DD2460 Software Safety and Security

Security Policies and Policy Enforcement

Dilian Gurov

1

Security Policies

Security policies specify the **acceptable executions** of programs. Typical policies are:

- Access control policies
- Information flow policies
- Availability policies

2

Security Policies

- **Access control policies** define restricted access to resources, like:
  - ”*only user A can read file foo*” (on a multi-user system)
  - ”*an applet can allocate at most 100KB memory*” (on a smart card platform)
  - ”*a game may send at most 3 SMSs per game*” (on a mobile device)

3

Security Policies

- **Information flow policies** capture that confidential information does not flow to a location where this confidentiality is not preserved, like:
  - ”*information about a patient should not leak from the hospital database*”
  - ”*the value of a key should stay confidential*”

4

Security Policies

- **Availability policies** restrict continuous denial of service:
  - ”*if the web page is requested, it will eventually be available*”
  - ”*the bandwith can not reduce by more than 30% of its peak value*”

5

Security Policies

- Security policies can be phrased and specified as sets of (acceptable) executions
- Hence, they are naturally phrased as predicates on sets of executions, and even better (whenever possible) as **predicates on executions**
- Program executions are usually filtered on a set of **security-relevant actions**, determined by the nature of the policy at hand

6

Security Automata

- Many security policies can be expressed by means of **security automata**
- Security automata are essentially Büchi automata with accepting states only (and an implicit non-accepting ”error” state)
- The input alphabet of the automaton is the set of security-relevant actions of the given policy
- Security automata capture **safety properties** only

7

Enforcing Security Policies

Security policies can be enforced on a program in a variety of ways:

- **Static analysis**: verify that the program adheres to the policy (i.e. satisfies the property)
- **Reference monitor**: observe the execution and terminate it if it is about to violate the policy
- **Program rewriting**: re-write the program to change its behaviour to adhere to the policy

8

Reference Monitors

- Common in hardware and system software
  - operating systems mediate access to files and other resources
  - traps caused when executing system calls can be used to invoke a reference monitor
- Must be protected from subversion by the target systems it monitors
- Must receive control whenever the target system participates in a security-relevant action

9

Reference Monitors

- The restrictions on what policies can be enforced by reference monitors come from:
  - the means by which target system events cause the monitor to be invoked (**observability**), and from
  - the means by which the monitor can influence the behaviour of the target system (**controlability**)
- For instance, security policies that govern operating system calls are feasible because traps accompany system calls

10

Reference Monitors

There are two principle kinds of reference monitors:

- **Explicit monitors**: all security-relevant actions are intercepted by a wrapper that mediates between the monitored application and the system
- **Embedded monitors**: the code of the application is re-written to merge (in-line) the monitor code around security-relevant actions

11

Embedding Security Automata

This can be performed in four steps:

1.Insert a copy of the security automaton before each security-relevant action (instruction)

2.For every automaton, evaluate all transition predicates based on the following instruction

3.Simplify the automata by eliminating edges with predicates that evaluate to false and resulting unreachable states

4.Compile the resulting automata to code that updates the monitor state or aborts the application on violation of the policy

12