Analyzing and Comparing the Protection Quality of Security Enhanced Operating Systems

Hong Chen Ninghui Li Ziqing Mao

CERIAS and Dept. of Computer Science, Purdue University

MAC in Operating Systems

- Host compromise – a serious problem
- Operating system security enhancement
  - DAC + MAC

SELinux

AppArmor

User Account Control

IE Protected Mode

What can a Security Policy Offer?

- Understand and compare the Quality of Protection

- What attacks prevented?
- How to penetrate?
- Use another distribution?

The Challenge

```
asterisk.te
####################
policy_module(asterisk, 1.0)
####################
#
# Type declarations
#
# asterisk domain
type asterisk_t;

# asterisk entrypoint
type asterisk_exec_t;

#mark asterisk_t as a domain and asterisk_exec_t
#as an entry point into that domain
init_daemon_domain(asterisk_t, asterisk_exec_t)

# PID file /var/run/asterisk.pid
type asterisk_var_run_t;
files_pid_file(asterisk_var_run_t)
```

High-Level Security Properties

Low-Level Security Policy Rules

VulSAN

- **Vul**nerability **S**urface **AN**alyzer (**VulSAN**)
  - Analyze and compare the quality of protection offered by MAC policies in Linux
- Vulnerability Surface
  - A list of attack paths

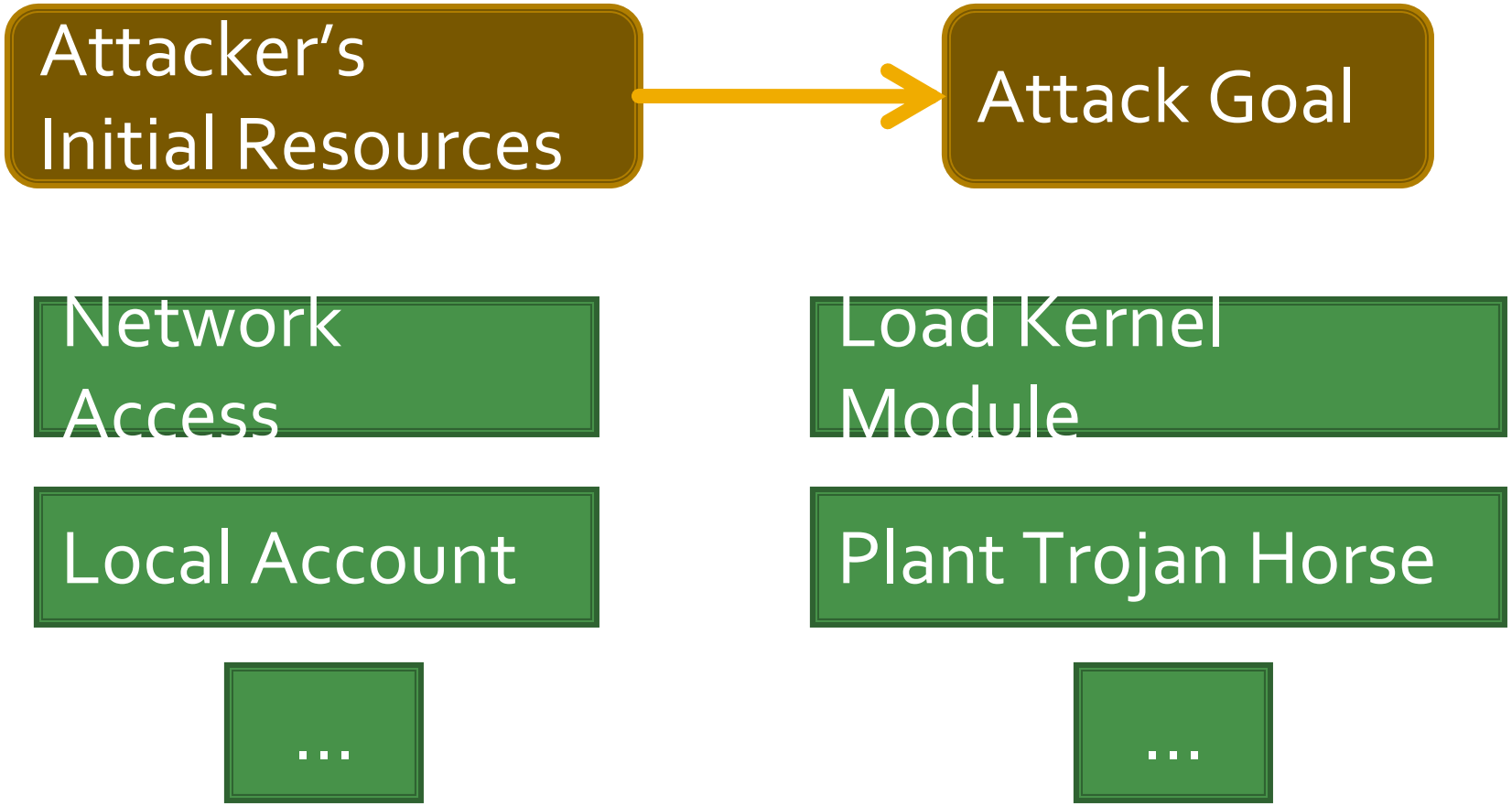
VulSAN

- Global view of the system weaknesses
- System hardening tool
- Handle both SELinux and AppArmor
- Some experiment results are provided

Outline

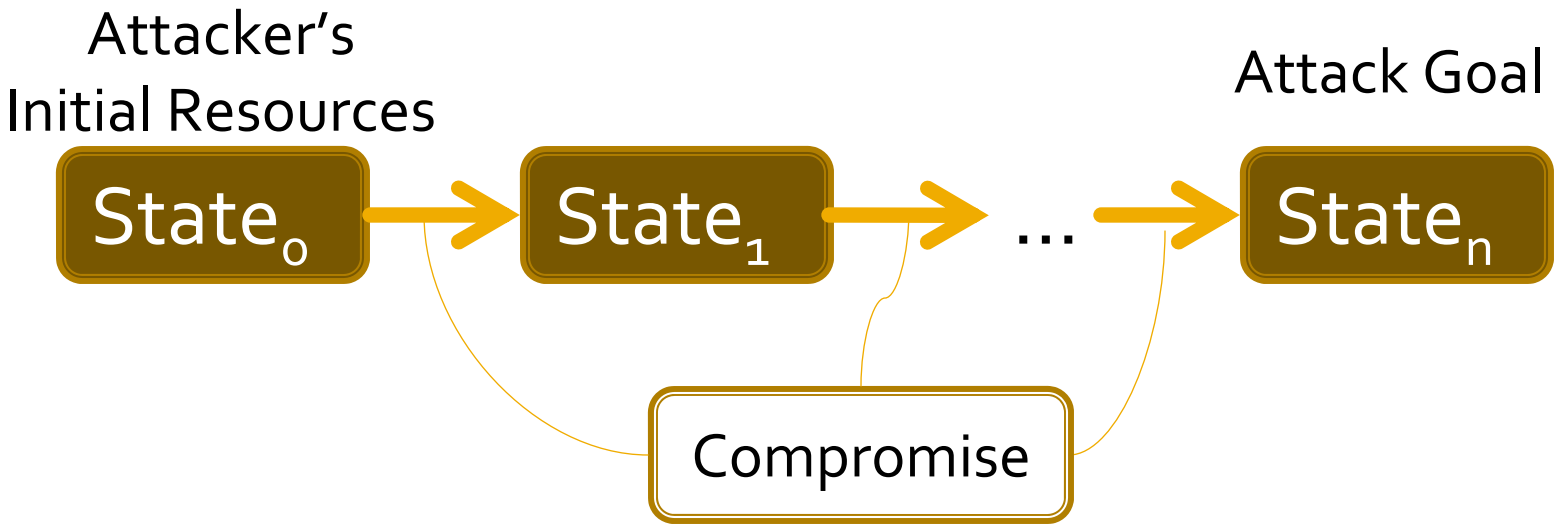
- Solution
- Implementation
- Analysis Results
- Discussion & Related Work

Attack Scenarios



The Model of Penetration

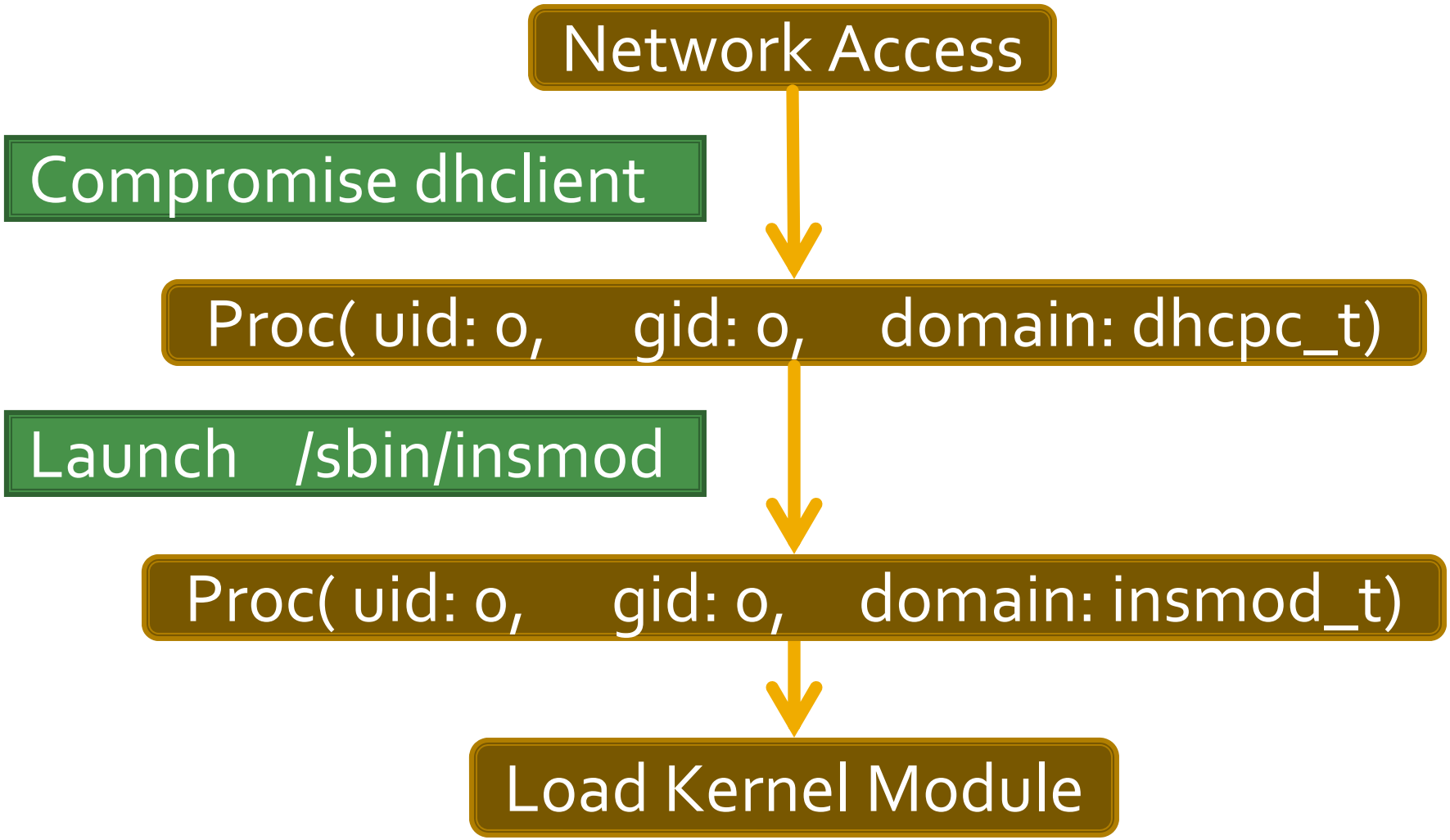
State Transition Process



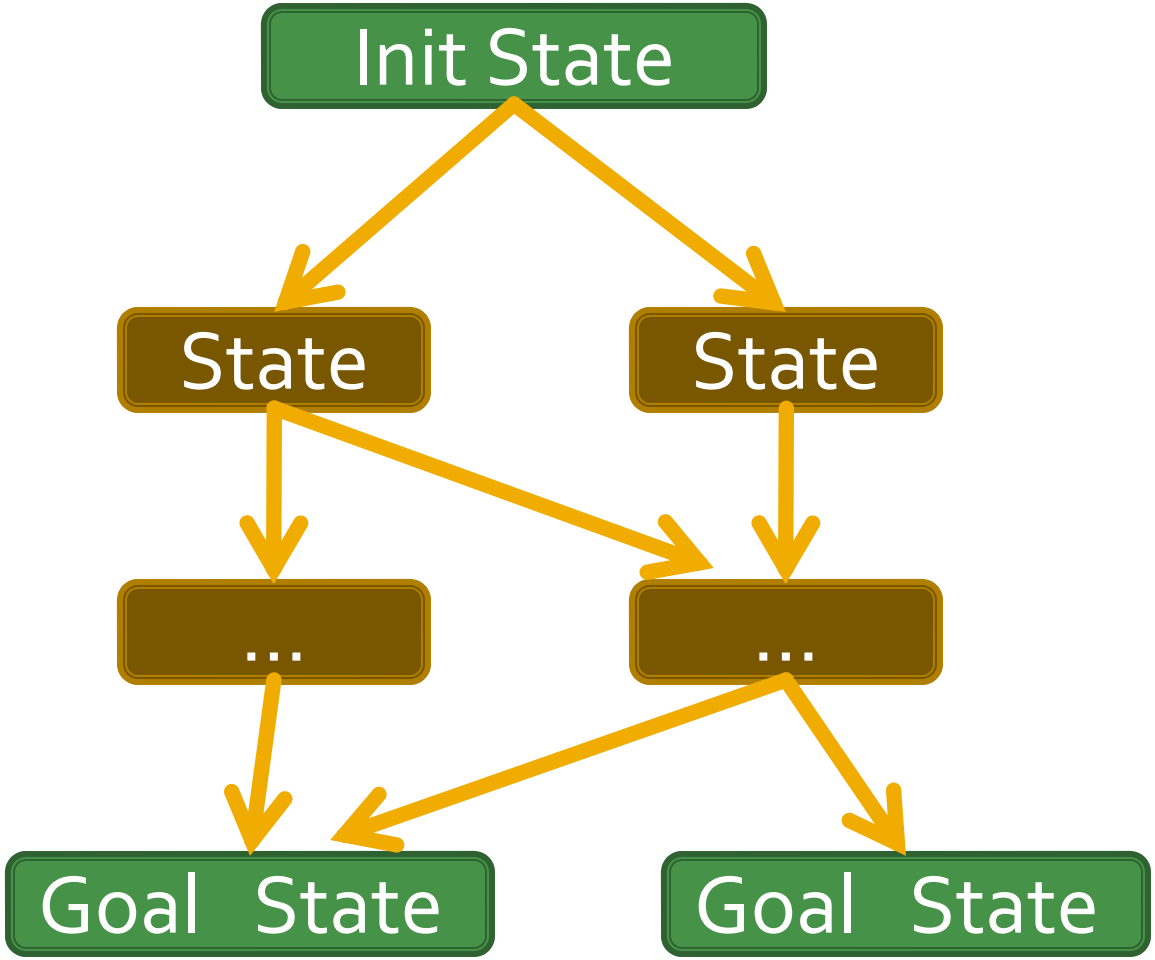
Define the State

- At process level
- Attributes relevant to access control decisions
- SELinux: *proc(uid, gid, domain)*
- AppArmor: *proc(uid, gid, profile)*

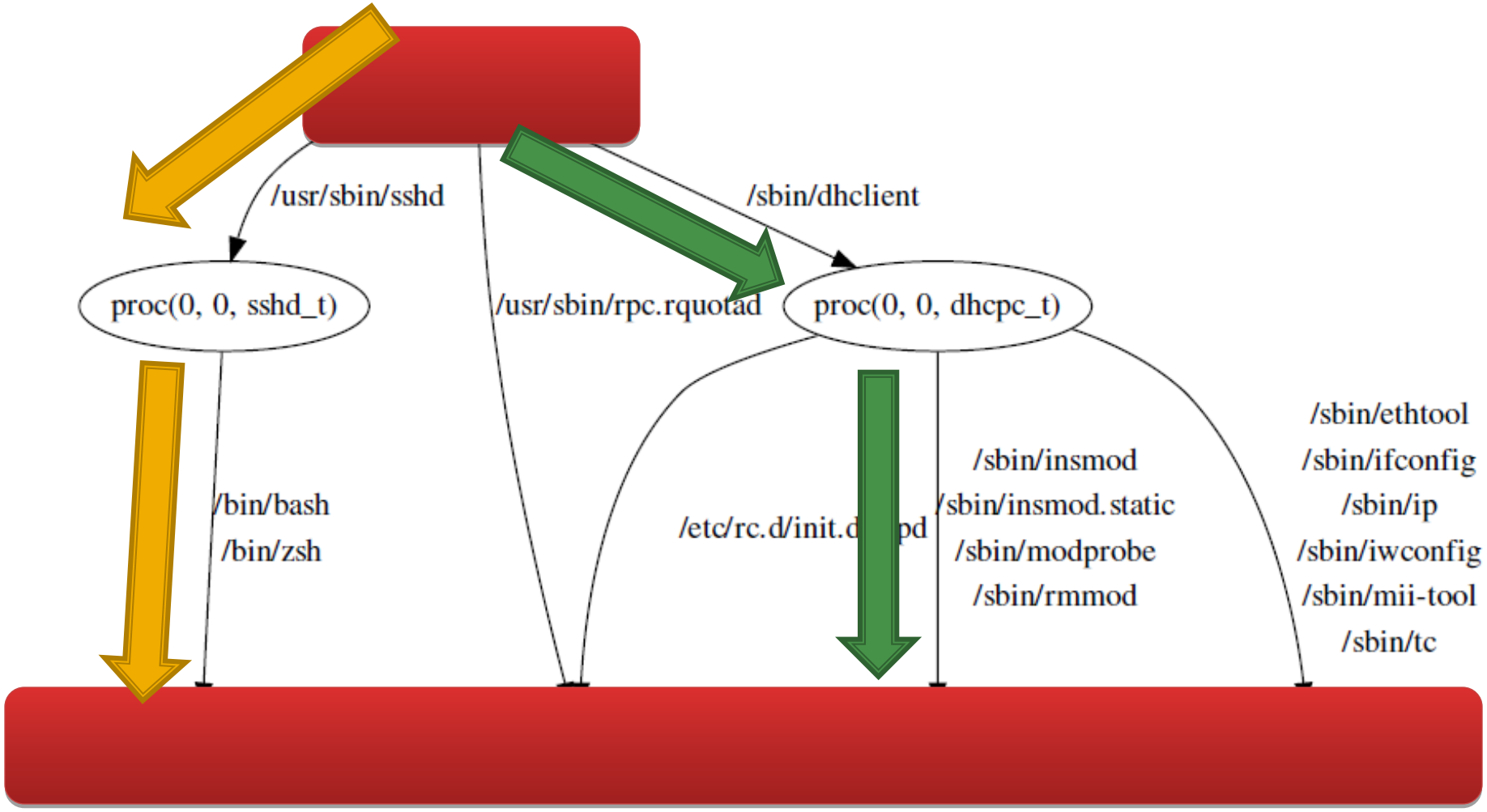
A Sample Penetration



Host Attack Graph

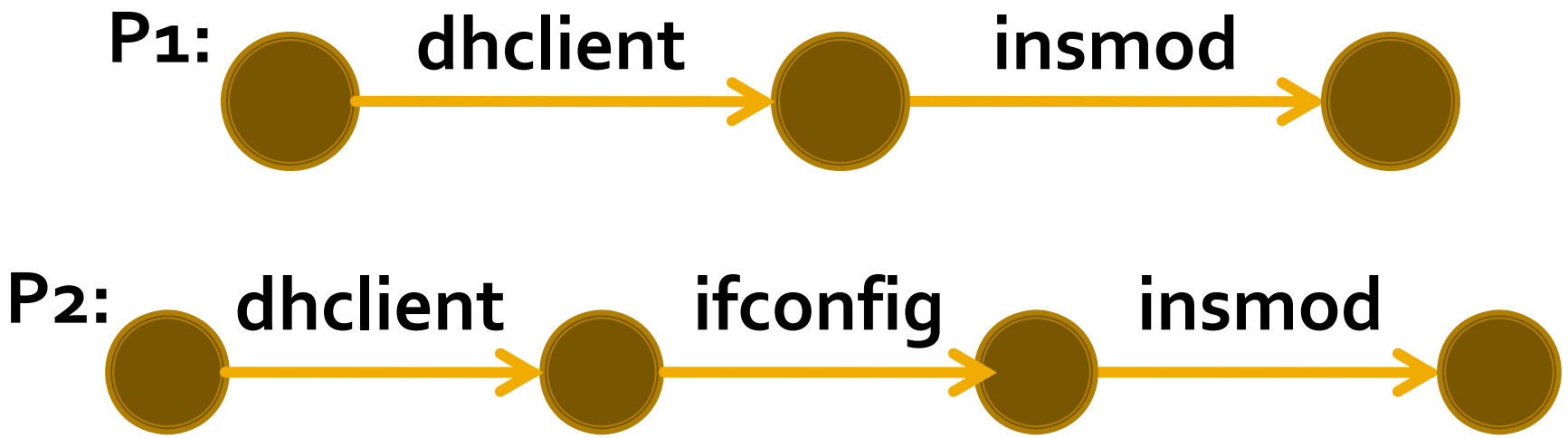


Sample Host Attack Graph



Minimal Attack Path

- Attack paths that are not superset of other attack paths

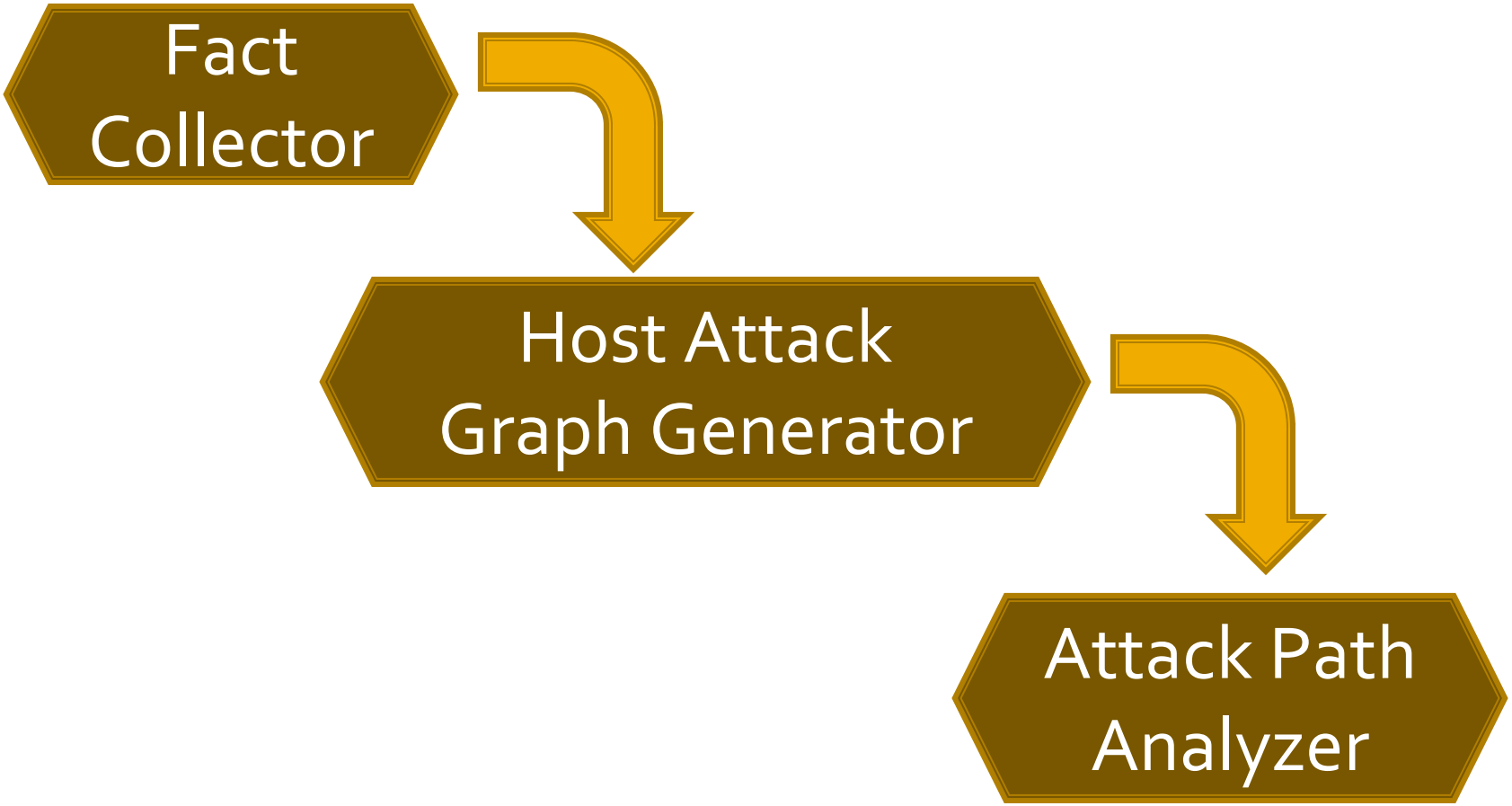


Vulnerability Surface

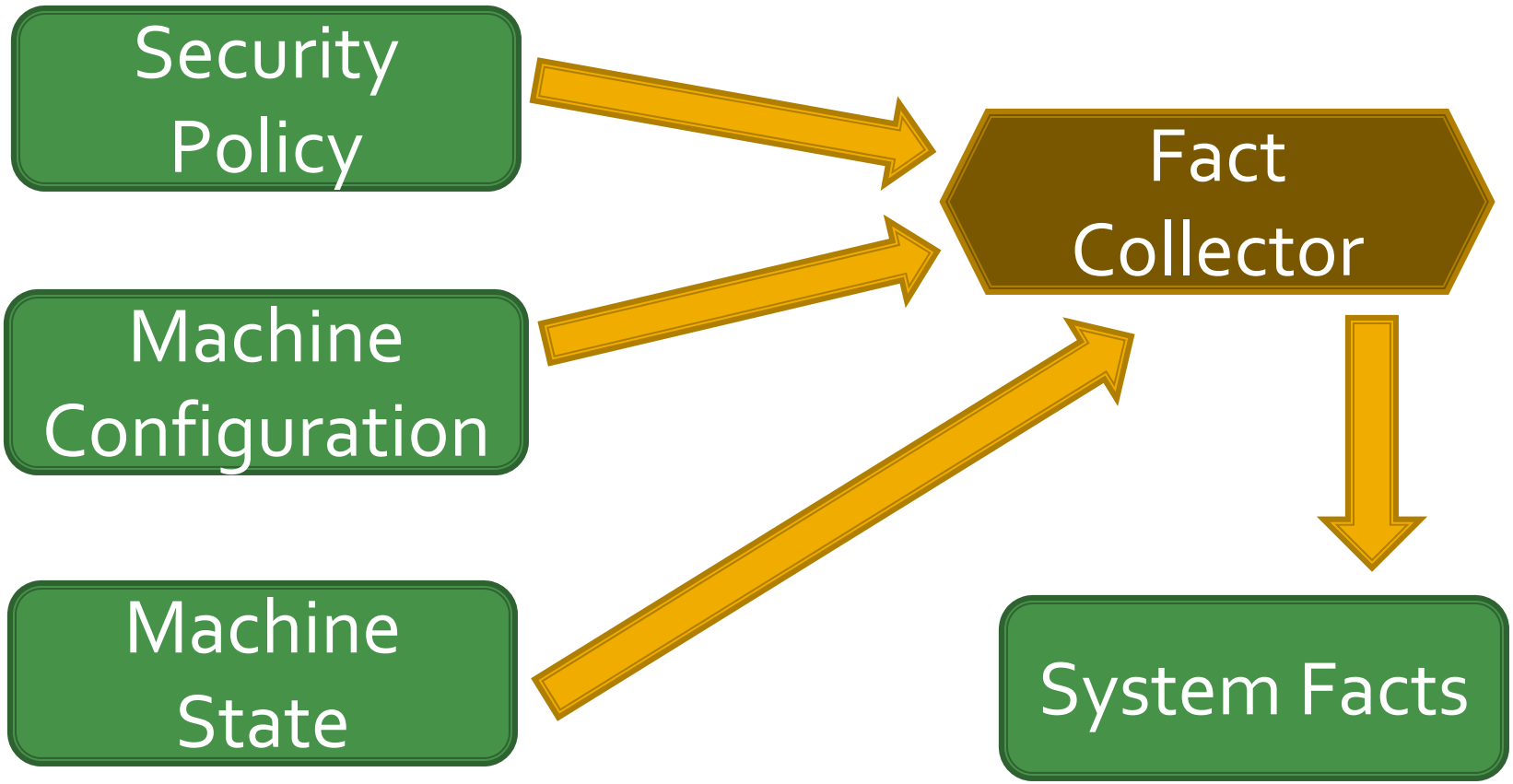
- All the minimal attack paths of an attack scenario in the host attack graph
- Different ways to compromise a system

Potential penetrations allowed by the policy
Does *not* mean a vulnerability is readily available

Components of VulSAN



Fact Collector



Sample System Facts

- Sample system fact

```
file_info(path('/usr/bin/passwd'),
  type(regular), owner(0), group(0),
  uper(1,1,1), gper(1,0,1), oper(1,0,1),
  setuid(1), setgid(0), sticky(0),
  se_user('system_u'), se_role('object_r'),
  se_type('bin_t')).
```

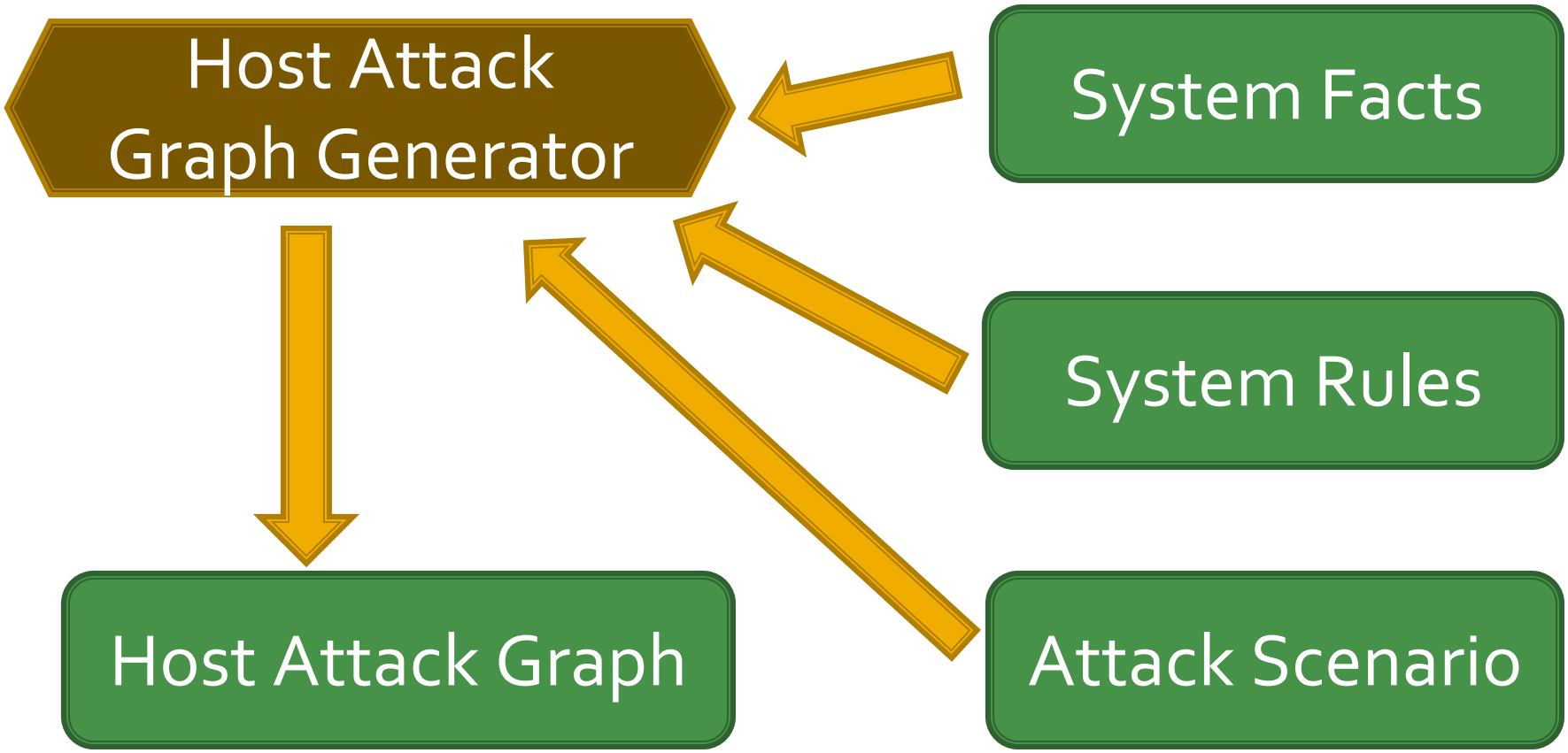
- Sample SELinux rule

```
dom_priv('user_ssh_t', 'bin_t', 'file',
  ['ioctl', 'read', 'getattr', 'lock',
   'execute', 'execute_no_trans']).
```

- Sample AppArmor rule

```
aa_access_mode('/usr/lib/postfix/master',
  '/etc/samba/smb.conf', r(1), w(0),
  ux(0), px(0), ix(0), m(0), l(0)).
```

Host Attack Graph Generator

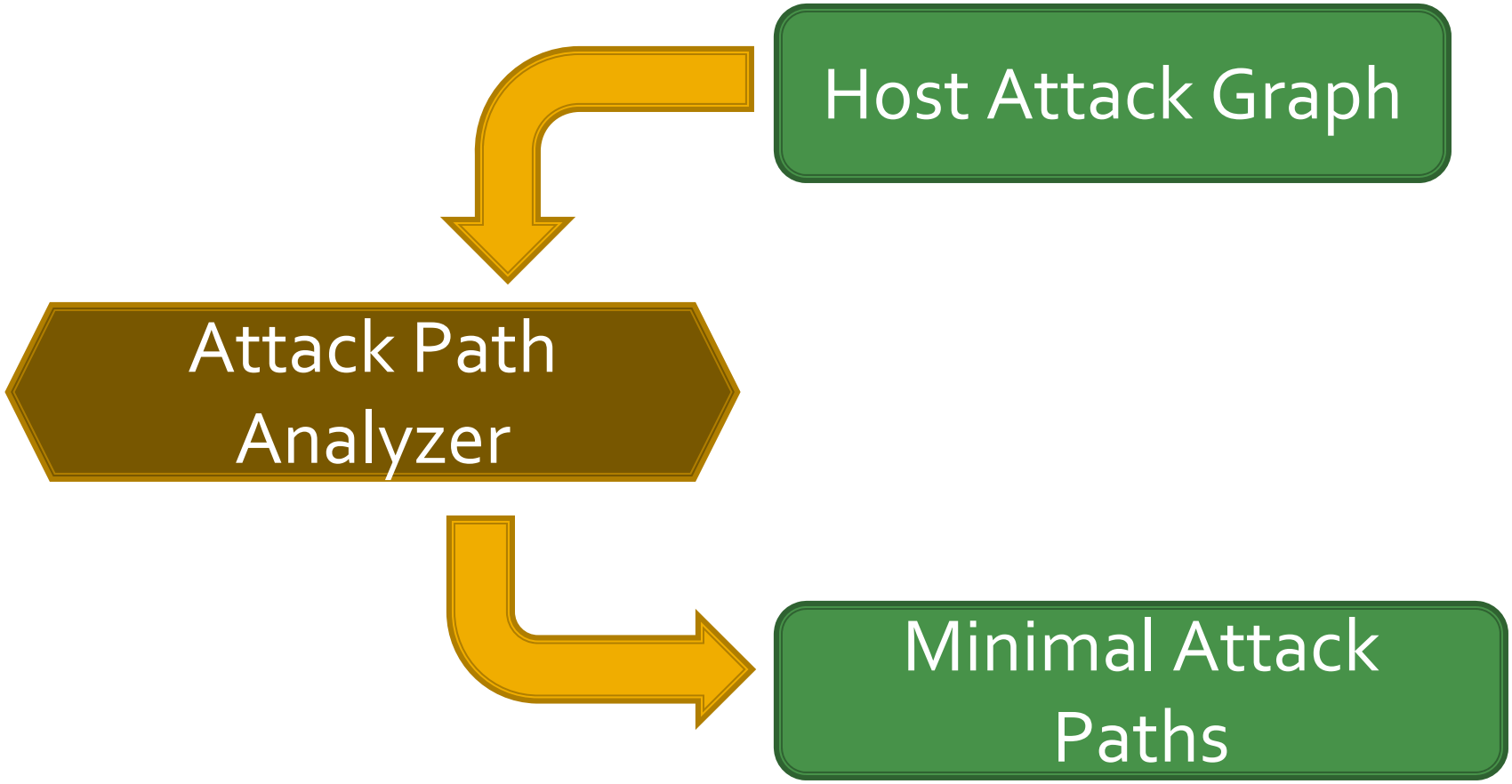


Sample System Rules

```
se_can_execute_type(Domain, Type, NewDomain) :-
se_typetrans(old_dom(Domain), new_dom(NewDomain), type(Type)),
!,
se_domain_privilege(domain(Domain), type(Type), class(file), op(execute)),
se_domain_privilege(domain(Domain), type(NewDomain), class(process), op(transition)),
se_domain_privilege(domain(NewDomain), type(Type), class(file), op(entrypoint)).

se_can_execute_type(Domain, Type, NewDomain) :-
se_domain_privilege(domain(Domain), type(Type), class(file), op(execute)),
se_domain_privilege(domain(Domain), type(Type), class(file), op(execute_no_trans)),
NewDomain = Domain.
```

Attack Path Analyzer



Analysis & Comparison

SELinux

Ubuntu Server Edition 8.04

DAC only

AppArmor

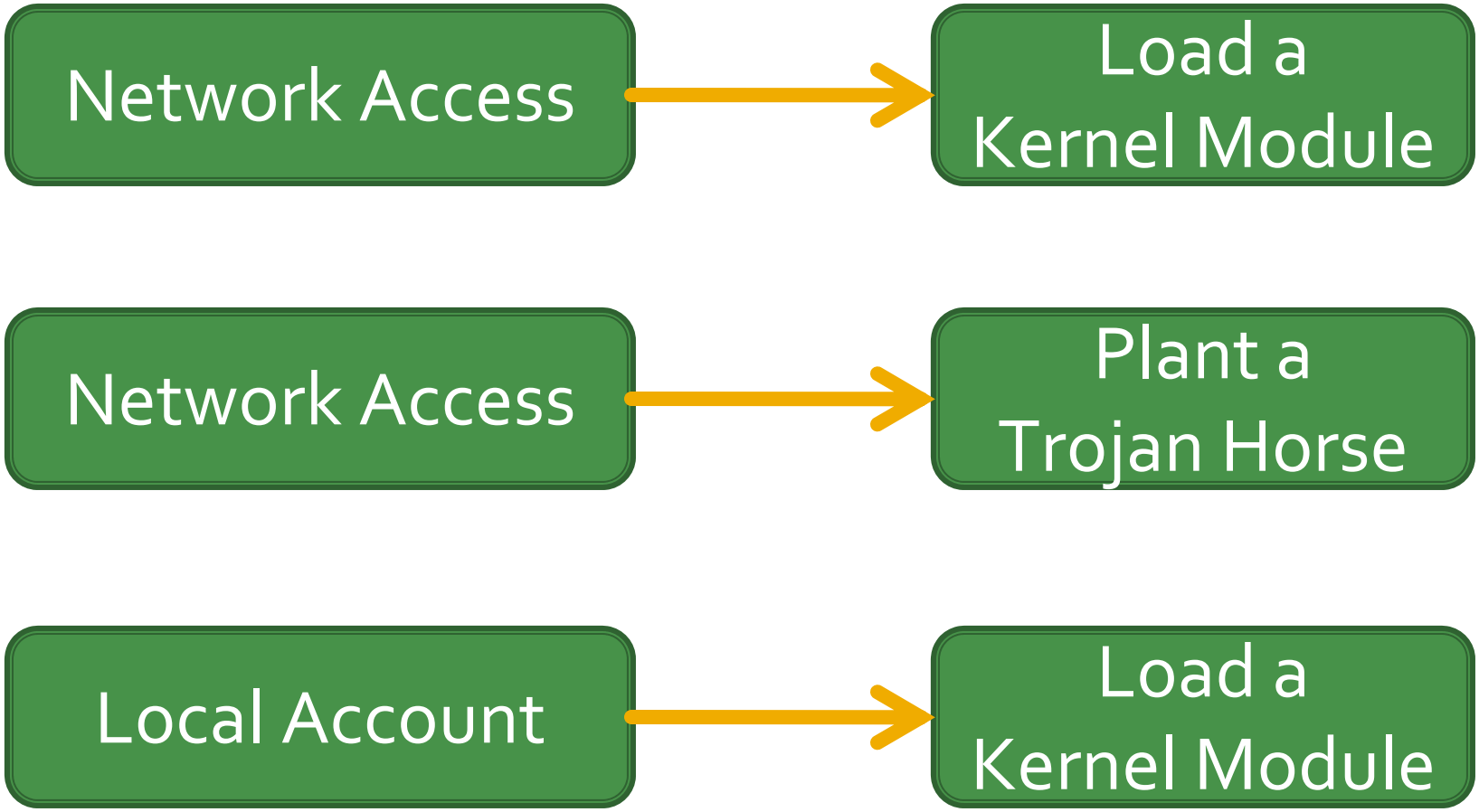
AppArmor

SUSE Linux Server Edition 10

SELinux

Fedora 8

Attack Scenarios

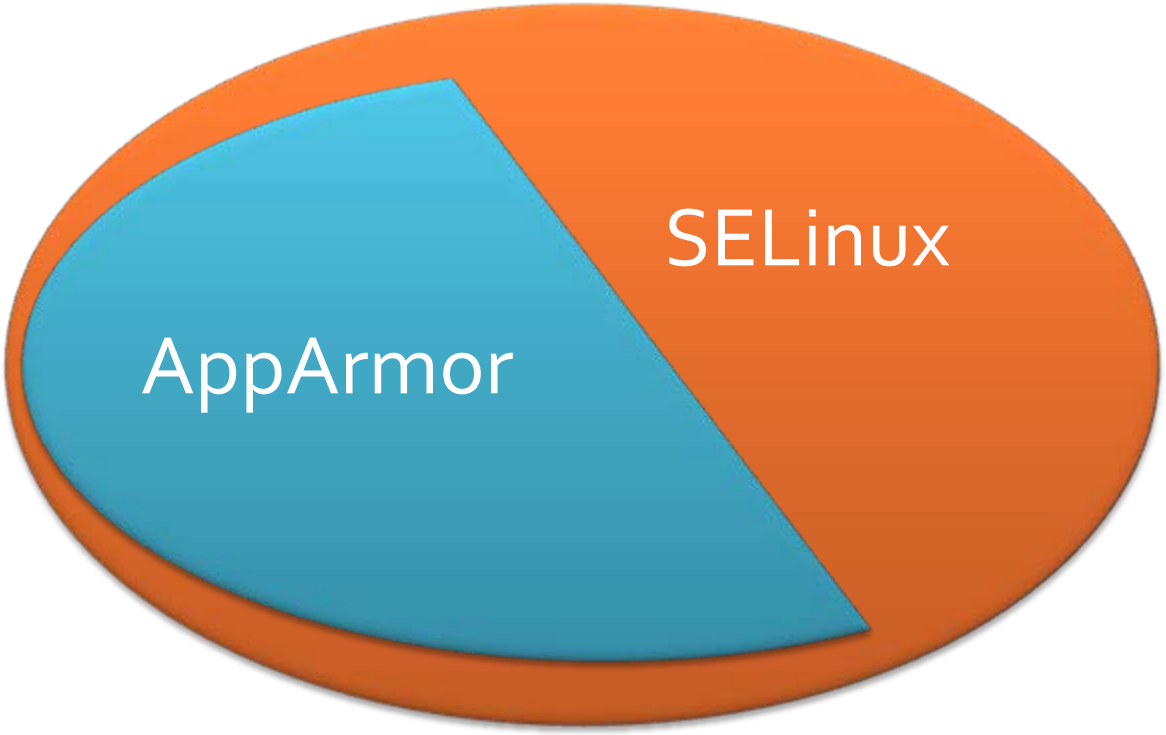


SELinux vs. AppArmor

- Which one is better?
- Mechanisms + policy + host configuration
- Configuration
  - Ubuntu Server Edition 8.04
  - SELinux & AppArmor with standard policy
  - A certain set of daemon programs

SELinux vs. AppArmor

Vulnerability Surface



SELinux vs. AppArmor

- Unique attack paths of SELinux
  - Privileged programs run under unconfined_t:
    nmbd, smbd, vsftpd, portmap, and rpc.statd
  - Confinement not as tight as AppArmor:
    cupsd and dhclient
  - Setuid confinement:
    ping, passwd
- Conclusion – with data
  - In this configuration, AppArmor provides better protection

Discussion

- Control of the process – `1` or `0`
- Considers only direct exploitation
- Snapshot of the system

Related Work

- Attack surface [Howard 04, HPW 03]
  - Entrypoints of attacks
- Attack graph [SHJ+ 02, OBM 06]
  - Problem space: host vs. network
  - Consideration of potential attack
  - Additional analysis on host attack graph

Related Work

- SELinux policy analysis
  - APOL [setools, HN06]
  - Gokyo [JZC03, JSZ03]
  - NETRA [NSR+06]
  - PAL [SS04]
  - PALMS [HRC+07]
  - SLAT [GHR+05]
  - SELAC [ZV04]

- Differences
- Measurement
- Support AppArmor
- DAC policy + system configuration

Related Work

- Compare two mechanisms
  - SELinux vs. AppArmor [Leitner06]
  - Mechanism
  - Policy + System Configuration
  - Objectively measurement

Summary

- Introduce the notion of *vulnerability surfaces* under *attack scenarios* as the measurement of the QoP offered by MAC policies in operating systems
- Implement VulSAN for computing vulnerability surfaces for Linux systems with SELinux or AppArmor
- Analyze and compare SELinux and AppArmor in several recent Linux distributions, and show tightening opportunities

Thank you!

Questions?

Motivation (3)

- Challenge
  - Configurations: Low-level policy rules
  - Security Goals: High-level security properties
- Complex MAC systems
  - Difficult to analyze the quality of protection offered by different MAC policies
  - Difficult to compare between different policies and distributions

Implementation Prolog Encoding

- Sample system states

```
(1) file_info(path('/usr/bin/passwd'), type(regular), owner(0), group(0),
    uper(1,1,1), gper(1,0,1), oper(1,0,1), setuid(1), setgid(0), sticky(0),
    se_user('system_u'), se_role('object_r'), se_type('bin_t')).
(2) user_info('root', 0, 0).
(3) group_info('mail', 8, [dovecot]).
(4) process_running(4412, 0, 0, '/usr/lib/postfix/master', system_u, system_r, initrc_t).
(5) process_networking(4412).
```

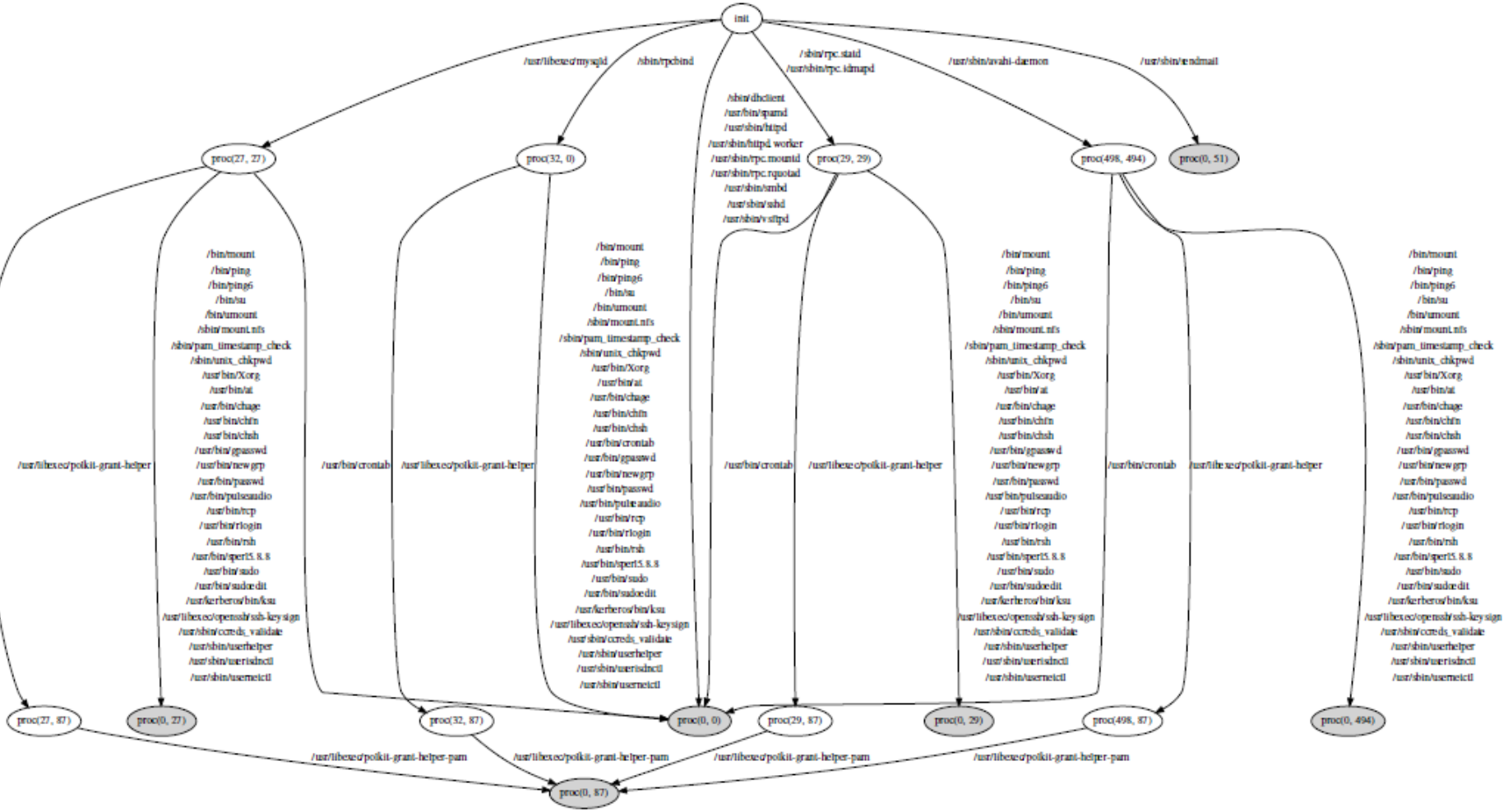
- Sample SELinux rules

```
(1) dom_priv('user_ssh_t', 'bin_t', 'file', ['ioctl', 'read', 'getattr',
    'lock', 'execute', 'execute_no_trans']).
(2) se_typetrans(old_dom('user_ssh_t'), new_dom('user_xauth_t'), type('xauth_exec_t')).
(3) se_domain('user_ssh_t').
(4) se_type('bin_t').
```

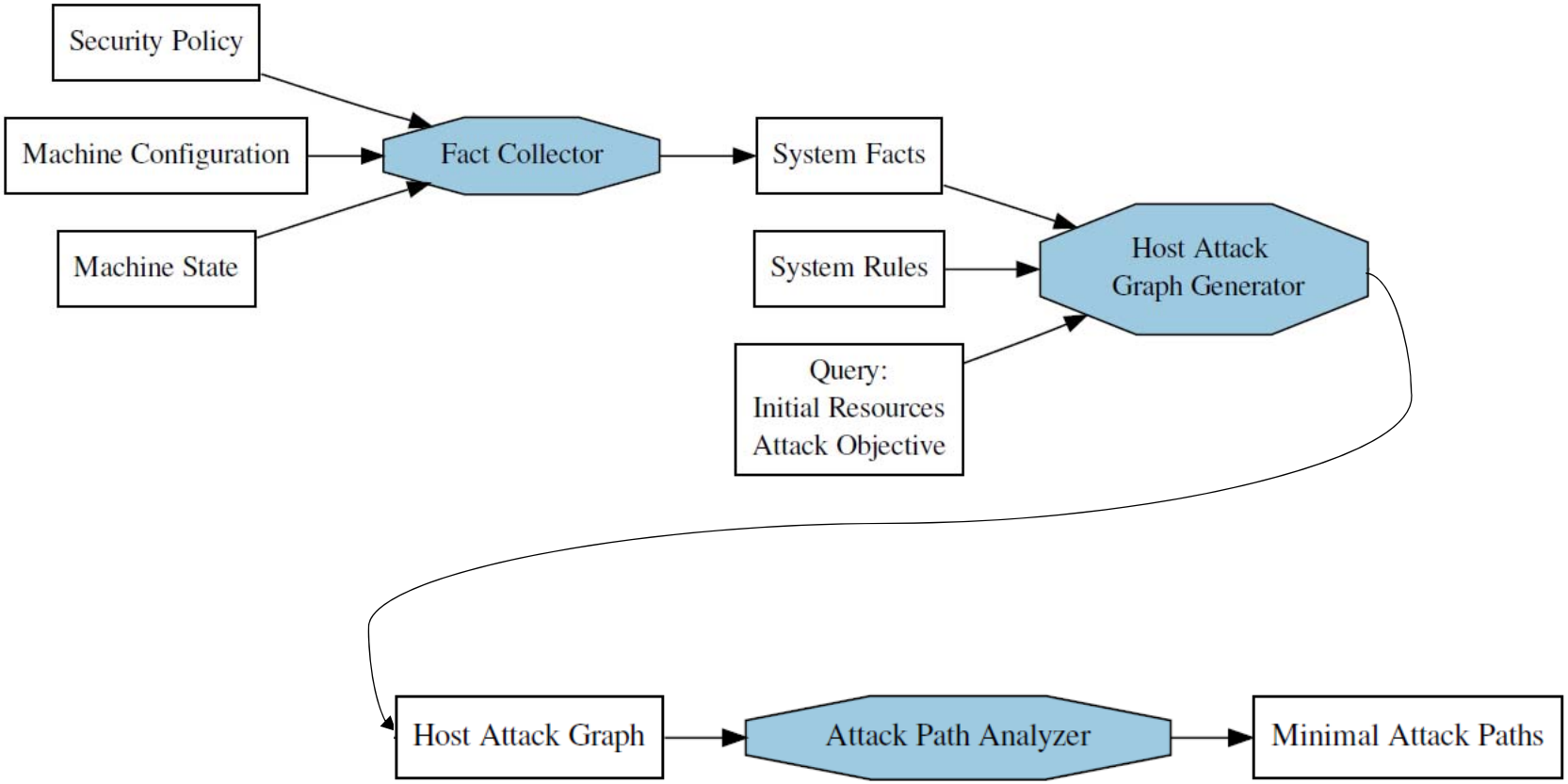
- Sample AppArmor rules

```
(1) aa_capability('/usr/lib/postfix/master', 'net_bind_service').
(2) aa_access_mode('/usr/lib/postfix/master', '/etc/samba/smb.conf',
    r(1), w(0), ux(0), px(0), ix(0), m(0), l(0)).
```

Another Host Attack Graph



Implementation Components



SELinux vs. AppArmor

Common Minimal Attack Paths

Unique Attack Paths of SELinux

SELinux compared to AppArmor
/usr/sbin/apache2 /usr/sbin/rpc.mountd /usr/sbin/named SUID* /usr/sbin/mysqld SUID* /usr/sbin/sshd
/usr/sbin/nmbd /usr/sbin/smbd /usr/sbin/vsftpd /sbin/portmap SUID** /sbin/rpc.statd SUID** /usr/sbin/cupsd /sbin/unix_chkpwd /sbin/dhclient SUID** /sbin/dhclient /lib/dhcp3-client/call-dhclient-script /usr/sbin/named /bin/ping /usr/sbin/named /usr/bin/passwd /usr/sbin/mysqld /bin/ping /usr/sbin/mysqld /usr/bin/passwd

Sample Host Attack Graph