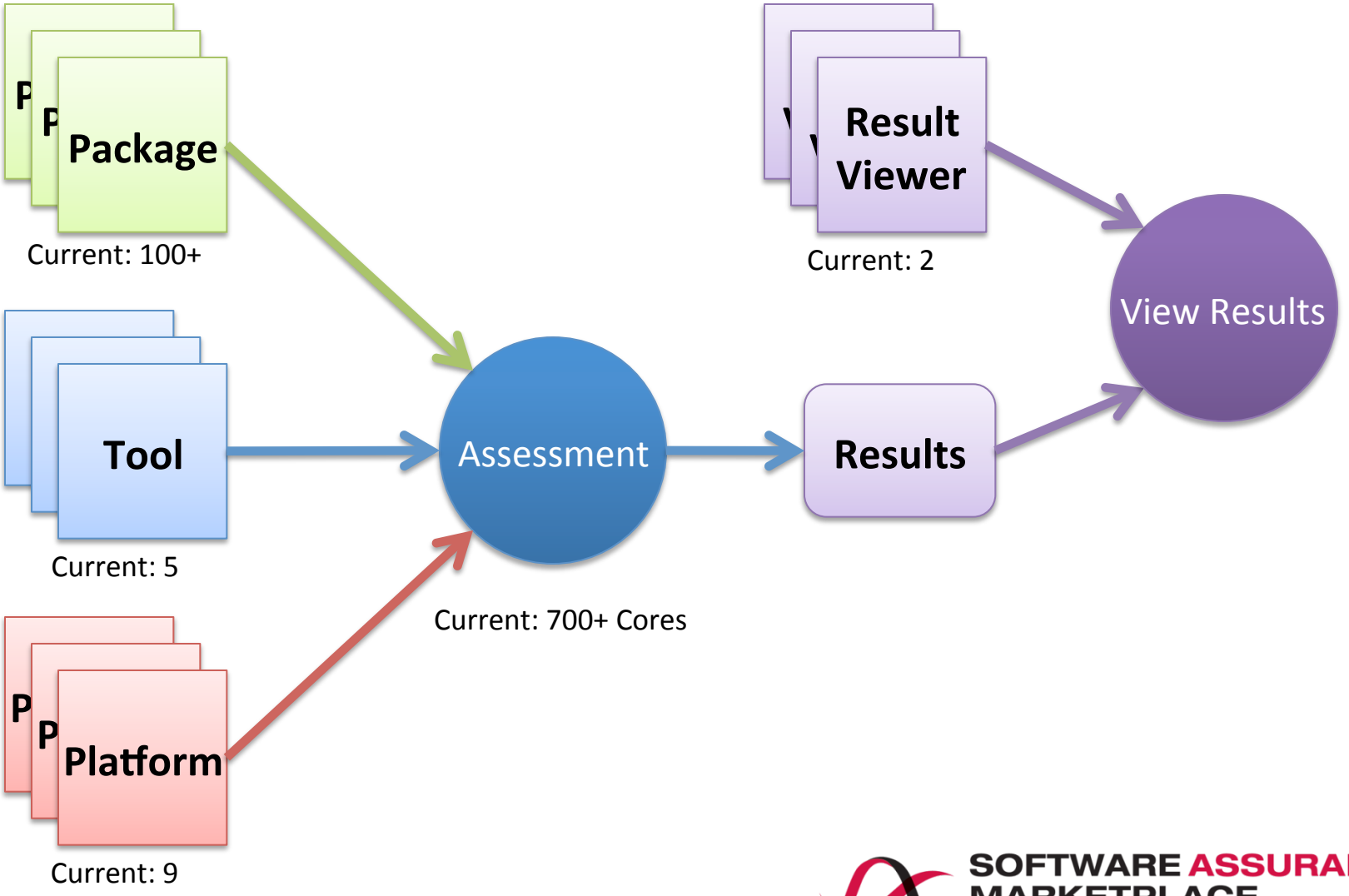
http://continuousassurance.org

Software Assurance Marketplace

- Web based facility to assess software using software assurance tools
- No cost to use
- Funded by Department of Homeland Security, Science and Technology Directorate (5 year grant)
- Supports multiple communities including:
  - Software Developers
  - **SwA Tool Developers**
  - Researchers
  - Educators

SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE

The Marketplace



SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE

Collaboration Between 4 Institutions

SATE V Involvement

- Hosted VMs used in SATE V
- Produced results for 3 SwA tools applied to SATE V software packages:
  - FindBugs
  - PMD
  - Clang Static Analyzer
- Converted results to SATE XML format

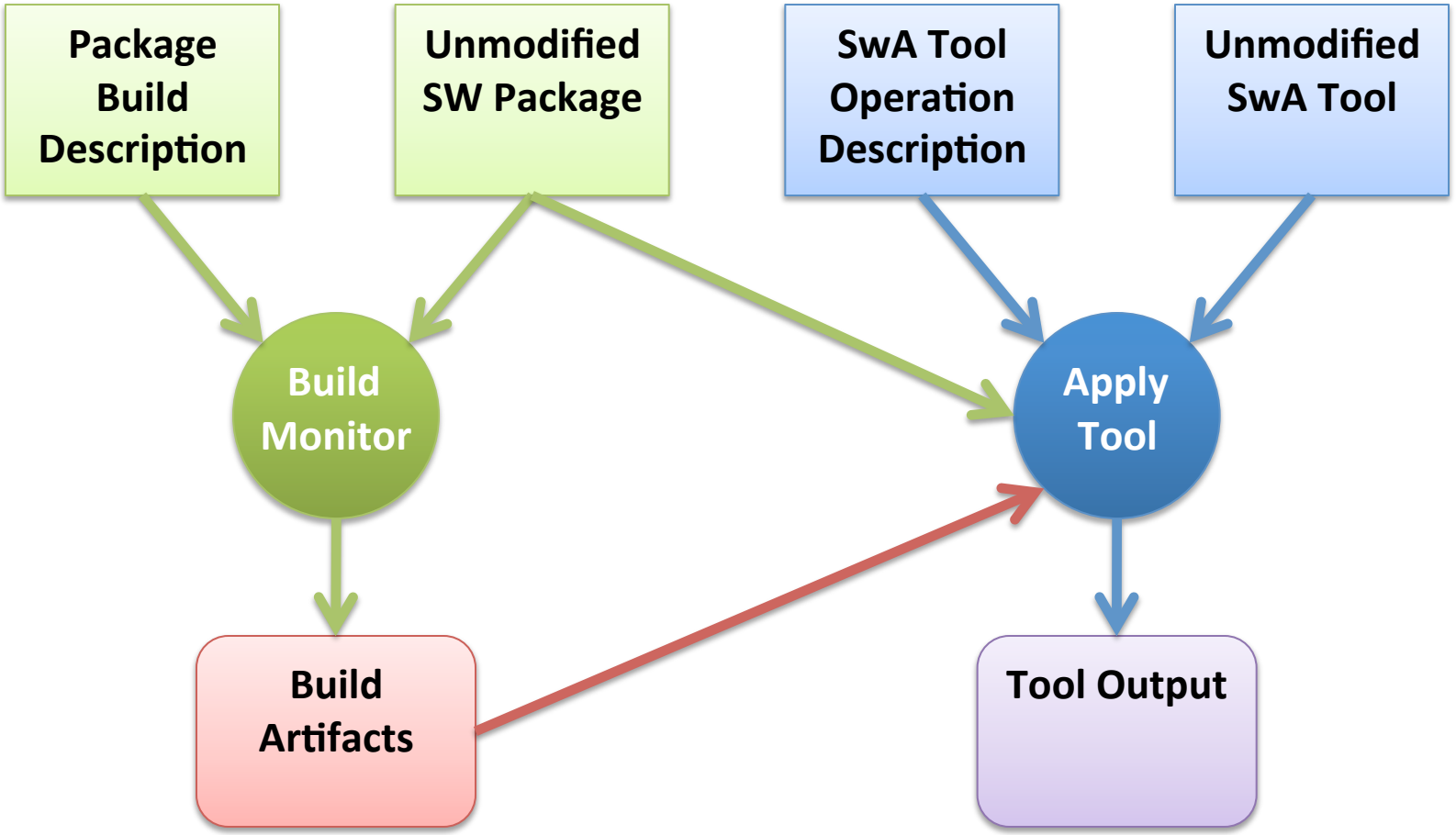
The Tool Developer and the SWAMP

(coming soon)

- Build and test your tool
- Continuous Integration and Assurance of your tool
- Expose your tool to users
  - Make your tool available to selected users
  - Easy for SWAMP users to try and use your tool
- Limited scope tools still add value
- SWAMP provides plumbing and glue
  - Simplifies correctly applying your tool
  - Easy to use for both the tool and package developer
    - No changes to code
    - No changes to build

SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE

Our Approach to Assessments



SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE

Build and Test (coming soon)

- Upload and build your tool on any of the SWAMP platforms
  - Apply SwA tools to your tool
  - Interactive debugging support
- Test your tool against packages
  - Our packages include
    - Juliet test suite for C/C++ and Java
    - Recent SATE C/C++ and Java packages
    - Working on adding older packages
  - Add your own packages

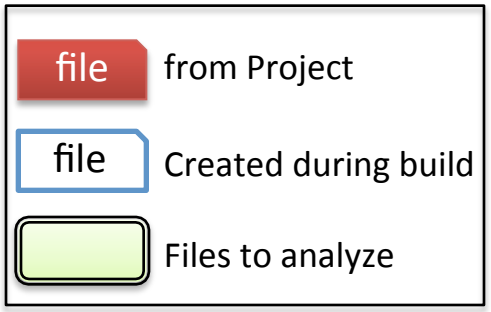
SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE

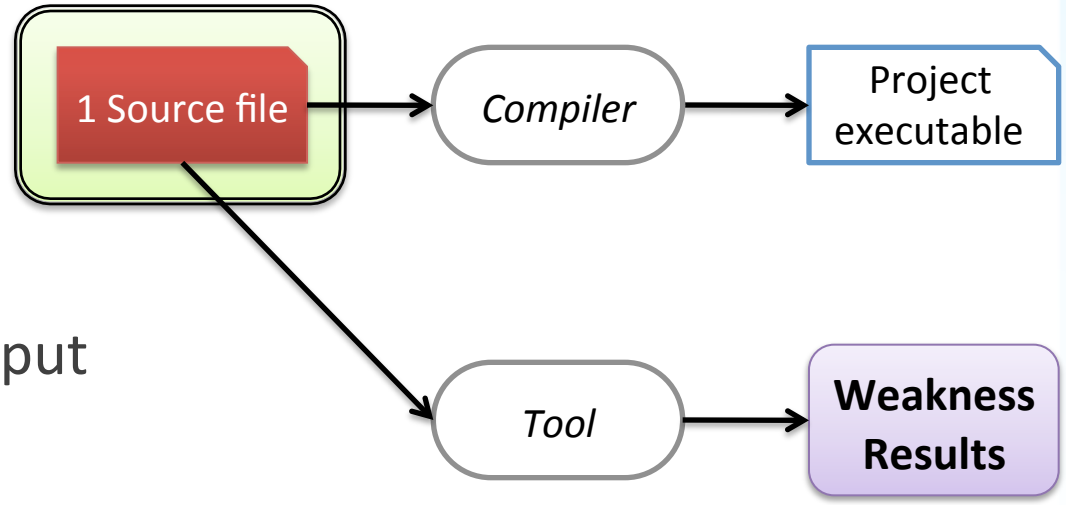
Current Approaches for Applying Open Source Tools

- Look for all sources under a directory, apply tool
  - May not be right set of files
- Give instructions on how to call the tool or modify the build system
  - Difficult for developer to modify build correctly
- Make assumptions about build environment to insert a shim to capture relevant build operations
  - Fragile if assumptions are violated
- Ignore aspects of build such as command options and environment
  - Tool operates on different code than compiled

Ideal Static Assessment

- One directory, one executable
- Source code
  - Few files
  - Standard include files
- Standard libraries
- One tool
  - Source as input
  - Weaknesses as output





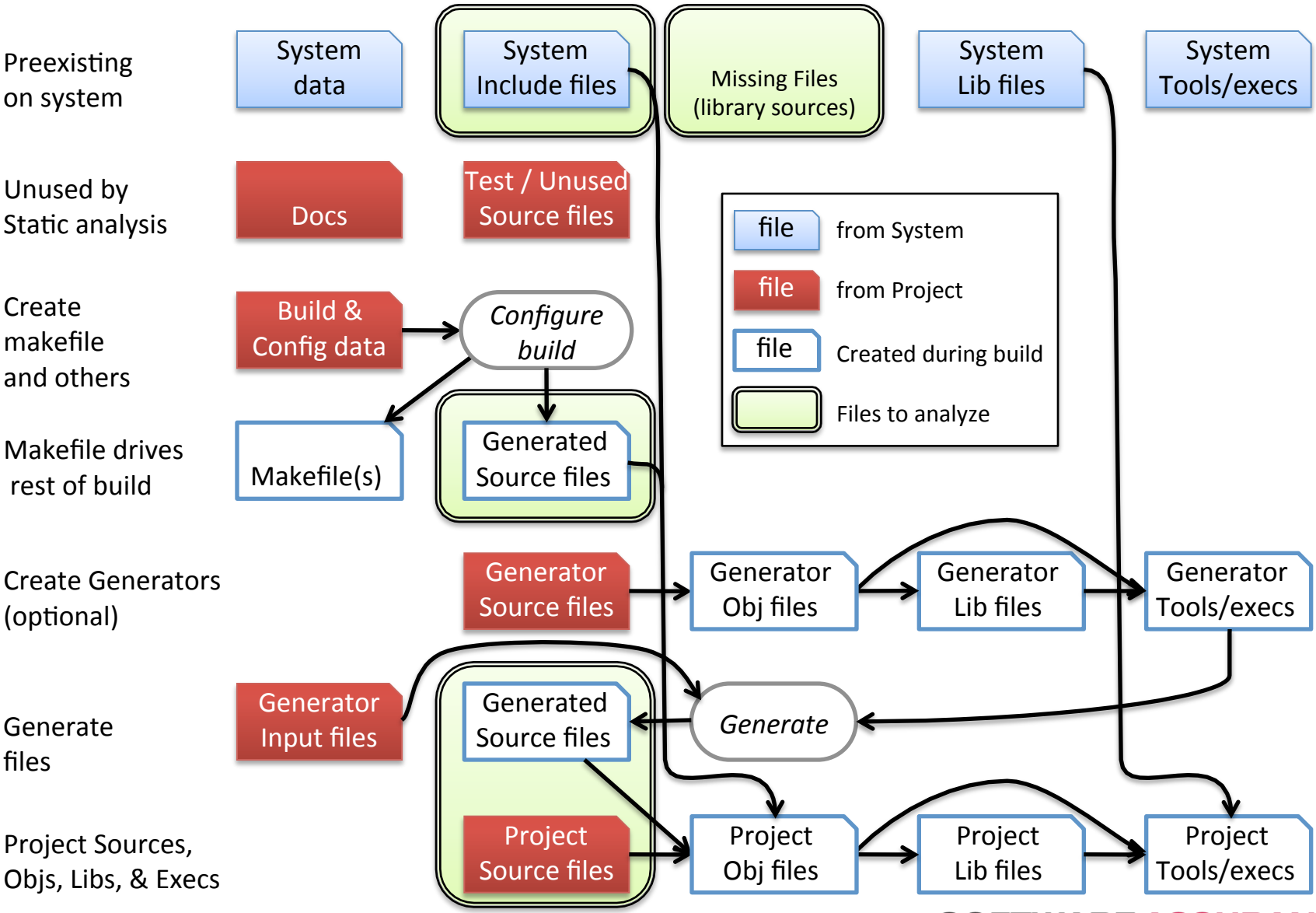
SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE

Real World Is More Complex

- Multiple source files
  - Separate compilation
  - Used to create libraries and executables
- Multiple executables
- Build generators, multi-level makefiles, custom scripts, complex conditional compiling criteria.
- Not obvious what to assess (no easily obtained list)
  - Generated files
  - System and 3rd party files outside of project directories
  - Command line arguments determine what is compiled, archived and linked, but also the source compiled through macro definitions and header file locations
- For each executable need to assess exactly those source files used in the creation of the executable

Difficult to Determine Sources to Assess

- Build information is not declarative
- The build is a many programs that run to build the software and has many layers
  - Build configuration layer
  - Make drives build process
  - Calls shell code snippets
  - Generates source code and executables for build
- Capturing build tool invocations is non-trivial

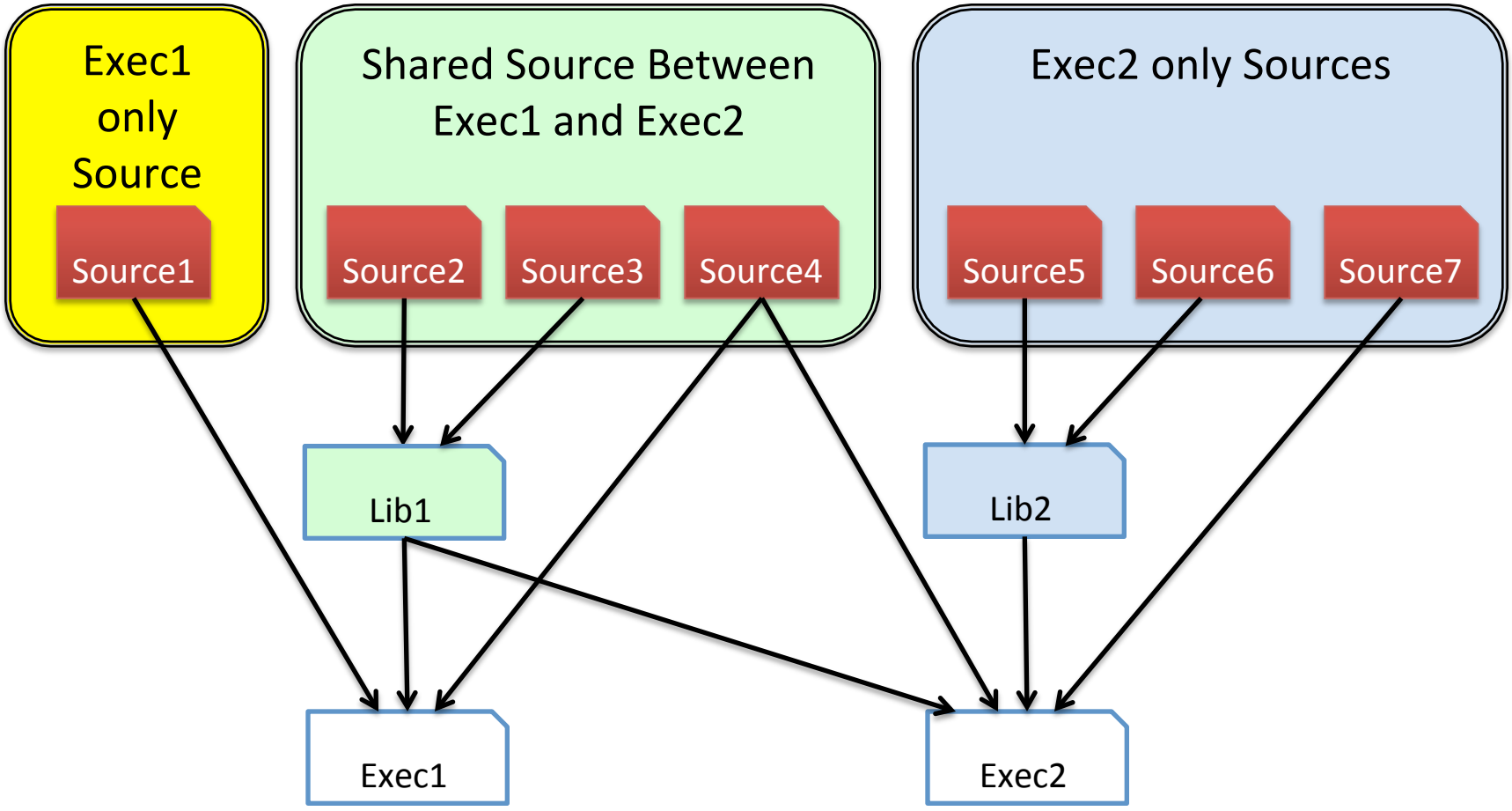


SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE

Assessing a Package

- A package is usually contains many executables
- Need to determine:
  - What executables exists
  - What source files used in build
  - Whole package analysis not acceptable
    Must be whole program by executable
- Too complex to leave to humans,
  requires automated tool
- Some tools seem to do whole package
  (all the sources compiled in the package)

Multiple Executable per Package

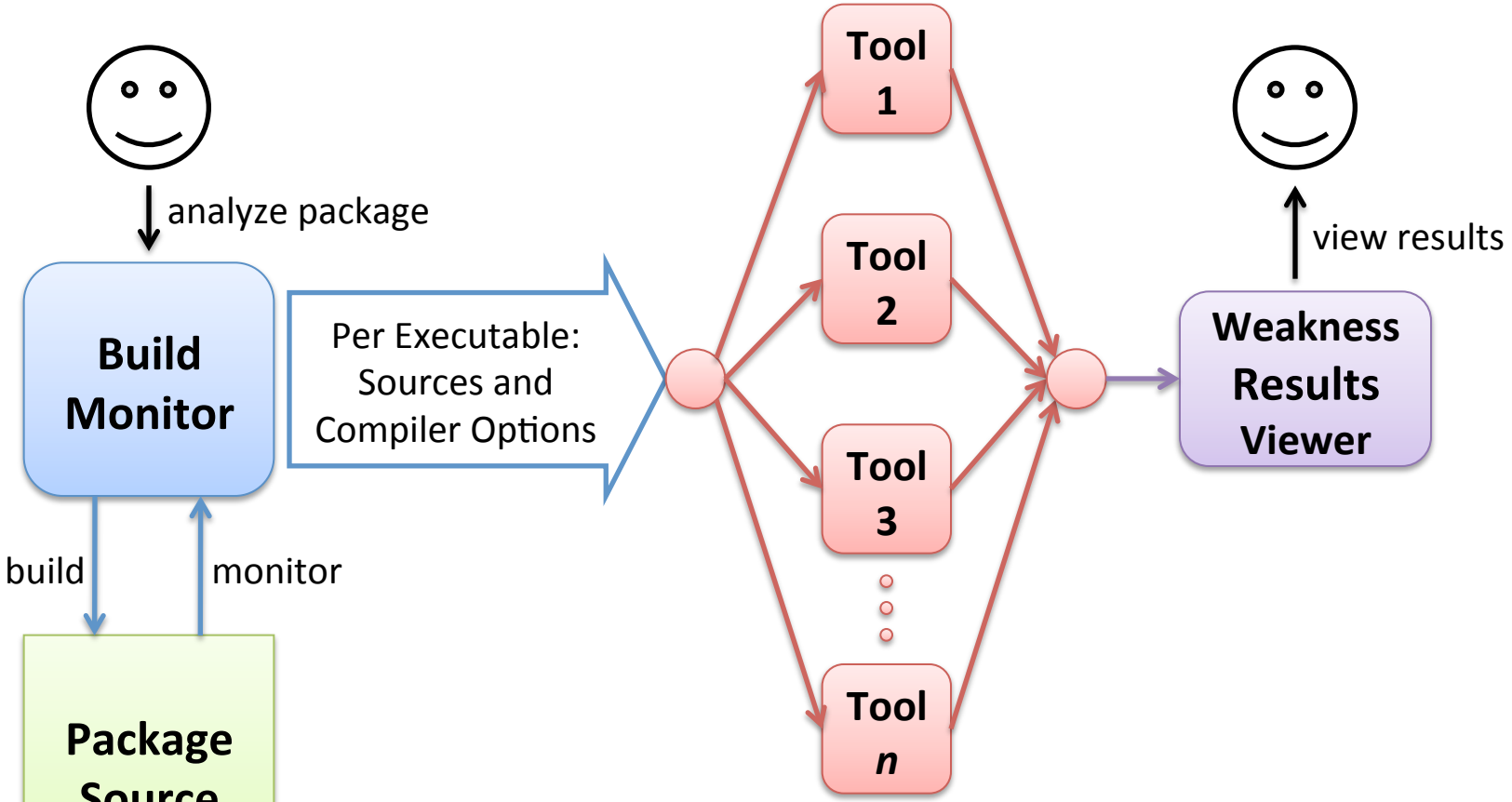


SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE

Current State

- Have generic tool to determine build information
  - Any build process for C/C++
  - Ant and Maven builds for Java
- Captures executable invocation of build process
  - Command path
  - Full arguments
  - environment
  - Current directory path
- Determine build information from known compiler, archiver, linker, and common utilities invocation
- This by itself gives tool writers a leg up
- Will be released as open source

Build Monitor Tool Use



Run each tool once for each executable with discovered sources and compiler options, tool produces weaknesses, aggregate for viewing.

SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE

Future Directions

- New tools: OCLint, Jlint, Checkstyle, …
- New versions of tools
- New language support
- New packages for testing
- New platforms: MacOS, Android, iOS, Windows, .Net, Java 8, …
- New types of testing: dynamic, …
- New result viewers
- Analytics
- APIs

SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE

Questions

http://continuousassurance.org

(Support for tool developers and other improvements coming soon)

SOFTWARE ASSURANCE MARKETPLACE
A NATIONAL CYBERSECURITY RESOURCE