MPI: Multiple Perspective Attack *Investigation* with Semantic Aware Execution Partitioning

Shiqing Ma, Juan Zhai, Fei Wang, Kyu Hyung Lee, Xiangyu Zhang, Dongyan Xu

PURDUE UNIVERSITY

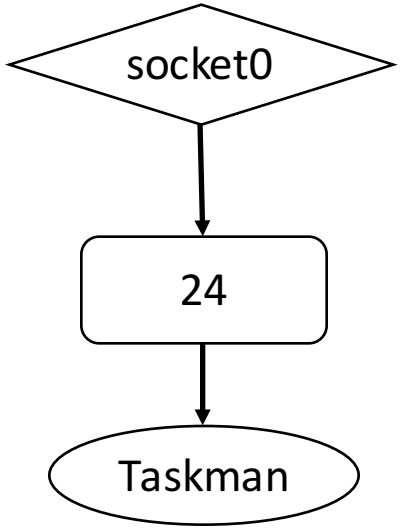
NANJING UNIVERSITY

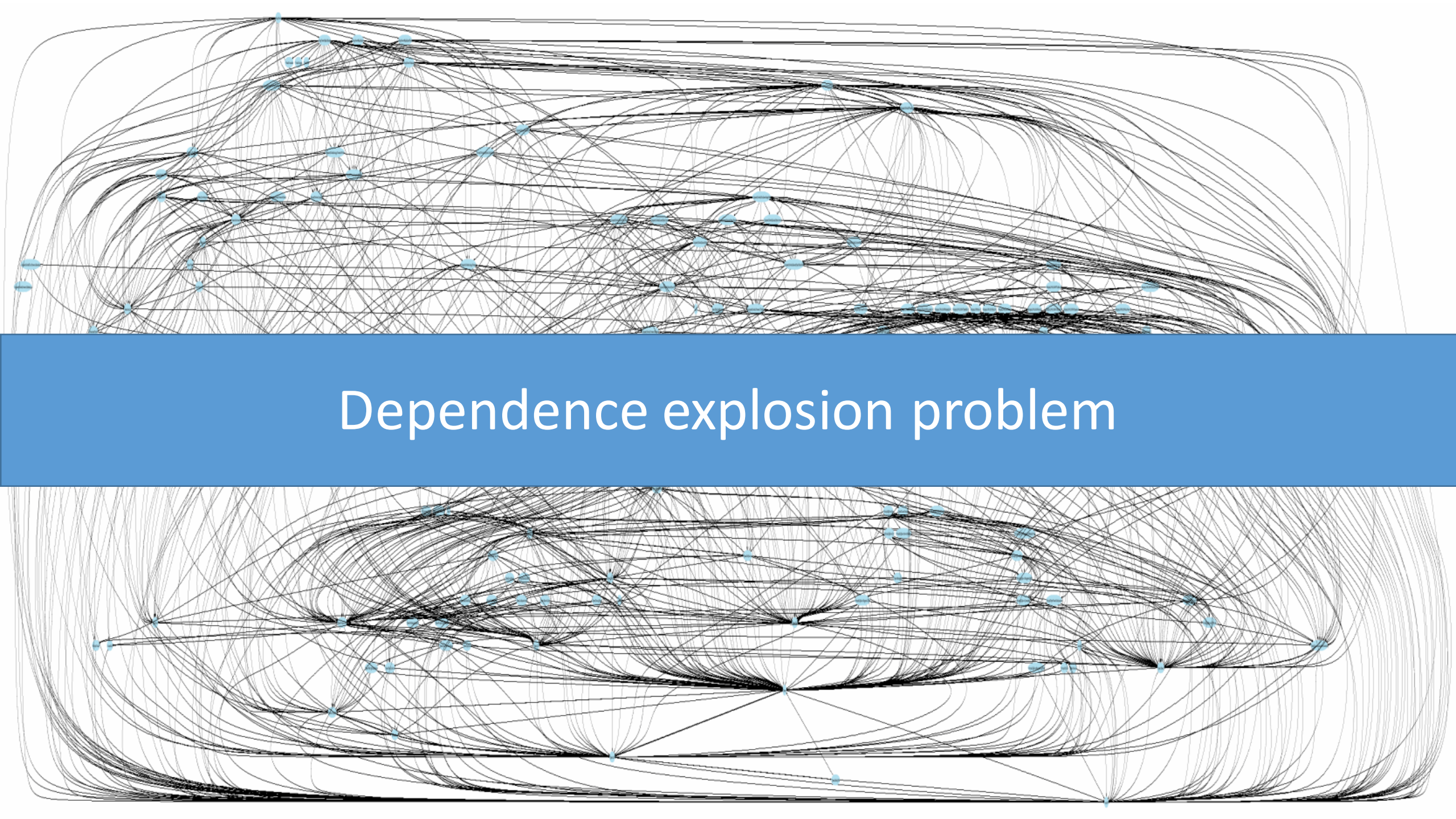
UNIVERSITY OF GEORGIA

Attack Investigation

- Provenance tracking system
  - Logging activities
- Dependence analysis
  - Analyzing activities

1. ……
2. *PID=24*, Receives from *socket0*
3. *PID=24*, Writes to File *Taskman*
4. ……

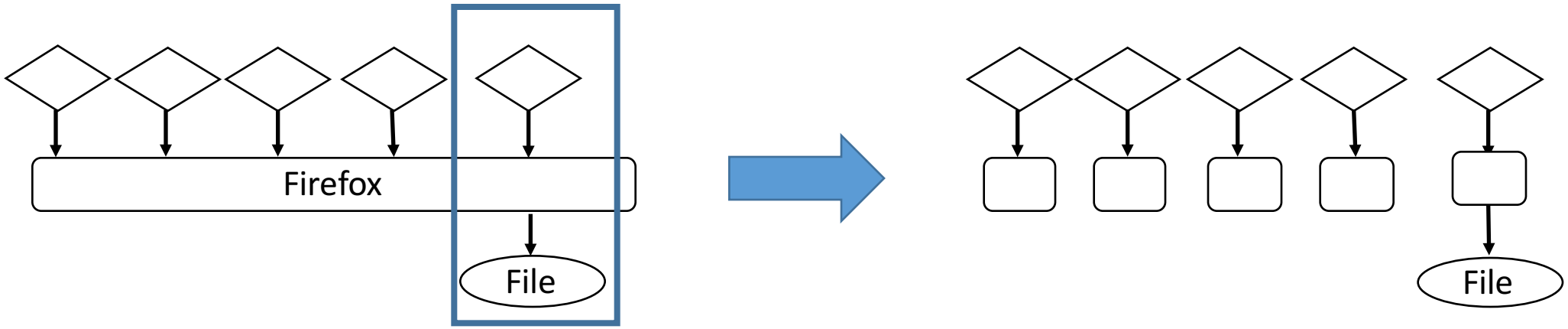




Dependence explosion problem

Dependence Explosion Problems

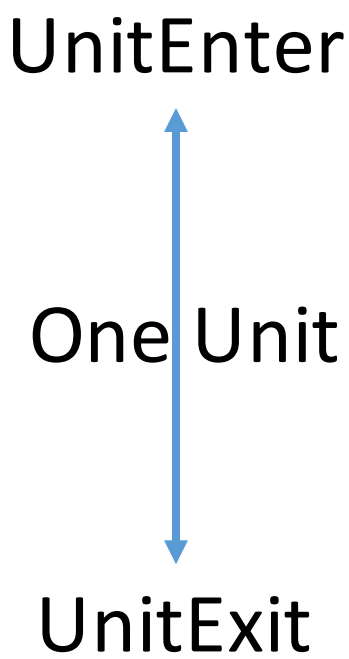
- Assumption: One write event depends on all read events that happen before it (the same process).
- Solution: Execution partitioning
  - Partition a process into execution units
  - Events are causality related within one unit, independent across units
  - Inter-unit dependence



Existing Execution Partitioning Technique

- BEEP: Event handling loop based
  - One iteration is one execution unit

```
while(True) {
  e = fetch_pending_event();
  switch(e) {
    case KeyPress:
      ...
      break;
    case MouseLeftPress:
      ...
      break;
    ...
  }
}
```



BEEP Limitations

- Low level events
  - Mouse click
- Training for inter-unit dependences
  - Very heavy process
- Excessive Units
  - Mouse movement events

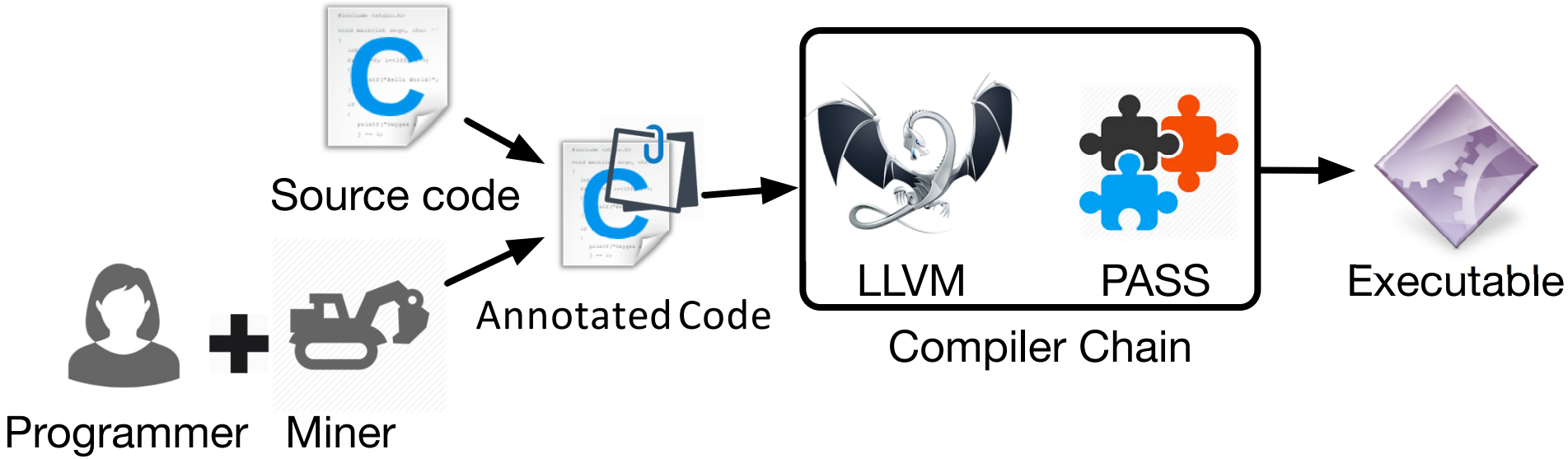
Desired

- Meaningful execution units
  - Tab
- Multiple Perspectives
  - E.g. tabs v.s. pages
  - No training
- Less units
  - Drop units if possible

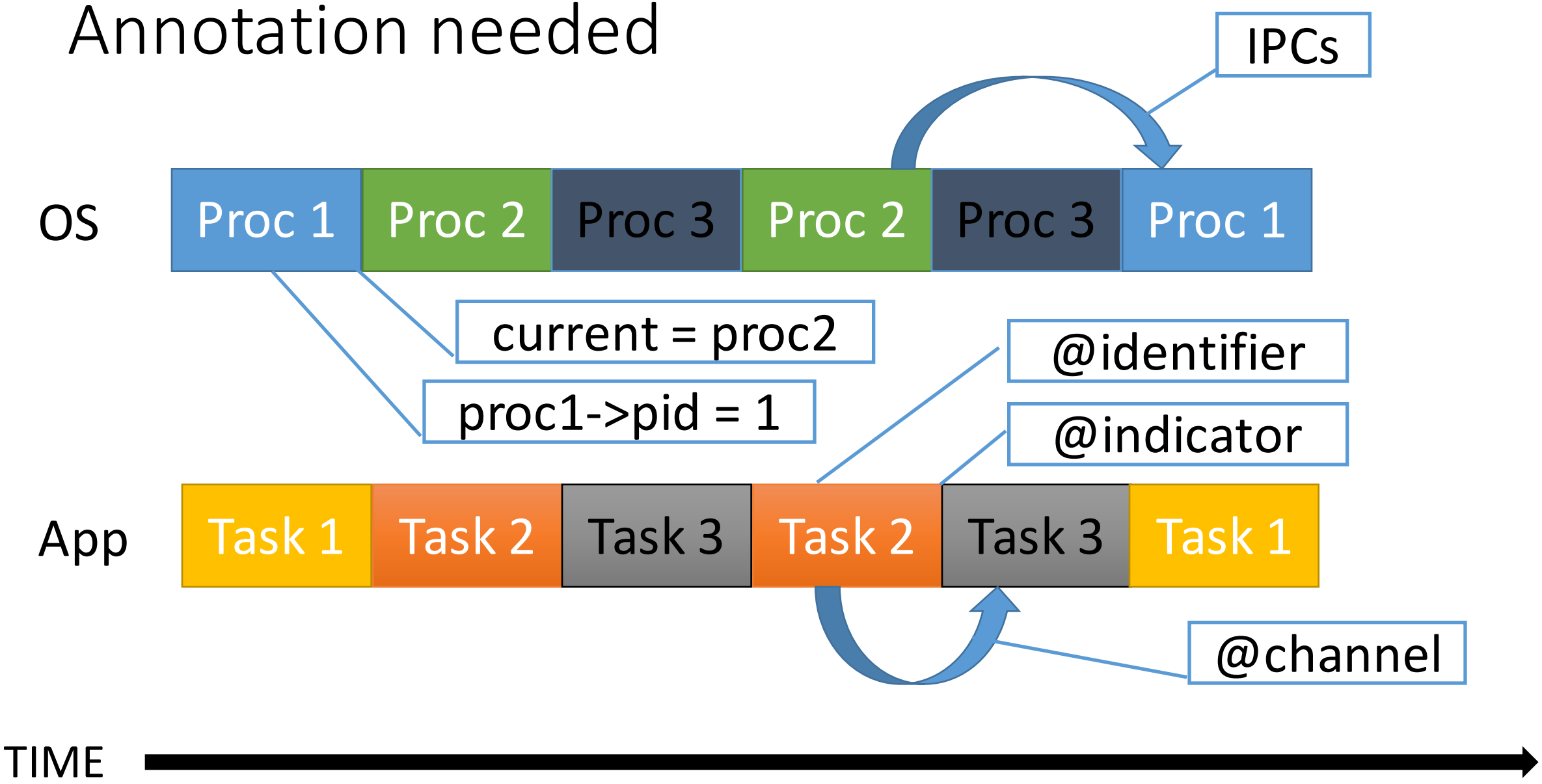
MPI Overview

- Inspired by process isolation mechanisms in operating systems
- **IDEA**: Execution partitioning based on user-defined *Tasks*
  - Task: represented by data structures
- Different tasks indicate different perspectives
  - Firefox: Tabs, Pages

MPI Workflow



Annotation needed

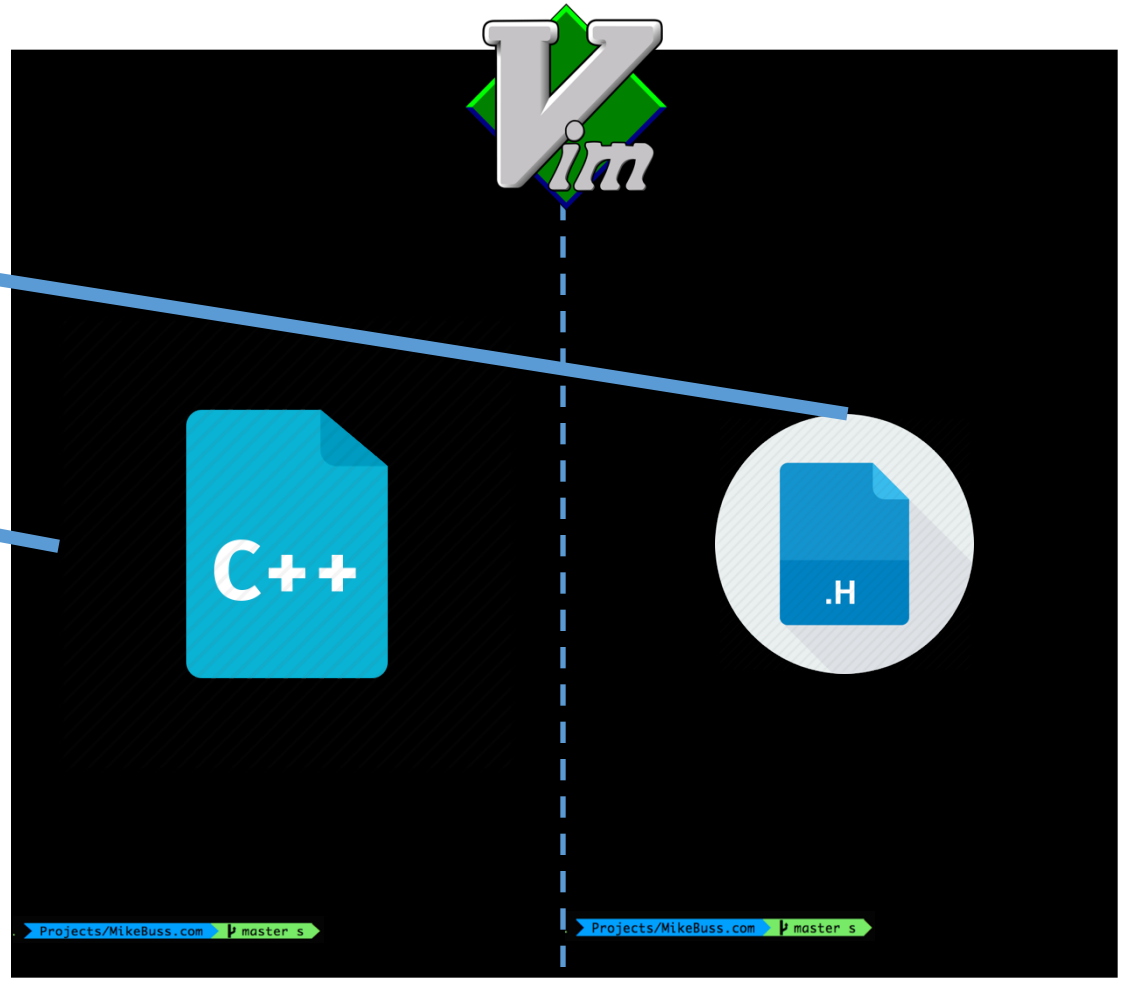


Annotating Vim

```
struct buf_T {
  @identifier
  int buf_id;
  char* name;
  . . .
};
```

```
@indicator
buf_T* curbuf;
```

```
@channel
struct yankreg *y_current;
```



Annotation Example

```
do_ecmd(. . .){
   . . .
   //a new buffer
   curbuf = buf;
   curbuf->name = . . .
}
```

```
if (curbuf != oldbuf) {
   oldbuf = curbuf;
   expose(curbuf->buf_id);
}
```

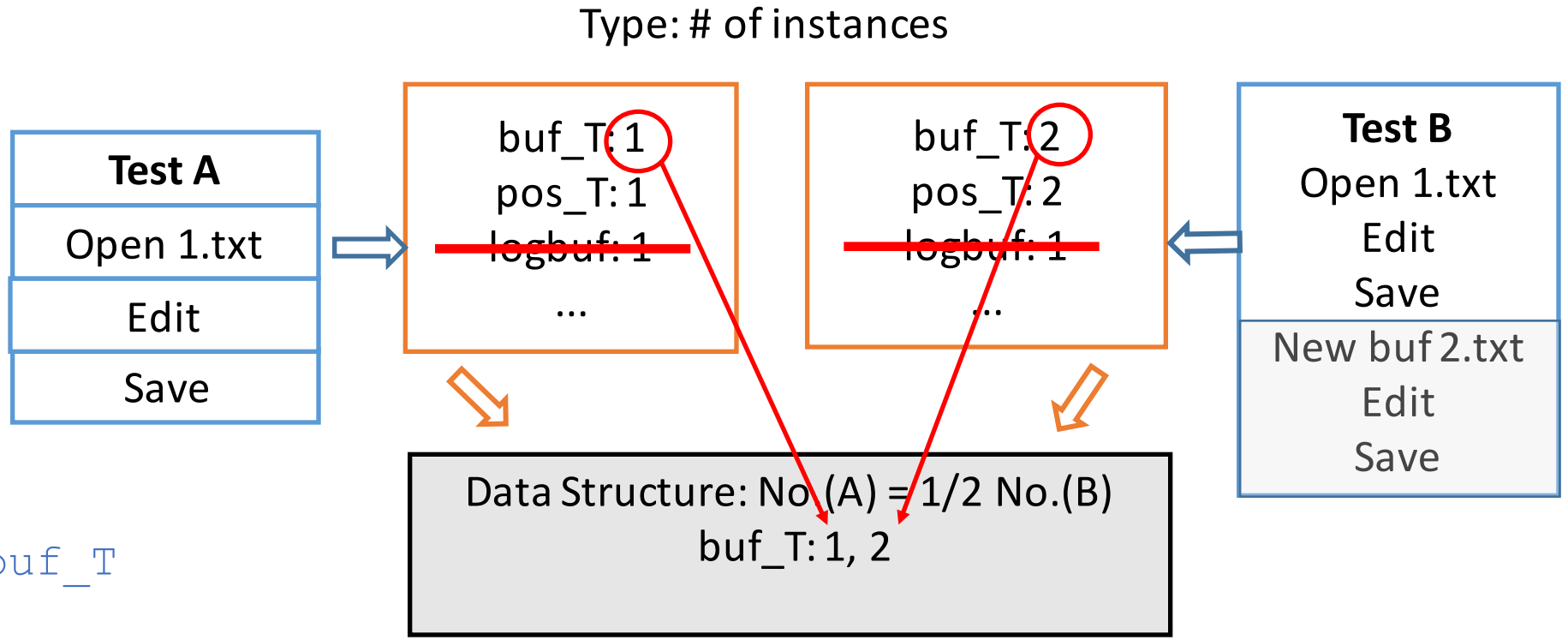
```
expose(id){
   kill(-100, id);//for linux audit system
}
```

Annotation miner

- Tasks can be represented by data structures
  - *buf_T* in Vim

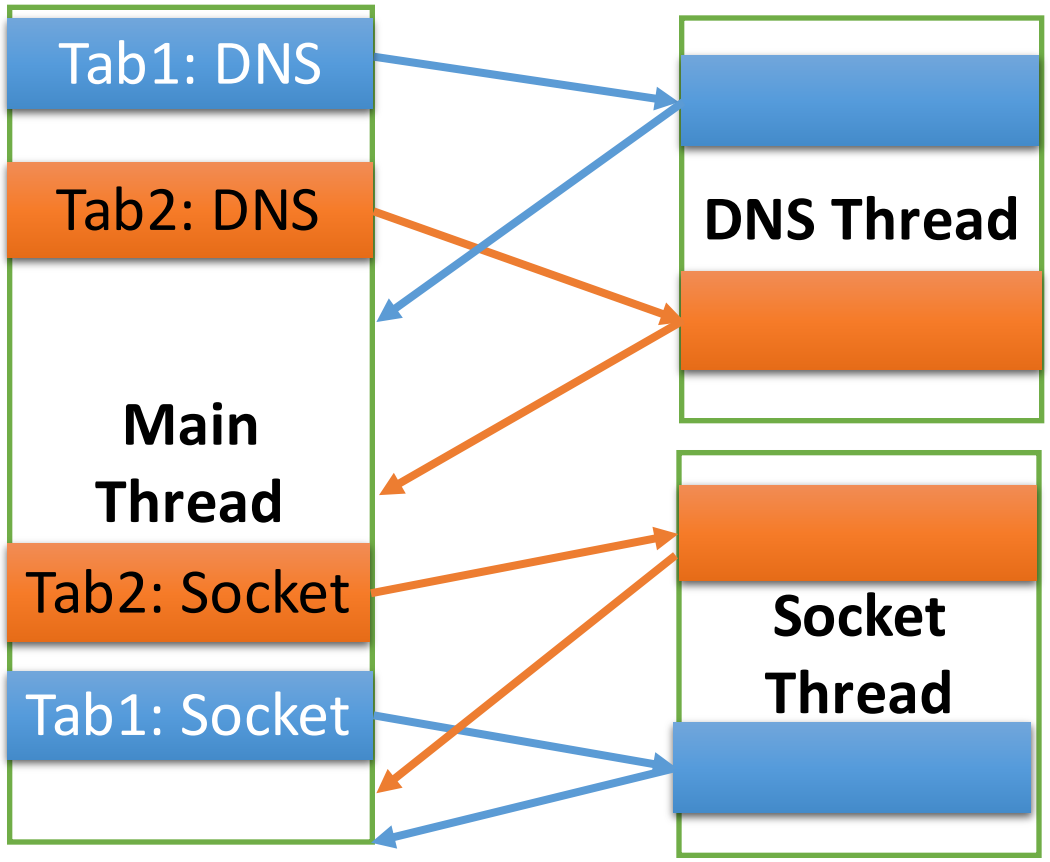
- Annotation miner is used to help developers find such data structures

Annotation Miner




```
struct buf_T
{
    /* position */
    pos_T pos;
}
```

Thread Model



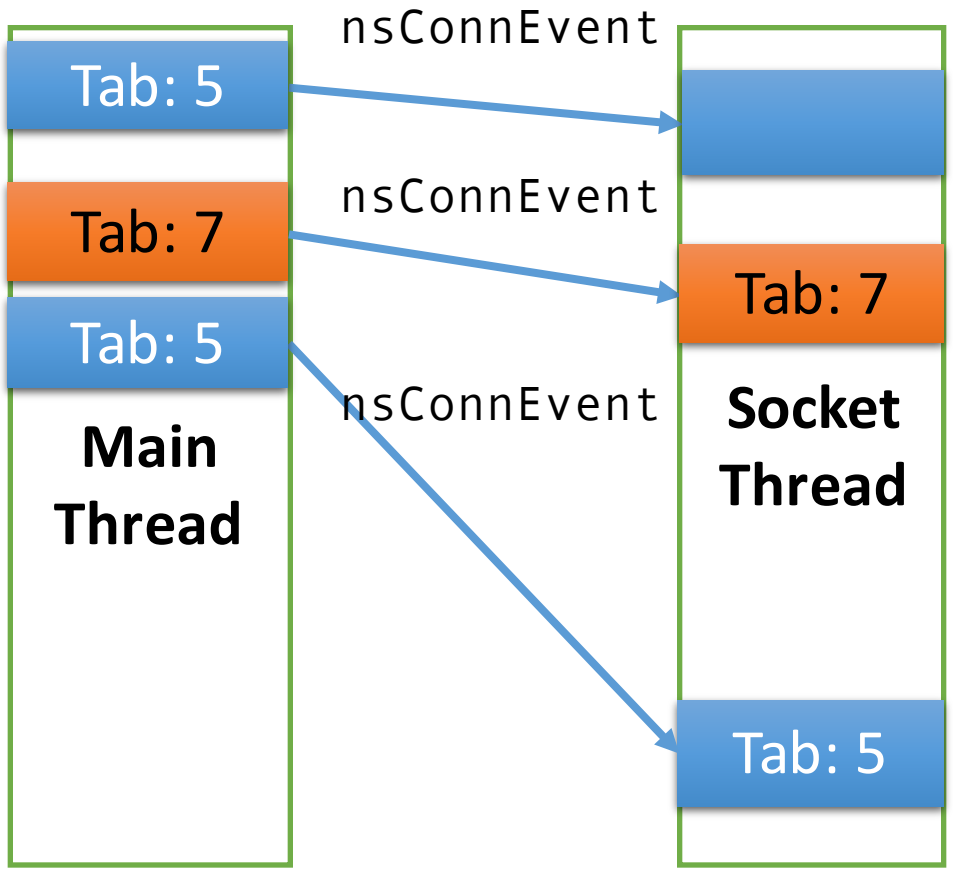
- Worker Threads
  - Working on the same type of tasks
  - Tasks in this thread are *sub-tasks*
  - One thread is serving for multiple *top-level tasks*
- *@delegator*
  - Add one field for top-leve task id
  - Inherit task identifiers from the top-level task

Annotating Firefox

```
@delegator
calss nsConnEvent {};

PostEvent(...) {
    event = new nsConnEvent(...);
    rv = target->Dispatch(event);
}
```

```
event->global_id = current_id;
```



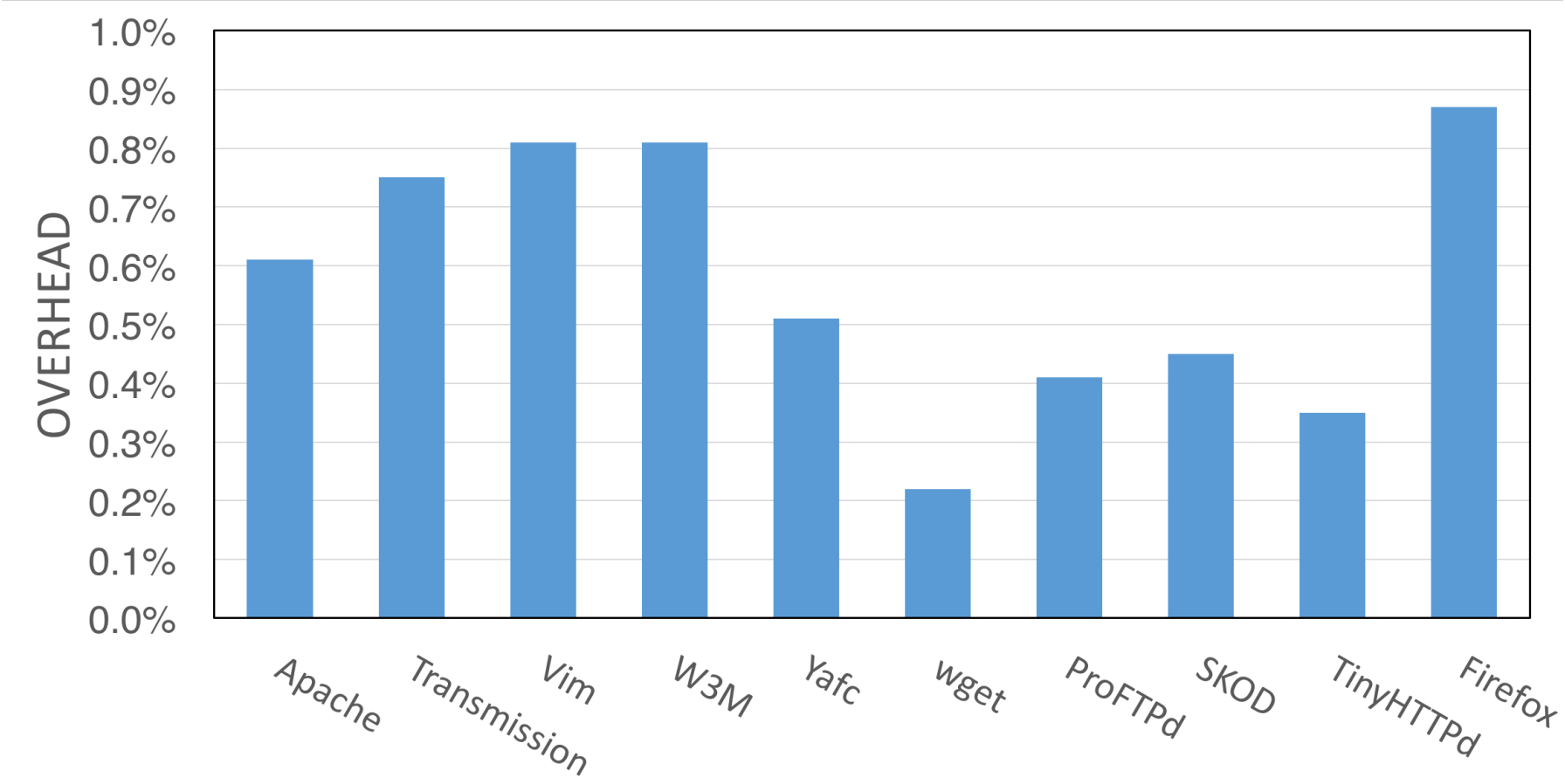
Evaluation: Annotations

Application	LOC	Annotation				Inst
		ID	IND	Chann	DEL	
Vim	313,283	3	3	2	0	878
Yafc	22,823	2	3	0	1	111
Firefox	8,073,181	3	32	0	1	6,867
TuxPaint	41,682	2	2	0	0	121
Pine	353,665	2	2	2	0	746
Apache	168,801	2	2	0	1	2,437
MC	135,668	2	2	1	0	3,332
ProFTPd	307,050	3	3	0	1	4,905
Transmission	111,903	2	4	0	1	66
W3M	67,291	2	2	0	1	3,718

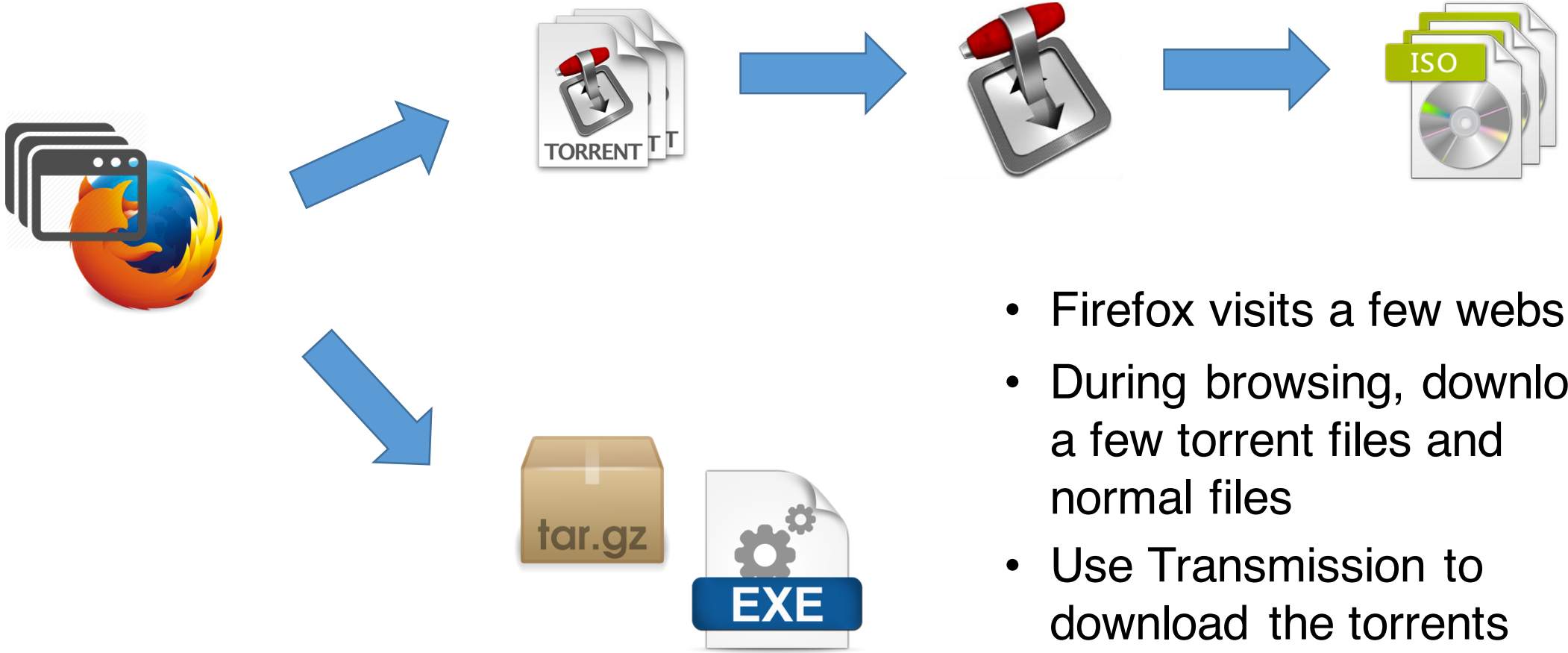
Evaluation: Space Overhead

Applicatoin	Perspective	BEEP-Audit	BEEP-HiFi	MPI-Audit	MPI-HiFi
Apache	Connection	15.38%	12.87%	5.37%	3.75
Bash	Command	0.45%	0.34%	0.41%	0.34%
Firefox	Tab	42.16%	38.23%	18.20%	13.24%
Pine	Command	8.11%	6.09%	7.28%	4.09%
Vim	File	2.23%	2.32%	0.13%	0.13%

Evaluation: Runtime Overhead Caused By MPI

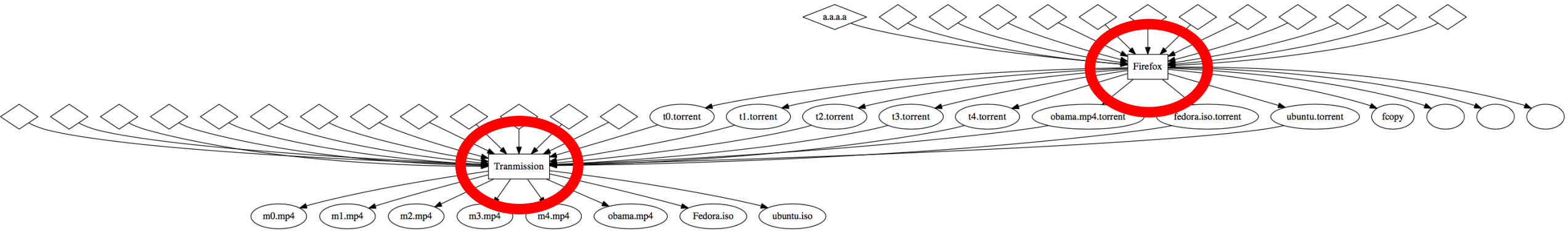


Case Study



- Firefox visits a few websites
- During browsing, download a few torrent files and normal files
- Use Transmission to download the torrents

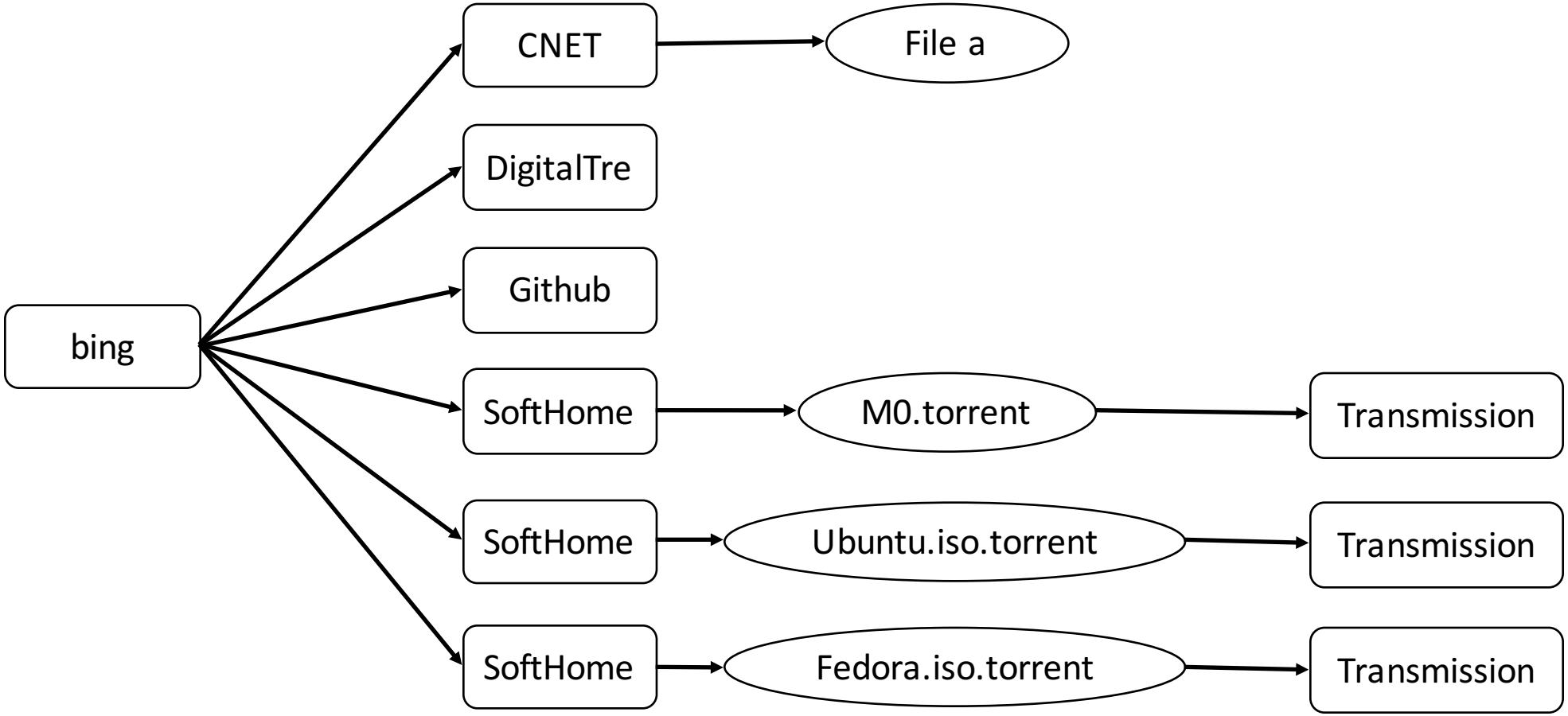
Traditional Solution



BEEP (Only Show the Firefox Part)

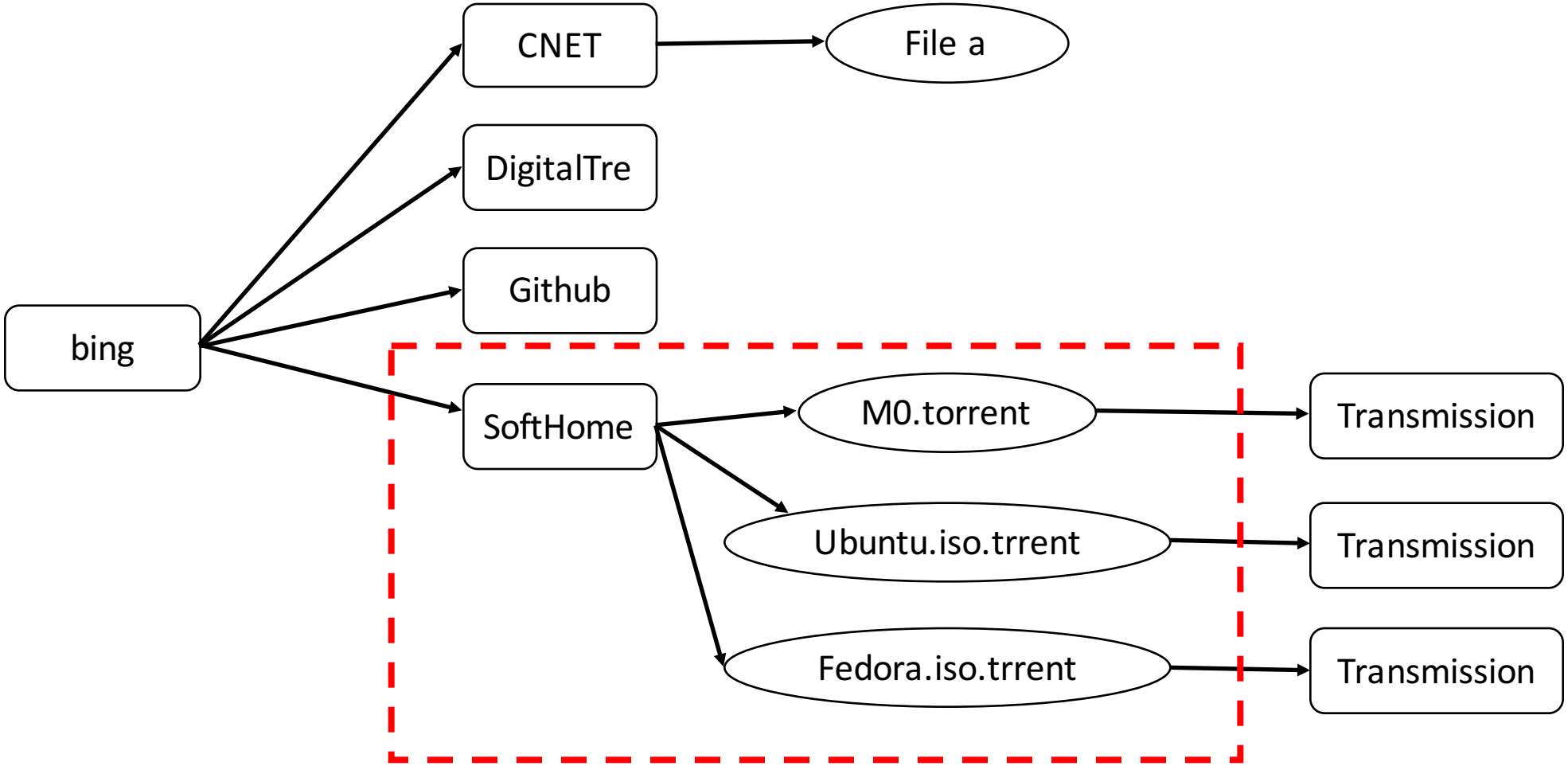
MPI

(Firefox partitioned by tab and Transmission partitioned by input file)



MPI

(Firefox partitioned by website and Transmission partitioned by input file)



Related Work

- Provenance System
  - Butler [Security ’15, ACSAC ’12], Lee [NDSS ’16, ACSAC ‘15, NDSS ’13], Xu [ICDCS ’06], Lara [SOSP ’05], King [NDSS ’05, SOSP ’03]
- Log storage
  - Lee [ACSAC ’15, CCS ’13], Butler [ACSAC ’12], Zhou [SOSP ’11]
- Log integrity:
  - Moyer [Security ’15], Sion [ICDCS ’08]

Limitations

- Source code
  - LLVM-based solution
- Annotation
  - Miner is helpful but not fully automated
  - Miner finds the data structure, not the @indicator/@identifier variables

Conclusion

- A new idea of execution partitioning based on data structures
  - User defined tasks
  - Multi-perspective, semantic-aware attack investigation
- Program analysis and runtime techniques to enable such partitioning
  - Small number of annotations
- Annotation miner
  - Help identify annotations
- Evaluation
  - Low overhead
  - Better investigation support

Thank you!