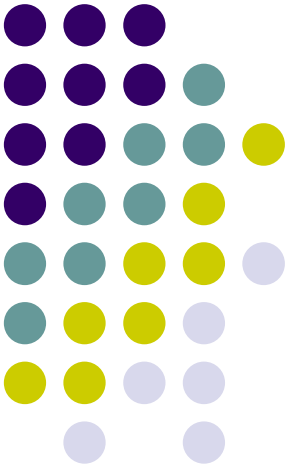
A Methodical Defense against TOCTTOU Attacks: The EDGI Approach

Calton Pu and Jinpeng Wei

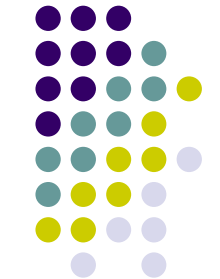
Georgia Institute of Technology



IEEE International Symposium on Secure Software Engineering

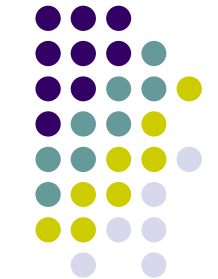
March 14, 2006. Arlington, Virginia

Outline of the Presentation



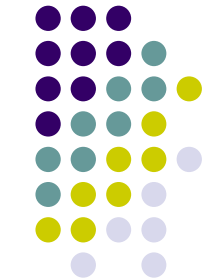
- Definition of TOCTTOU
- CUU model of TOCTTOU vulnerabilities (Under review)
- EDGI prevention of TOCTTOU vulnerabilities
- Related work and conclusion

Definition and Scope

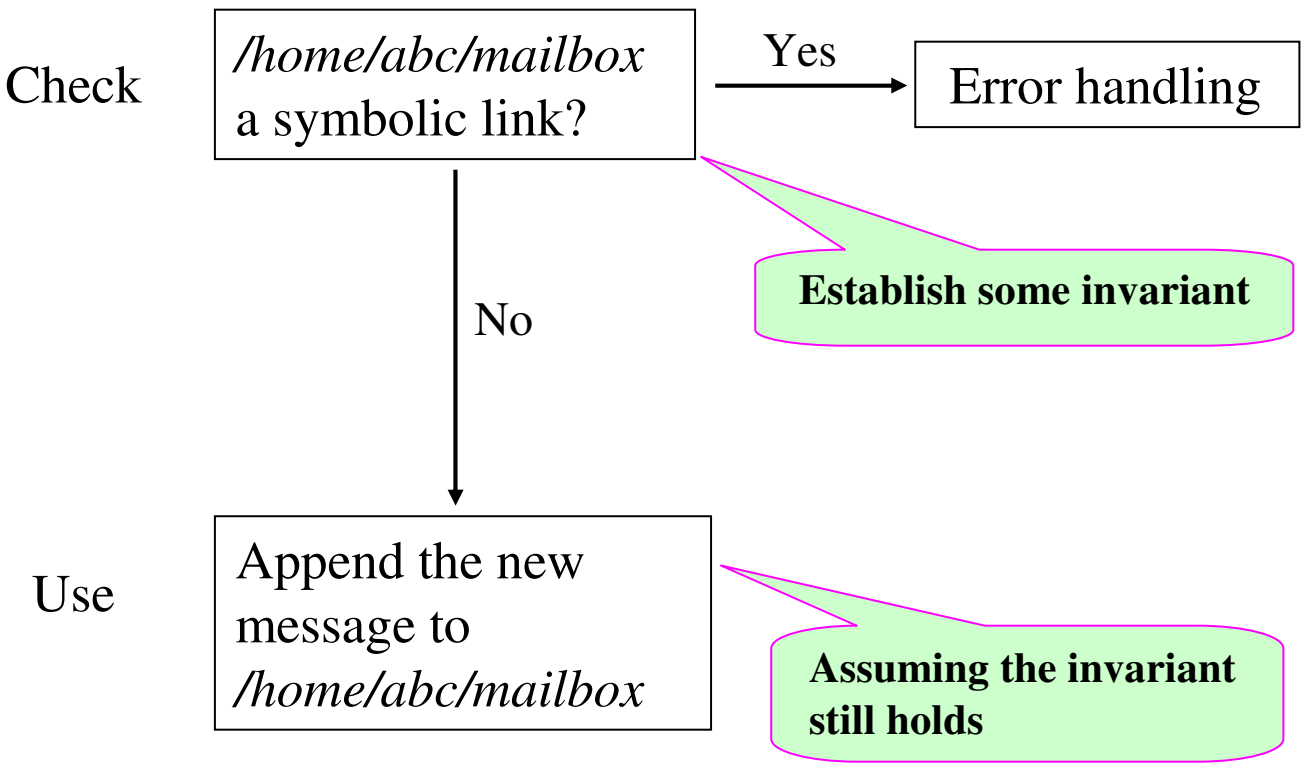


- TOCTTOU – Time of Check To Time of Use, a kind of race condition in Unix-style file systems
- Check – Establish some invariant (precondition) about a file
- Use – Operate on the file assuming that the invariant is still valid

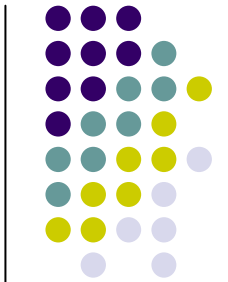
Sendmail Example

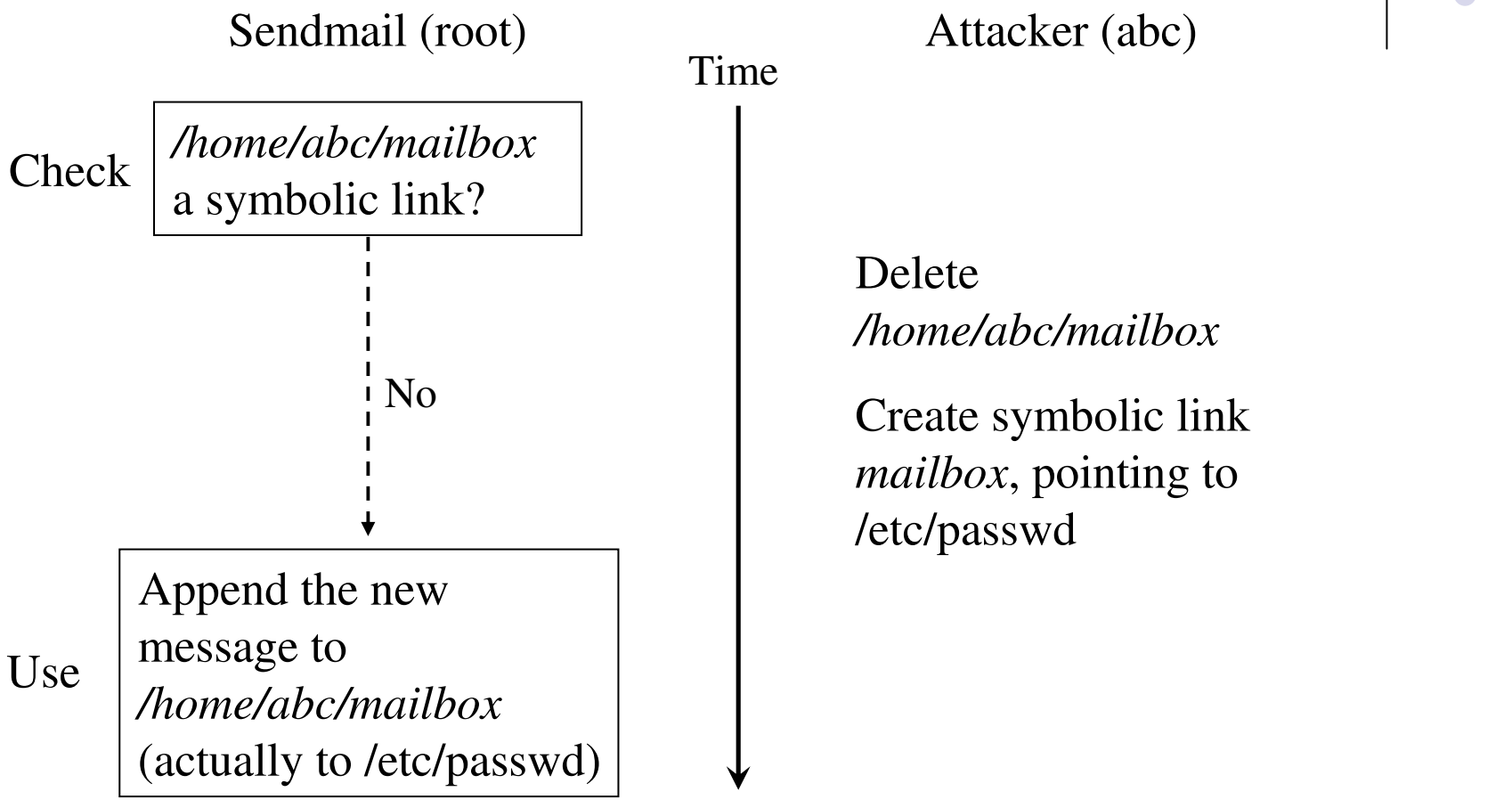


- Run as root
- Operate on files owned by normal users



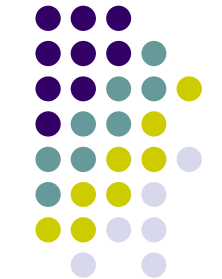
Sendmail Example (cont.)





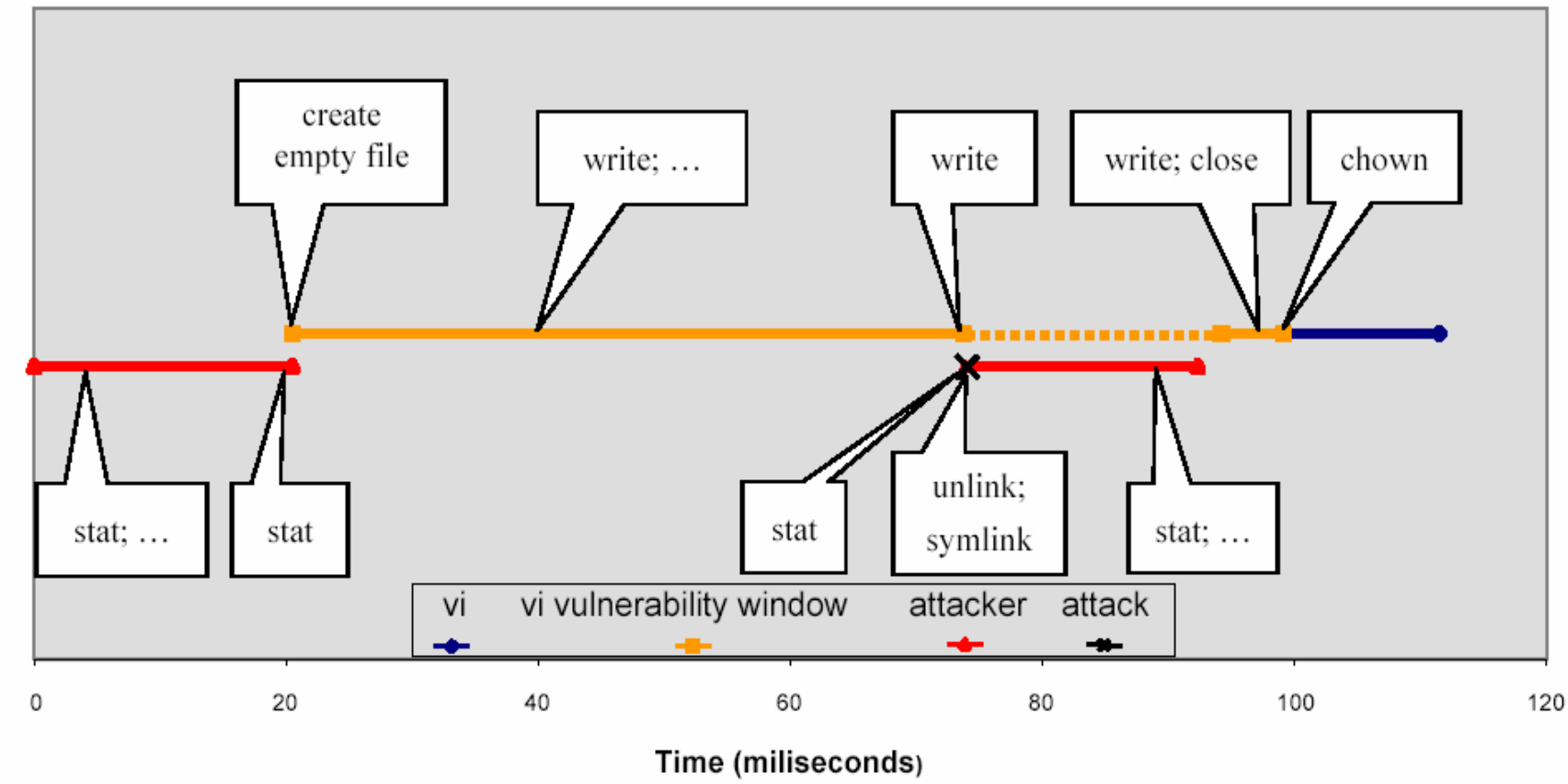
Effect: The attacker may get unauthorized root access!

Vi 6.1 Vulnerability [FAST'05]



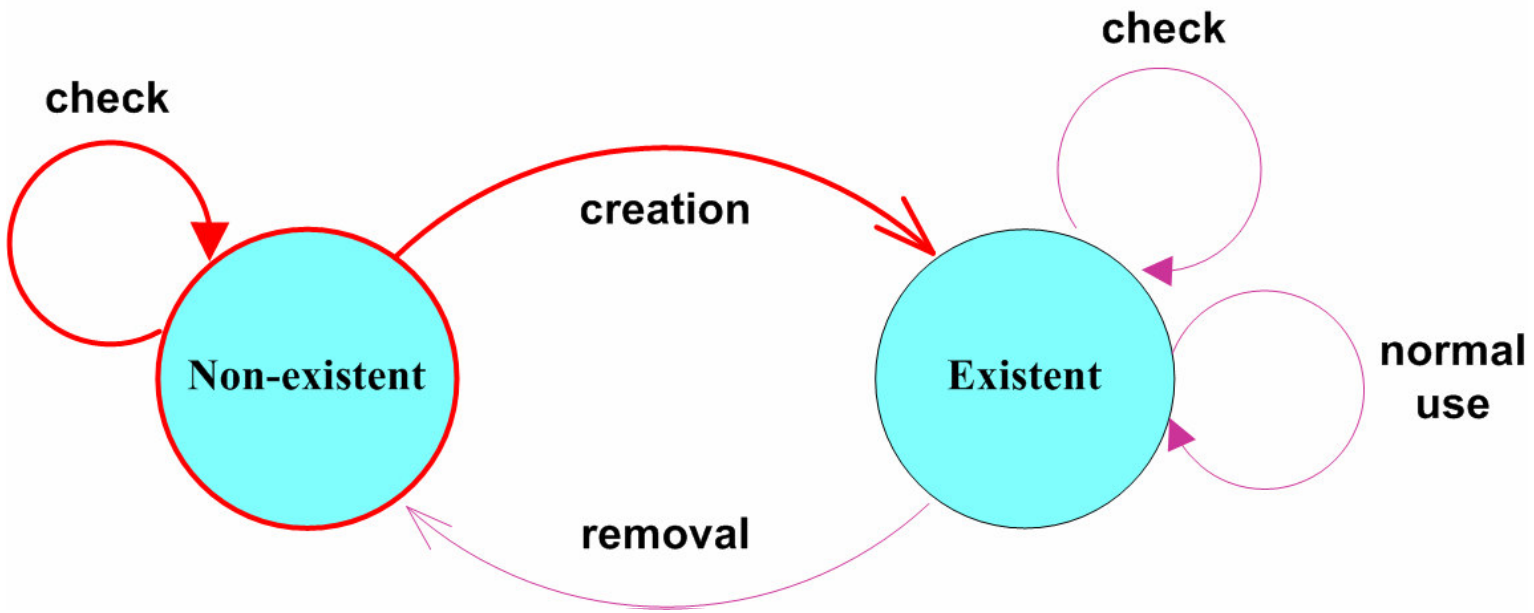
- The vulnerability happens when
  - vi is run by root
  - vi is editing a file owned by a normal user (can be an attacker)
  - vi saves the file being edited
- TOCTTOU pair: <**open, chown**>
  - **open** creates a new file for writing
  - **chown** changes the owner of the new file to the normal user.

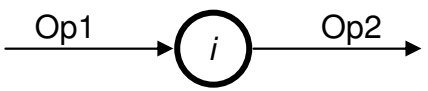
Event Analysis of Vi Exploit [FAST'05]



Successful attack changes the owner of /etc/passwd to the attacker!

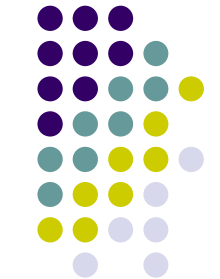
CUU Model (1)





TOCTTOU Pair	Invariant
<check, creation>	Non-existent
<removal, creation>	Non-existent
<check, normaluse>	Existent
<creation, normaluse>	Existent
<normaluse, normaluse>	Existent

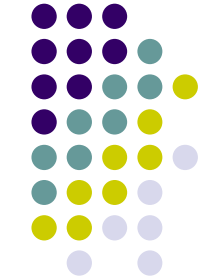
CUU Model (2)



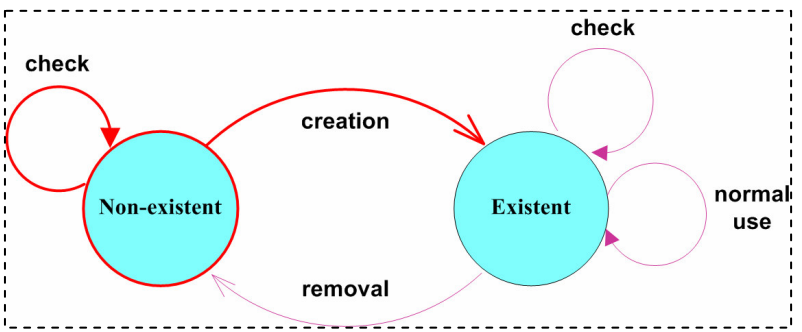
Use	Explicit check	Implicit check
Create a regular file	CheckSet × FileCreationSet	FileRemovalSet × FileCreationSet
Create a directory	CheckSet × DirCreationSet	DirRemovalSet × DirCreationSet
Create a link	CheckSet × LinkCreationSet	LinkRemovalSet × LinkCreationSet
Read/Write/Execute or Change the attribute of a regular file	CheckSet × FileNormalUseSet	(FileCreationSet × FileNormalUseSet)∪ (LinkCreationSet × FileNormalUseSet)∪ (FileNormalUseSet × FileNormalUseSet)
Access or change the attribute of a directory	CheckSet × DirNormalUseSet	(DirCreationSet × DirNormalUseSet)∪ (LinkCreationSet × DirNormalUseSet)∪ (DirNormalUseSet × DirNormalUseSet)

CreationSet = FileCreationSet ∪ LinkCreationSet ∪ DirCreationSet

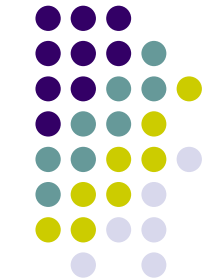
EDGI – Event Driven Guarding of Invariants



- Treat the invariant as a sophisticated lock
- The scope of this lock covers a TOCTTOU pair on a file.
- The lock owner is called an invariant holder
- Users other than the invariant holder are not allowed to remove or create the file associated with the lock.



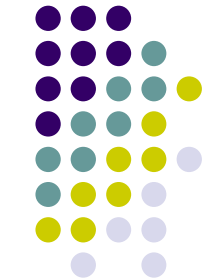
Invariant Design Options



- Providing new APIs to acquire and release the lock (invariant)
  - No false positives
  - Can have false negatives
  - Put a burden on application developers (legacy apps)

- Managing invariant-related locks within the kernel, transparent to the applications
  - No false negatives
  - No changes to kernel APIs and applications (legacy and future)
  - Can have false positives

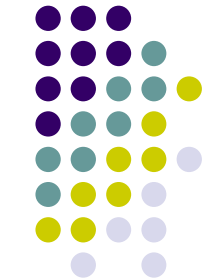
Inferring Invariant Scope



- The first user of a file becomes the invariant holder of that file
- Subsequent uses extends the invariant scope

  vi: open chown chmod …
- An Invariant prevents other users from creating or deleting the file ➔ preventing TOCTTOU!
- The sequence ends when the holder process terminates, upon which the invariant is released

Remaining Issues



- Deadlock and live lock – timeout, tainted flag

User 1: check, use, use?

User 2: delete/create (Failed)

- Invariant preemption

User 1: check, use, use, use, …

Root: delete/create (Failed)

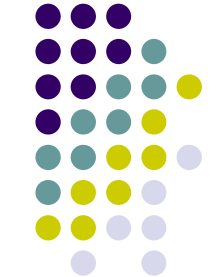
- Invariant inheritance

User 1 (process 1) check, fork, exit

User 1 (process 2) use …

User 2: delete/create

ECA and Invariant Handling



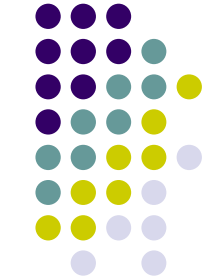
- Events:
  - File system calls such as **access, open, mkdir**, ...
  - Process operations: **fork, execve, exit**
- Conditions: specified in terms of some new state information
  - fsuid, refcnt, tainted, gh_list
- Actions:
  - Creation, removal, or update of invariants

Incarnation: An Example ECA Rule

Event	Condition	Action
Any system call on *f*	**refcnt** = = 0	Set *f*'s state as actively used (**refcnt**++); set its **tainted** flag as false, **fsuid** as current user id, record current pid and current system time in the **gh_list**.

fsuid, refcnt, tainted, gh_list are per-file state information for *f*

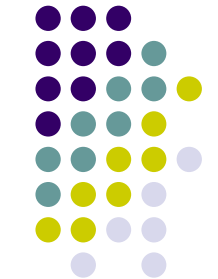
Implementation of EDGI



- Linux kernel 2.4.28
- Instrumented dentry cache code
- Added data structure: fsuid, refcnt, tainted, gh_list

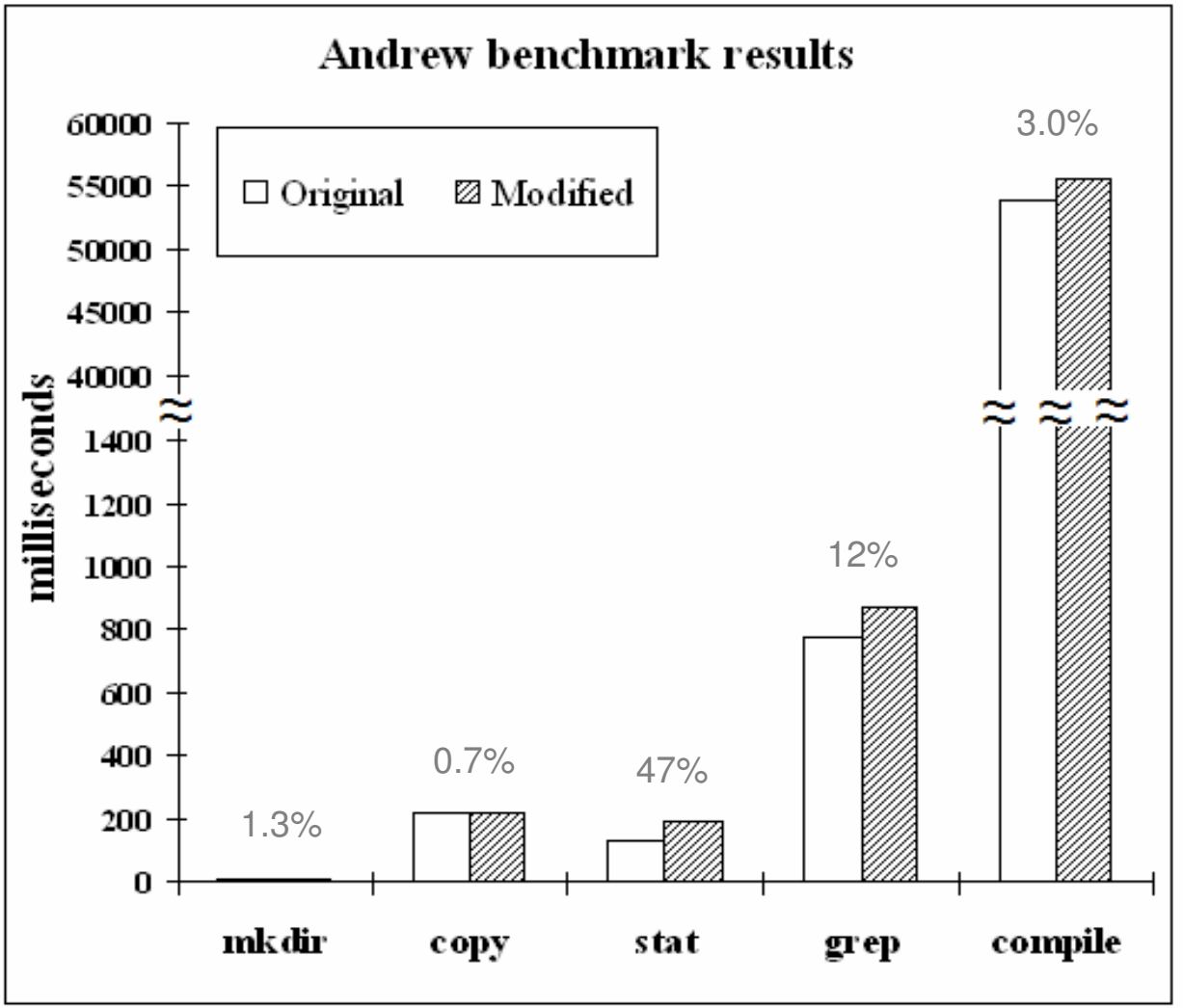
Source File	Modified Places	Original LOC	Added LOC
fs/dcache.c	4	1307	749
fs/namei.c	5	2047	84
fs/exec.c	1	1157	1
kernel/exit.c	1	602	1
kernel/fork.c	1	896	1

Evaluation of EDGI

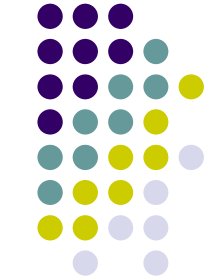


- False negatives: the completeness of CUU guarantees that EDGI has no false negatives
  - ☑ Prevents real attacks against logwatch, vi and emacs
- False positives
  - ❑ How to decide the timeout value?
- Overhead
  - ☑ Low

Overhead of EDGI

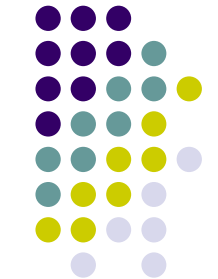


Other Potentially Effective Mechanisms to Prevent TOCTTOU Attacks



- RaceGuard (Crispin Cowan, et al. USENIX Security'01): <**stat, open**> for temporary file creation
- Probabilistic approach (Dean and Hu, USENIX Security'04): <**access, open**> ➔ <**access, open**, **access, open**, …>
  - Counter-attack (Nikita Borisov, et al. USENIX Security'05):
- Pseudo-transactions (Tsyrklevich and Yee, USENIX Security'03)

Conclusion



- The CUU model of TOCTTOU vulnerabilities
- EDGI design: maintaining the invariants
- EDGI implementation: ECA rules
- EDGI Evaluation: No false negatives, low false positives and low overhead, on changes to existing or future applications