High Accuracy Attack Provenance via Binary-Based ExEcution Partition (BEEP)

Kyu Hyung Lee,
Xiangyu Zhang,
Dongyan Xu

NDSS 2013, Feb 25 - 27

PURDUE UNIVERSITY

CERIAS

Introduction

- Attack provenance is important
  - Determine the "entry point" of an attack
  - Understand the damage to the victim system

PURDUE UNIVERSITY

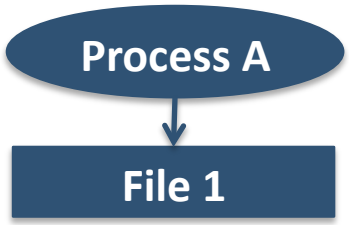
CERIAS

Introduction

- Previous approaches
  - Generate causal graph by detecting system level dependences

Introduction

- Previous approaches
  - Generate causal graph by detecting system level dependences
    - Process <-> File

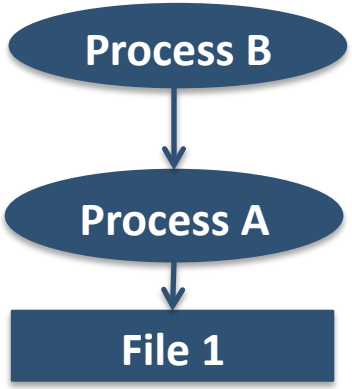


PURDUE UNIVERSITY

CERIAS

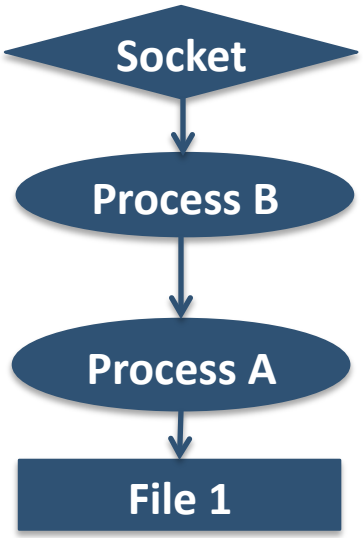
Introduction

- Previous approaches
  - Generate causal graph by detecting system level dependences
    - Process <-> File
    - Process -> Process



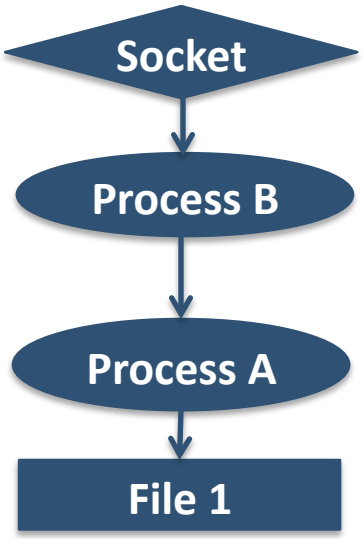
Introduction

- Previous approaches
  - Generate causal graph by detecting system level dependences
    - Process <-> File
    - Process -> Process
    - Process <-> Socket



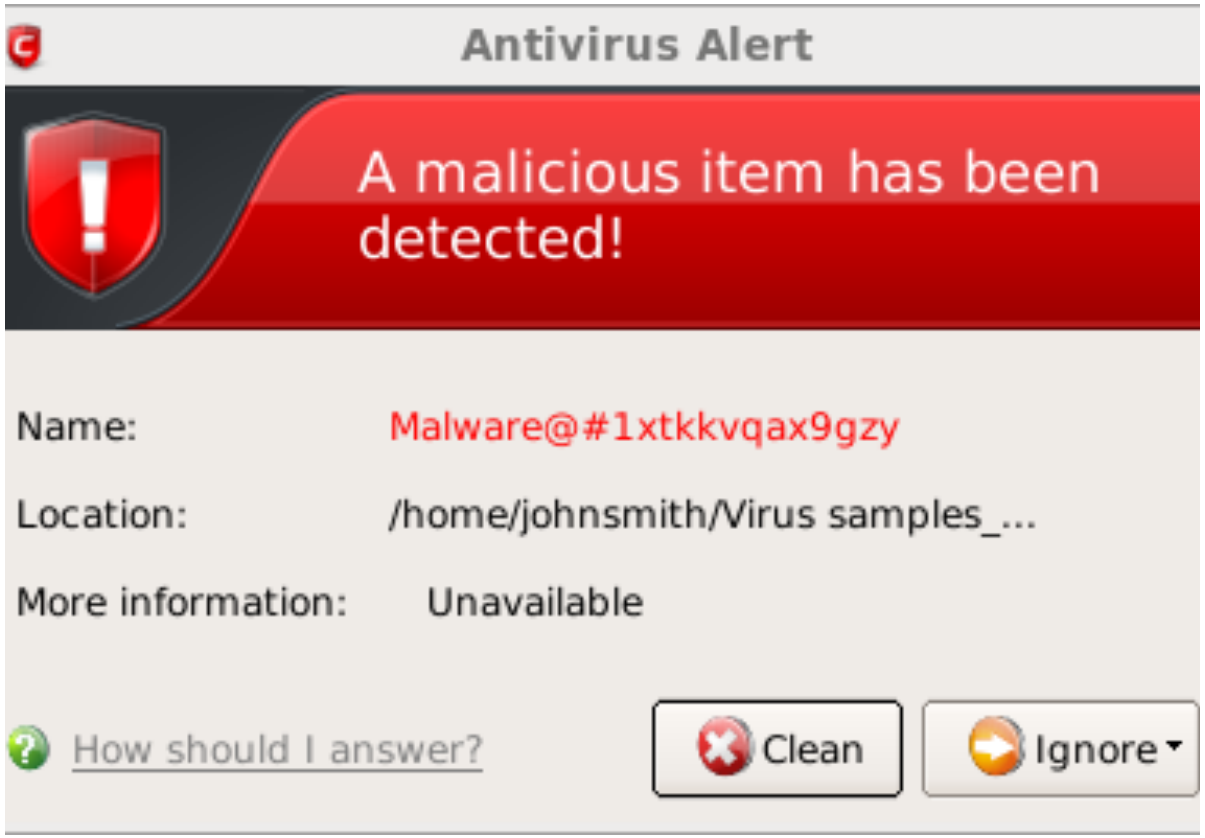
Introduction

- Previous approaches
  - Generate causal graph by detecting system level dependences
    - Process <-> File
    - Process -> Process
    - Process <-> Socket



  - Limitations
    - Dependence explosion problem
    - Granularity of "process" is too coarse

PURDUE UNIVERSITY CERIAS



PURDUE UNIVERSITY

CERIAS

Malware

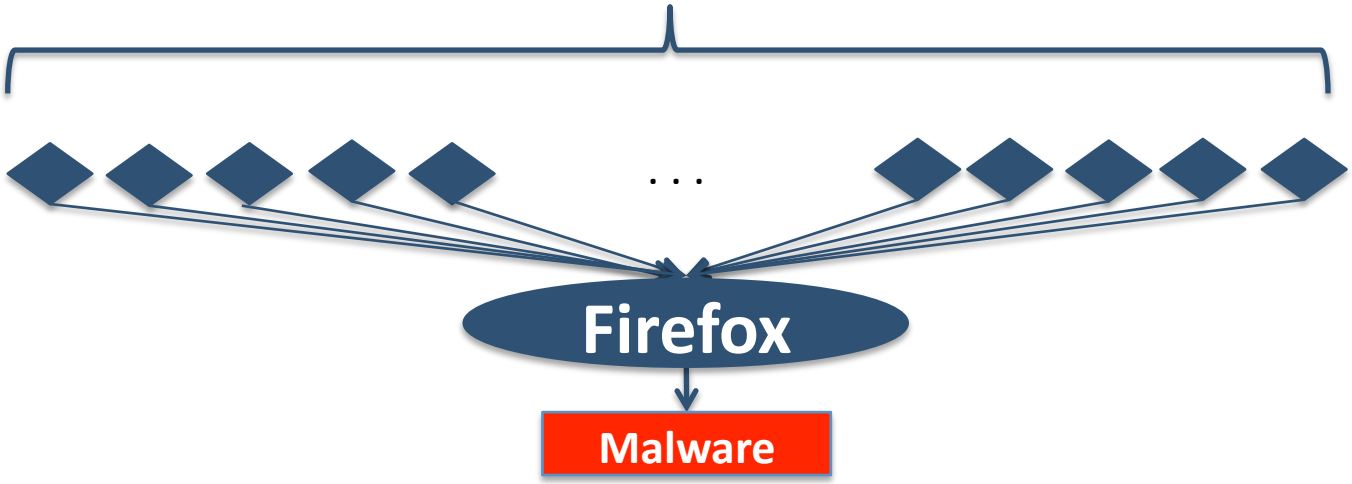
PURDUE
UNIVERSITY

CERIAS

The user visited 11 web sites

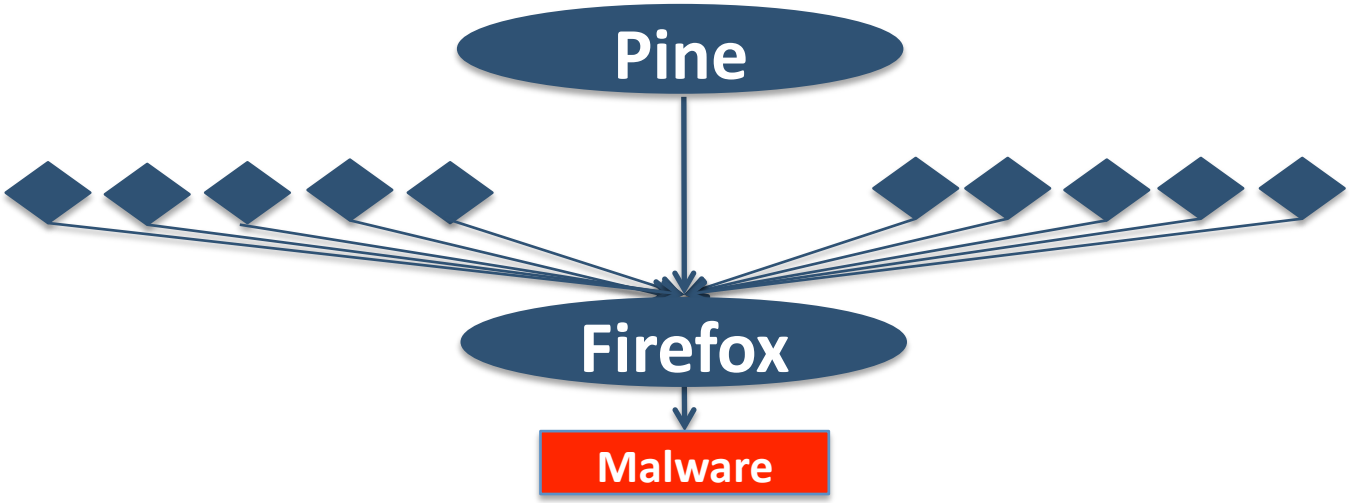
Dependence explosion!!

(229 IP Addresses)



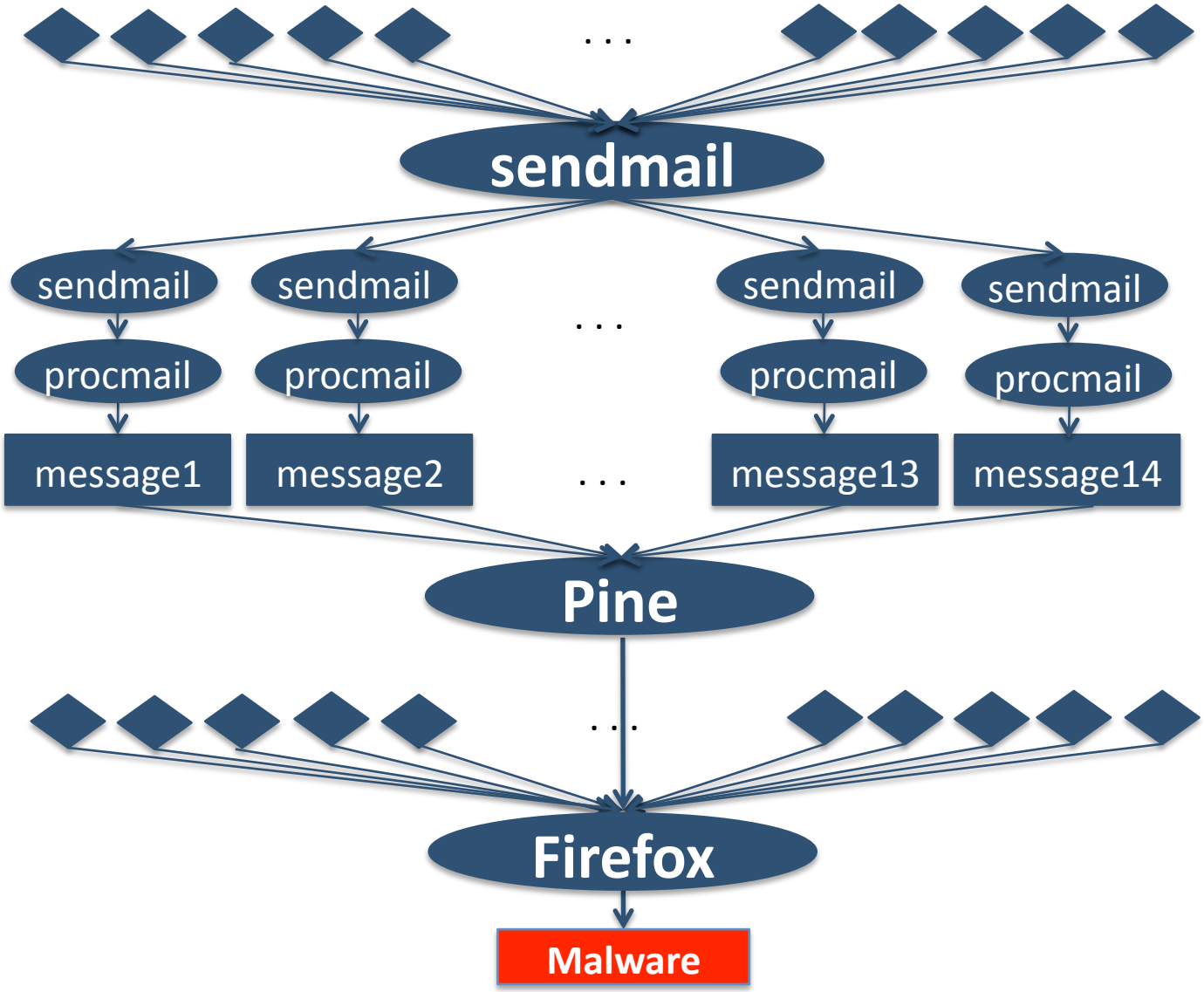
PURDUE UNIVERSITY

CERIAS



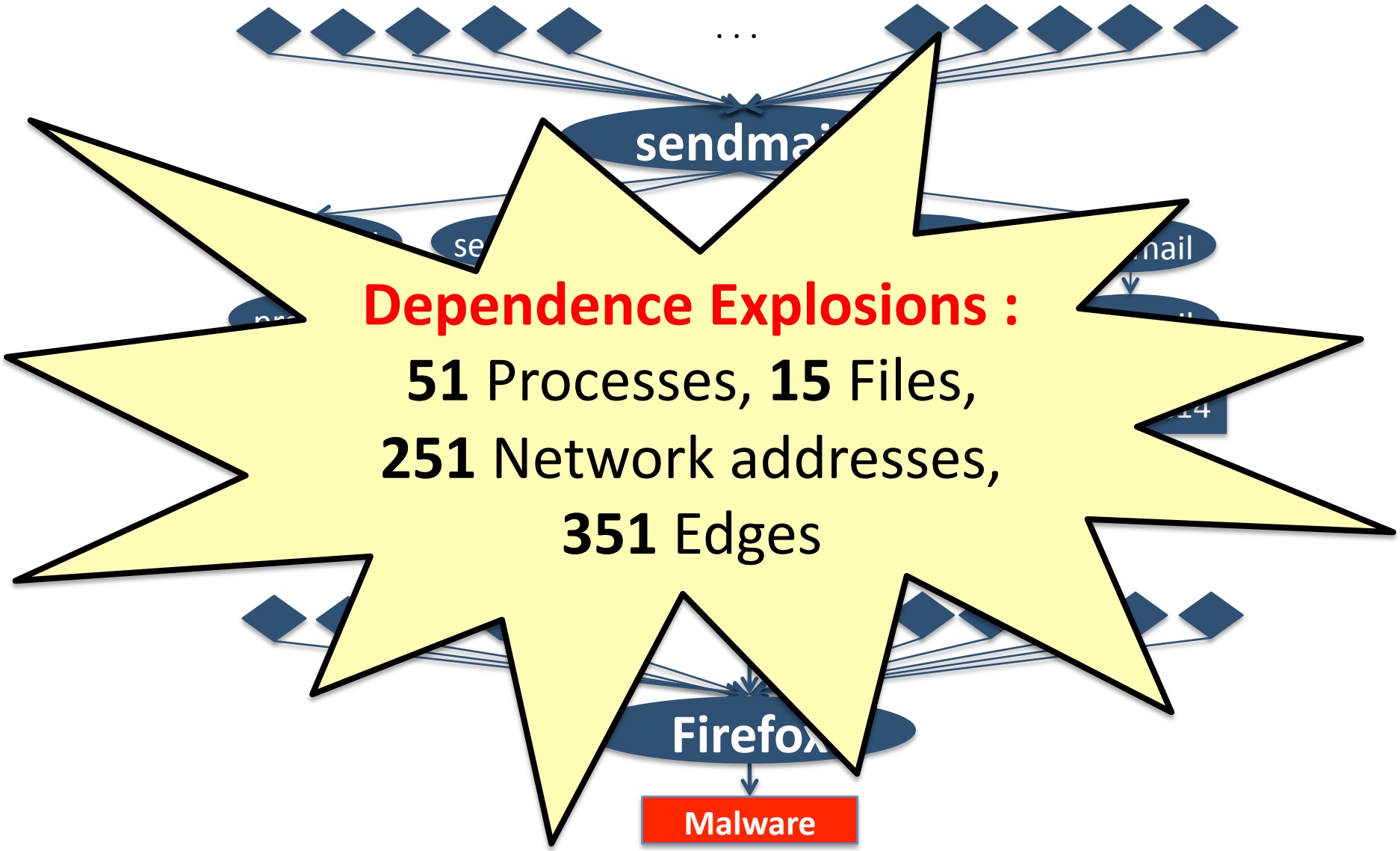
PURDUE UNIVERSITY

CERIAS



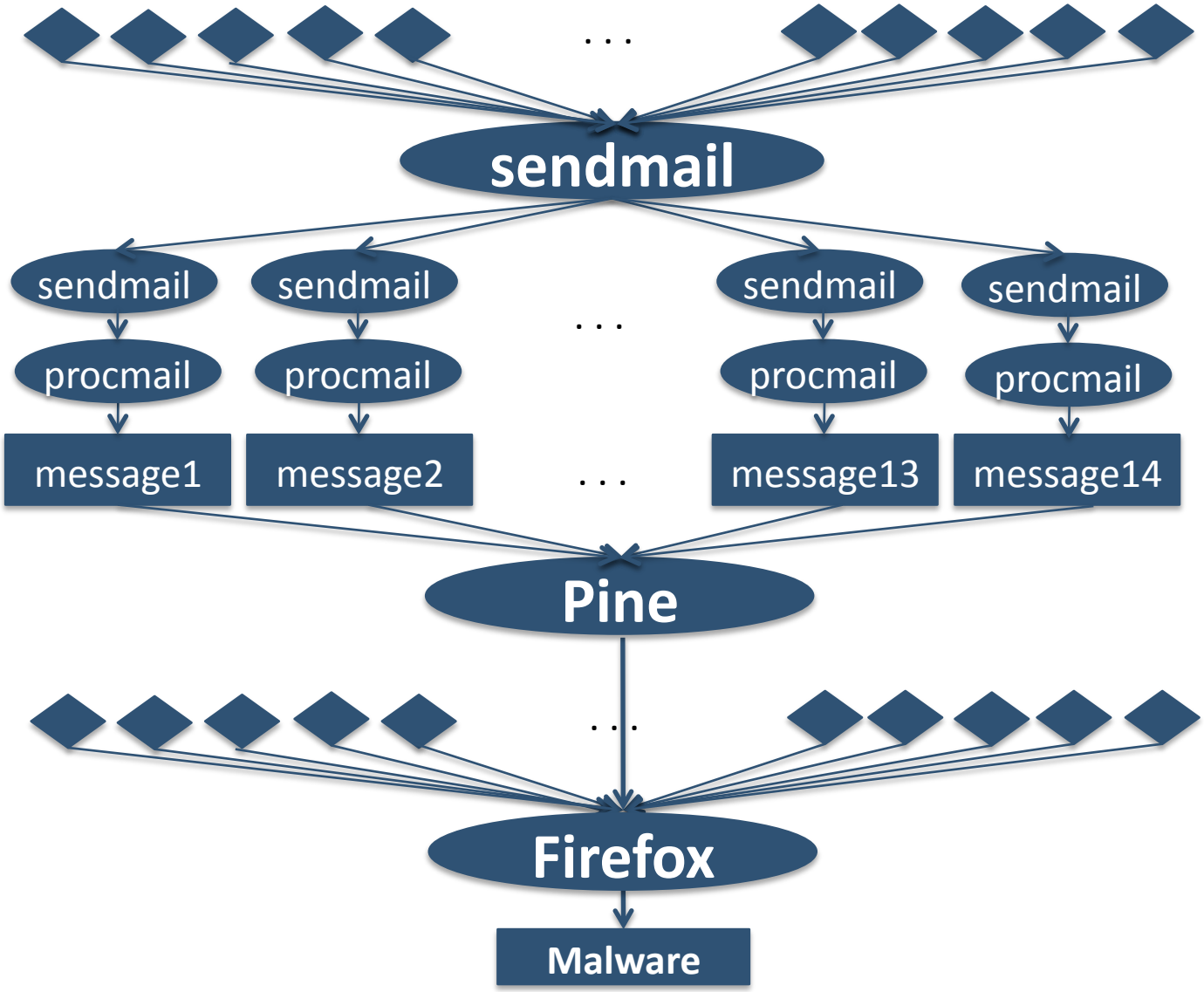
PURDUE UNIVERSITY

CERIAS



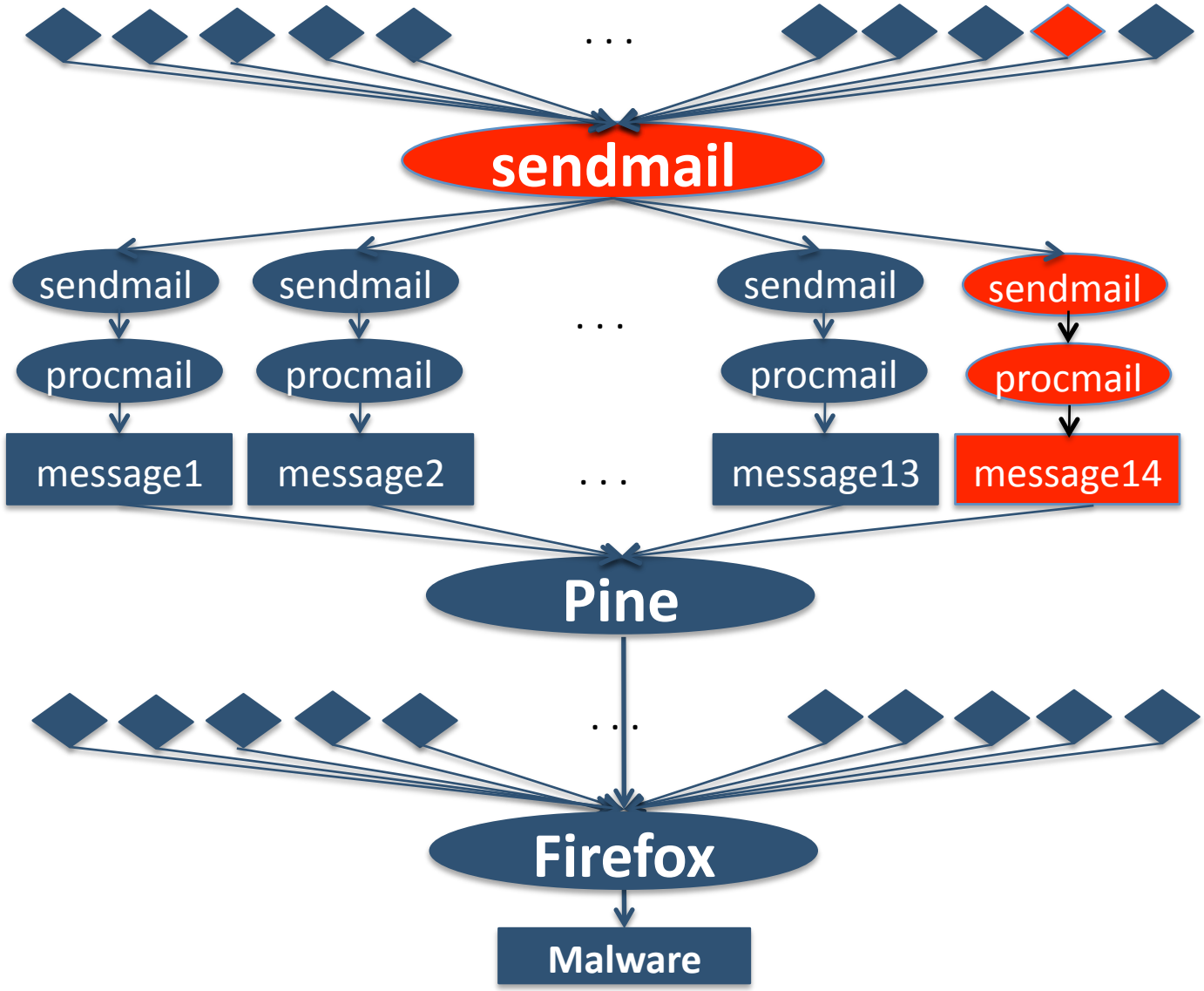
PURDUE UNIVERSITY

CERIAS



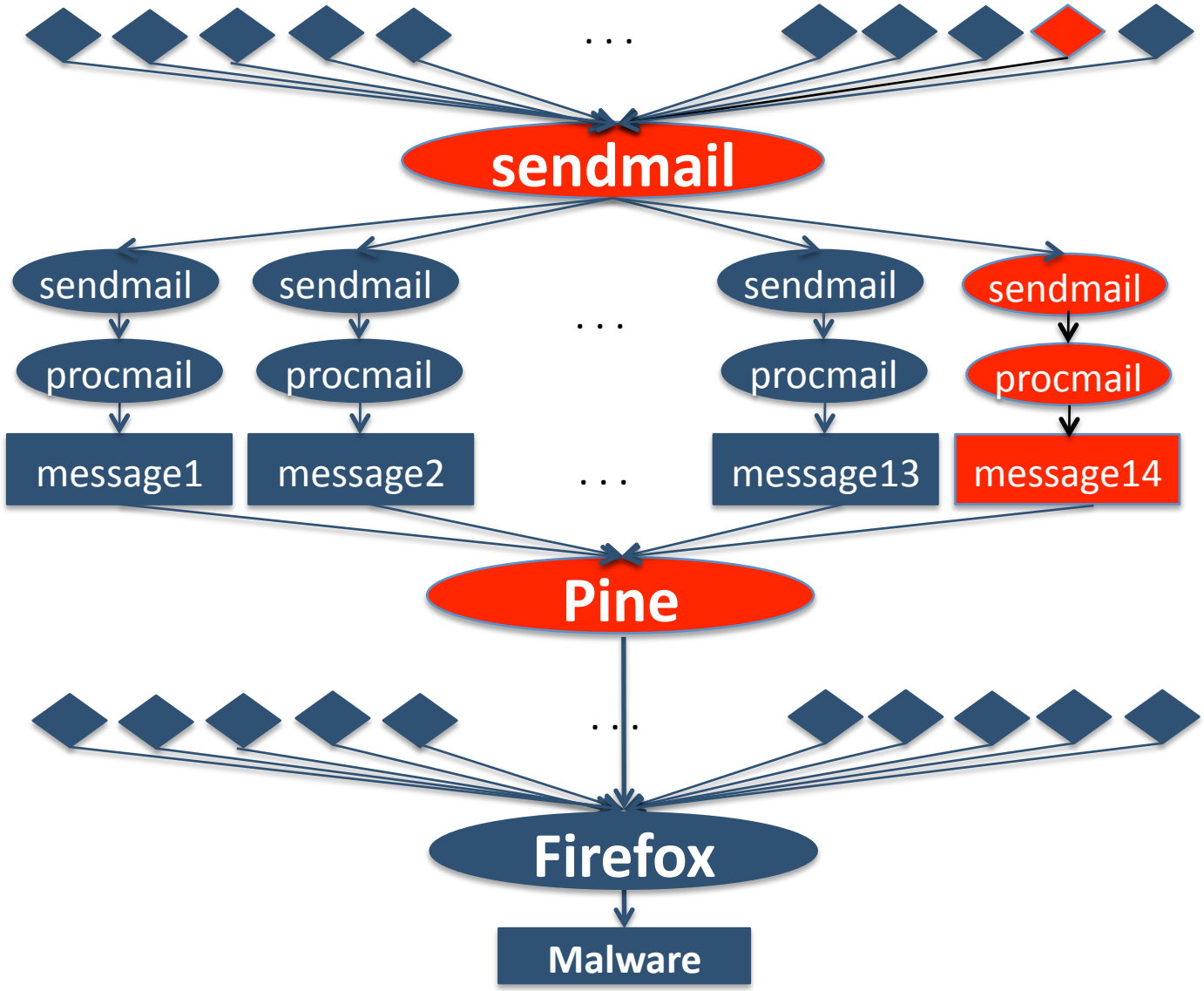
PURDUE UNIVERSITY

CERIAS



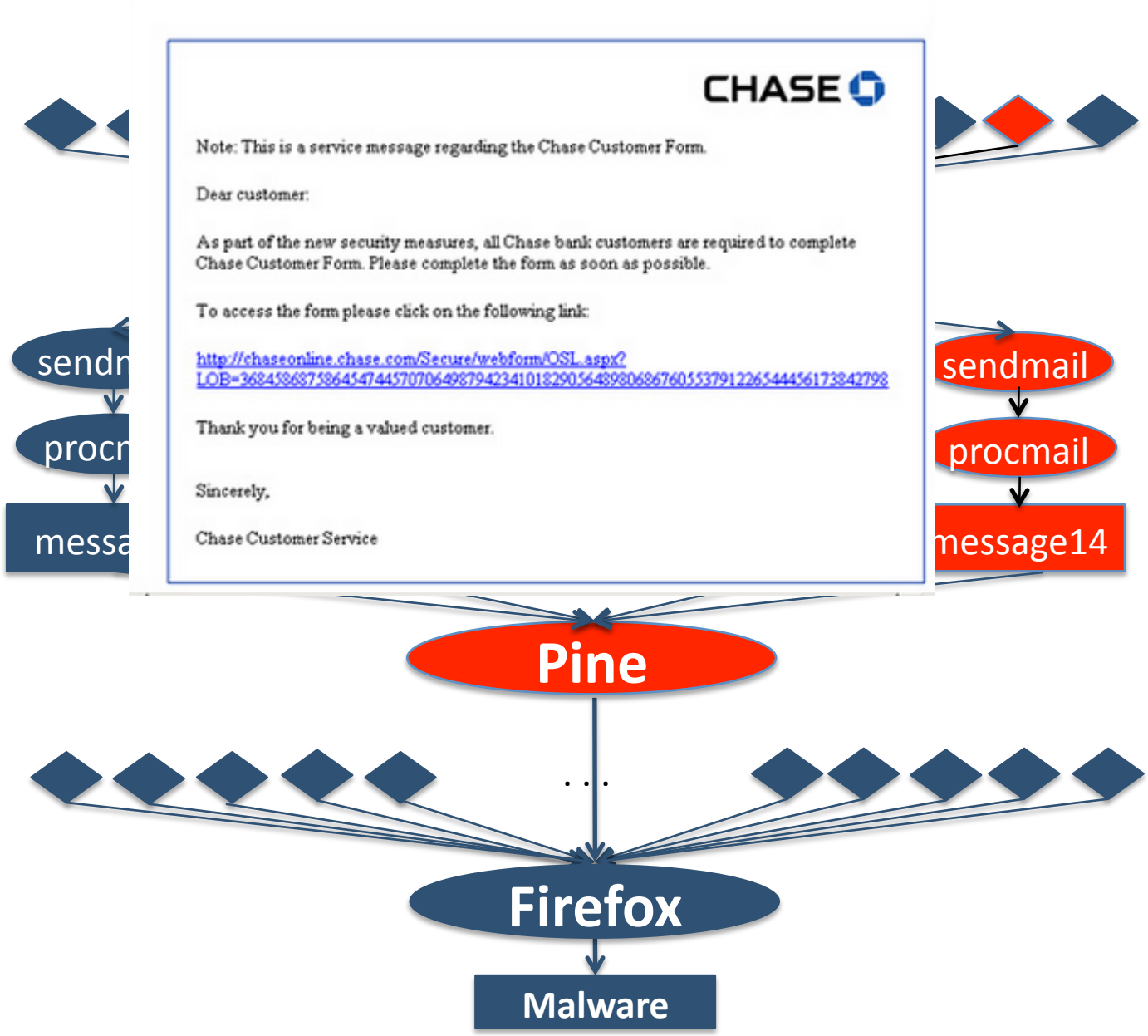
PURDUE UNIVERSITY

CERIAS



PURDUE UNIVERSITY

CERIAS

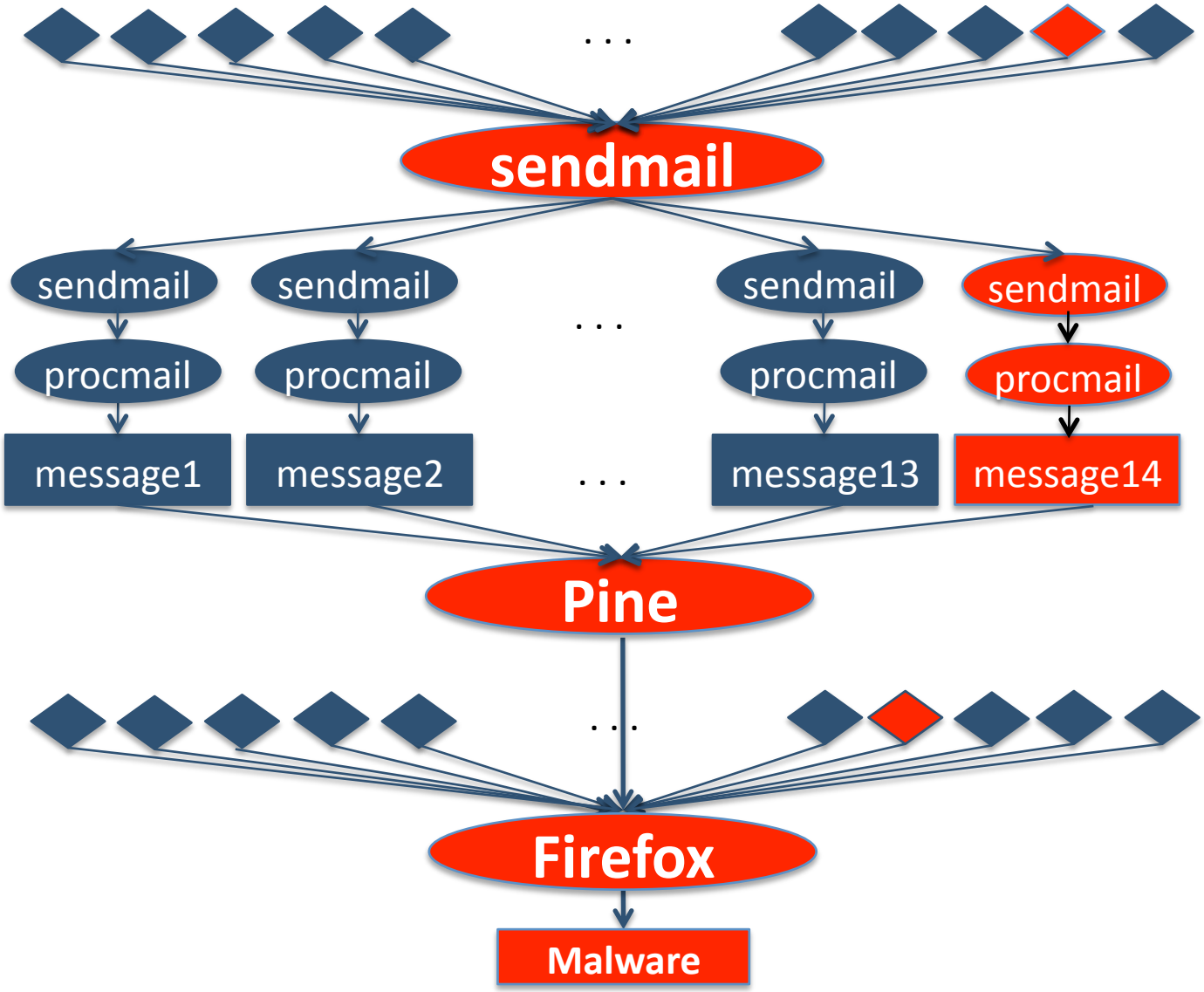


PURDUE UNIVERSITY

CERIAS

PURDUE UNIVERSITY

CERIAS

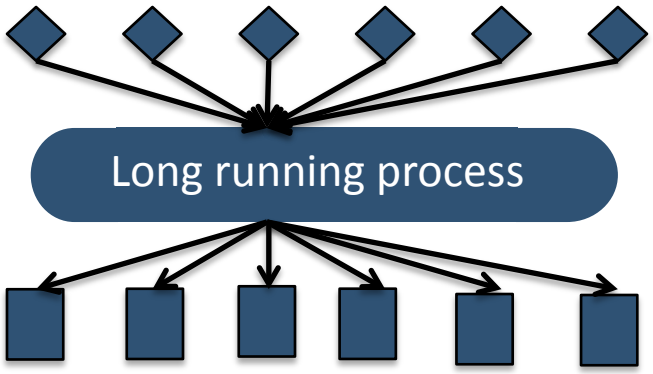


PURDUE UNIVERSITY

CERIAS

BEEP : Our Approach

- Finer-grained subject : Execution "UNIT"
  - Dynamically partition the execution of a process into **autonomous** execution segments

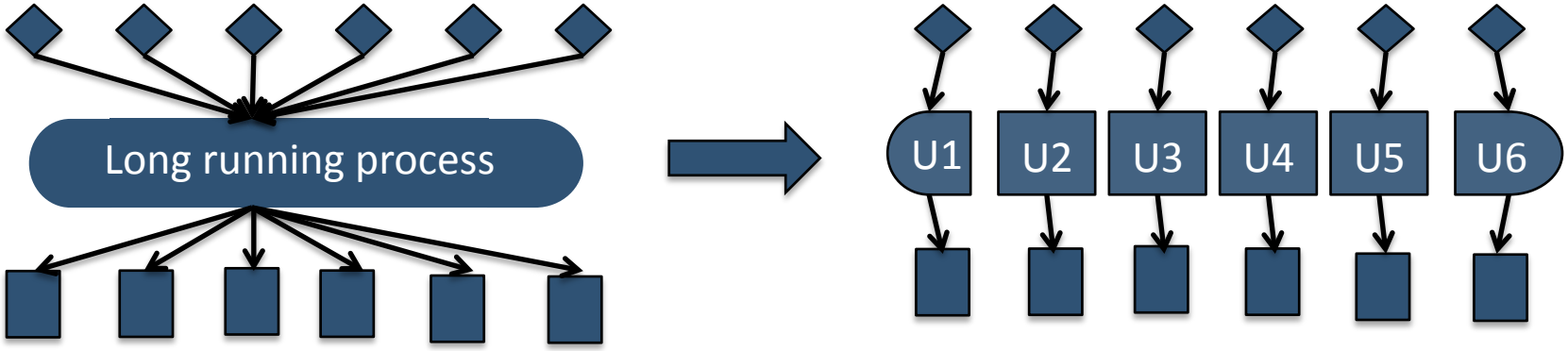


PURDUE UNIVERSITY

CERIAS

BEEP : Our Approach

- Finer-grained subject : Execution "UNIT"
  - Dynamically partition the execution of a process into **autonomous** execution segments

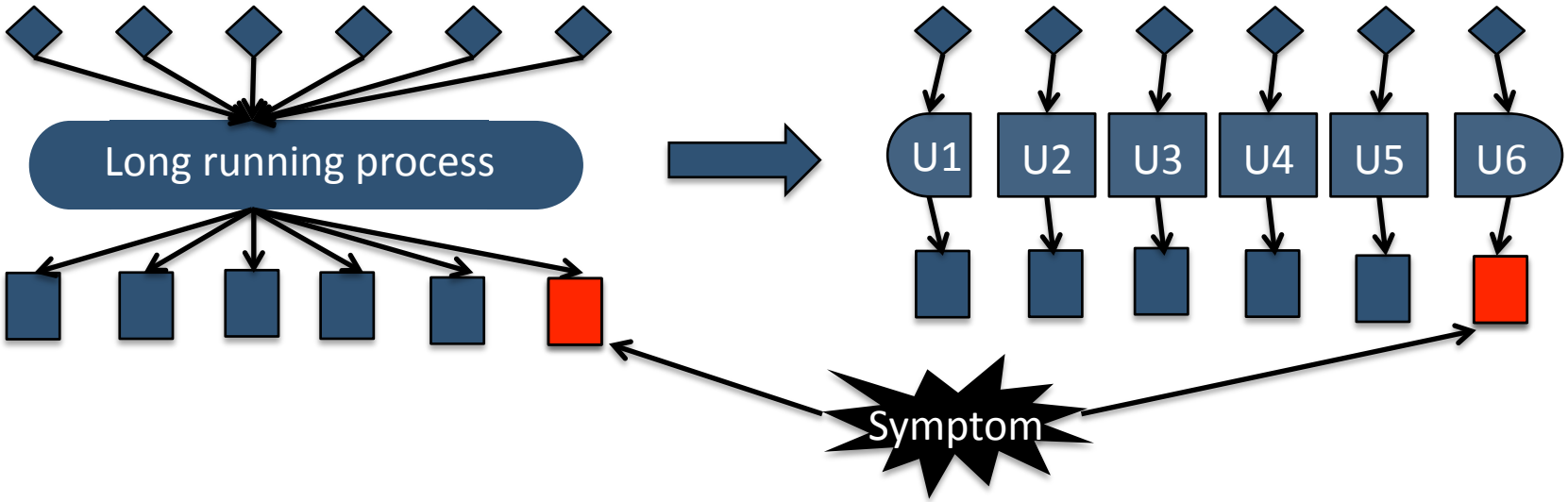


PURDUE UNIVERSITY

CERIAS

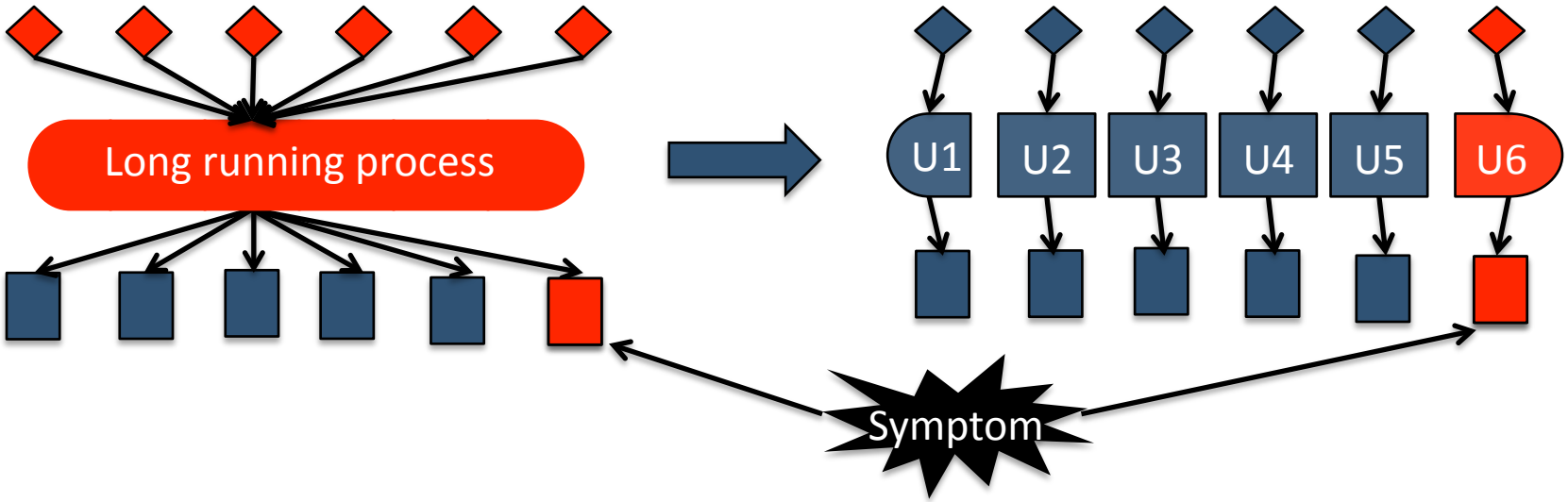
BEEP : Our Approach

- Finer-grained subject : Execution "UNIT"
  - Dynamically partition the execution of a process into **autonomous** execution segments



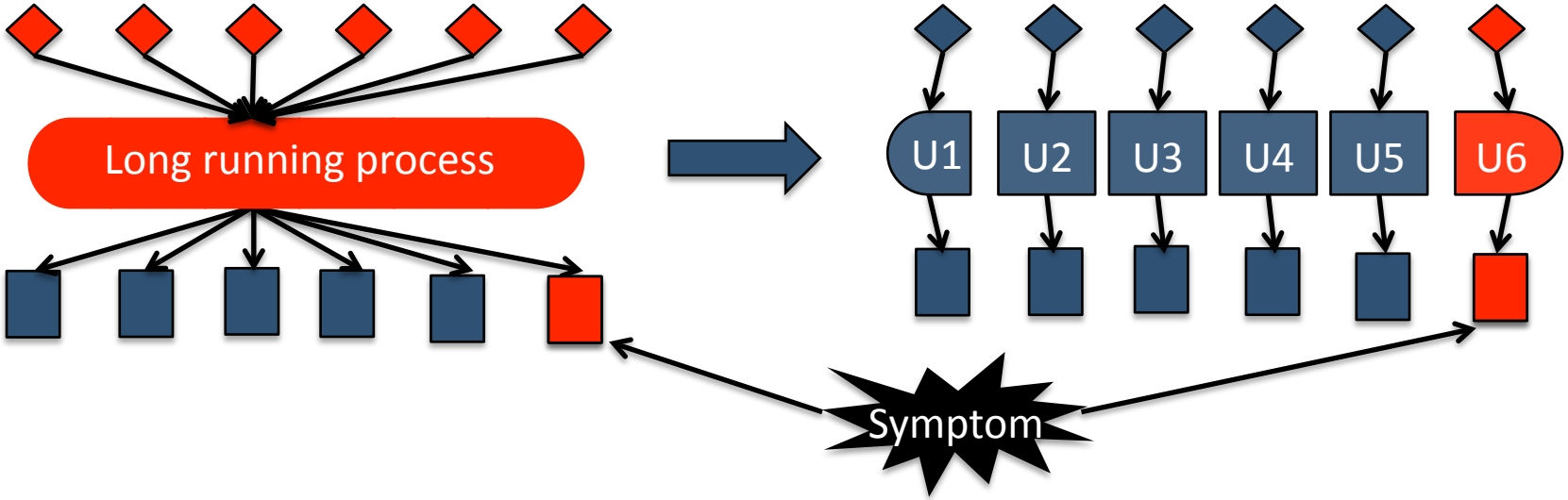
BEEP : Our Approach

- Finer-grained subject : Execution "UNIT"
  - Dynamically partition the execution of a process into **autonomous** execution segments



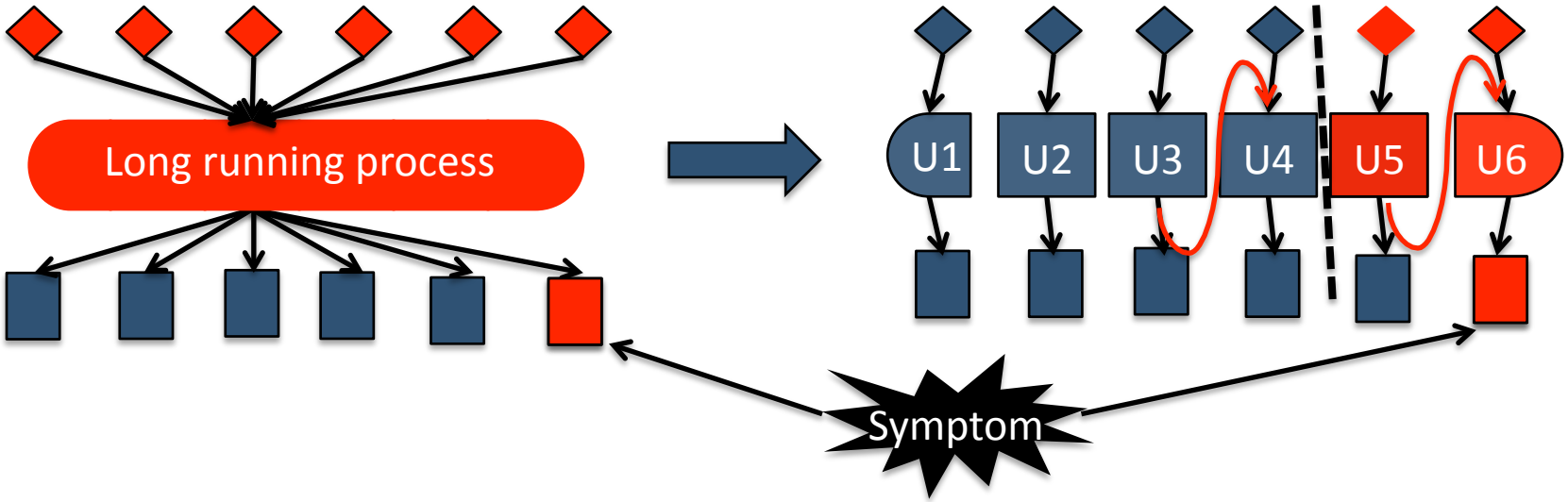
BEEP : Our Approach

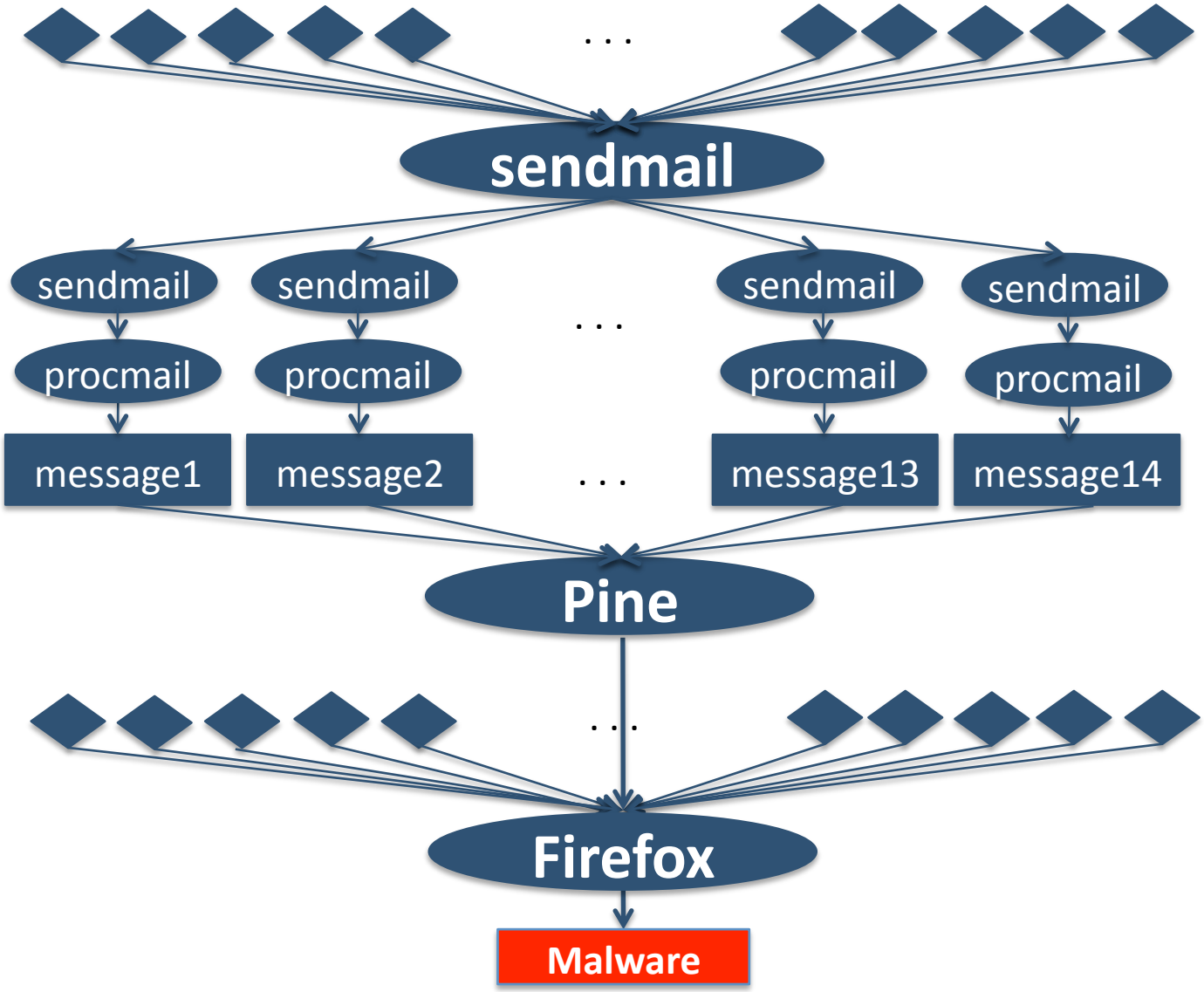
- Finer-grained subject : Execution "UNIT"
  - Dynamically partition the execution of a process into **autonomous** execution segments
  - Units are not always independent
    - –Detect causality between units



BEEP : Our Approach

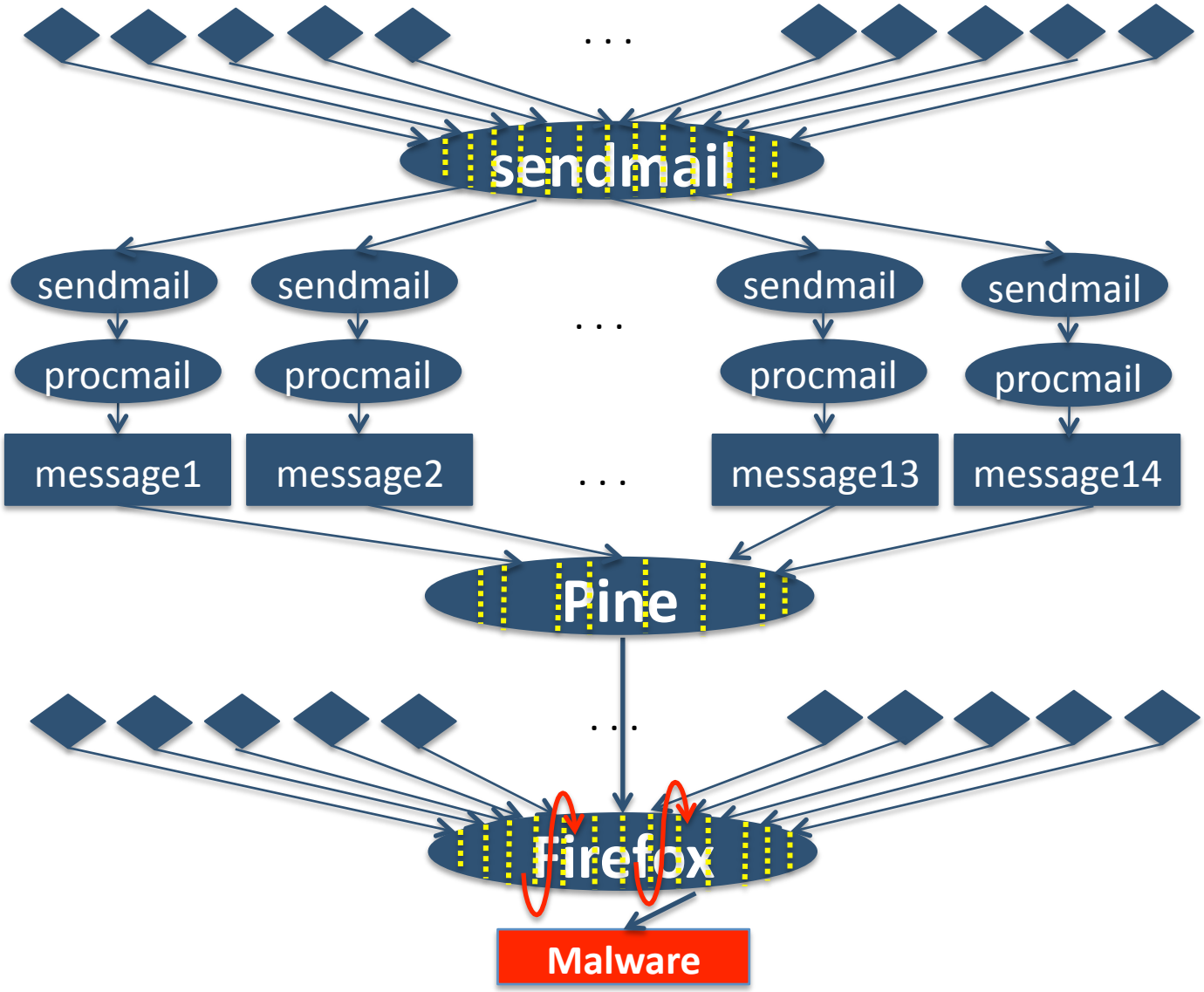
- Finer-grained subject : Execution "UNIT"
  - Dynamically partition the execution of a process into **autonomous** execution segments
  - Units are not always independent
    - Detect causality between units





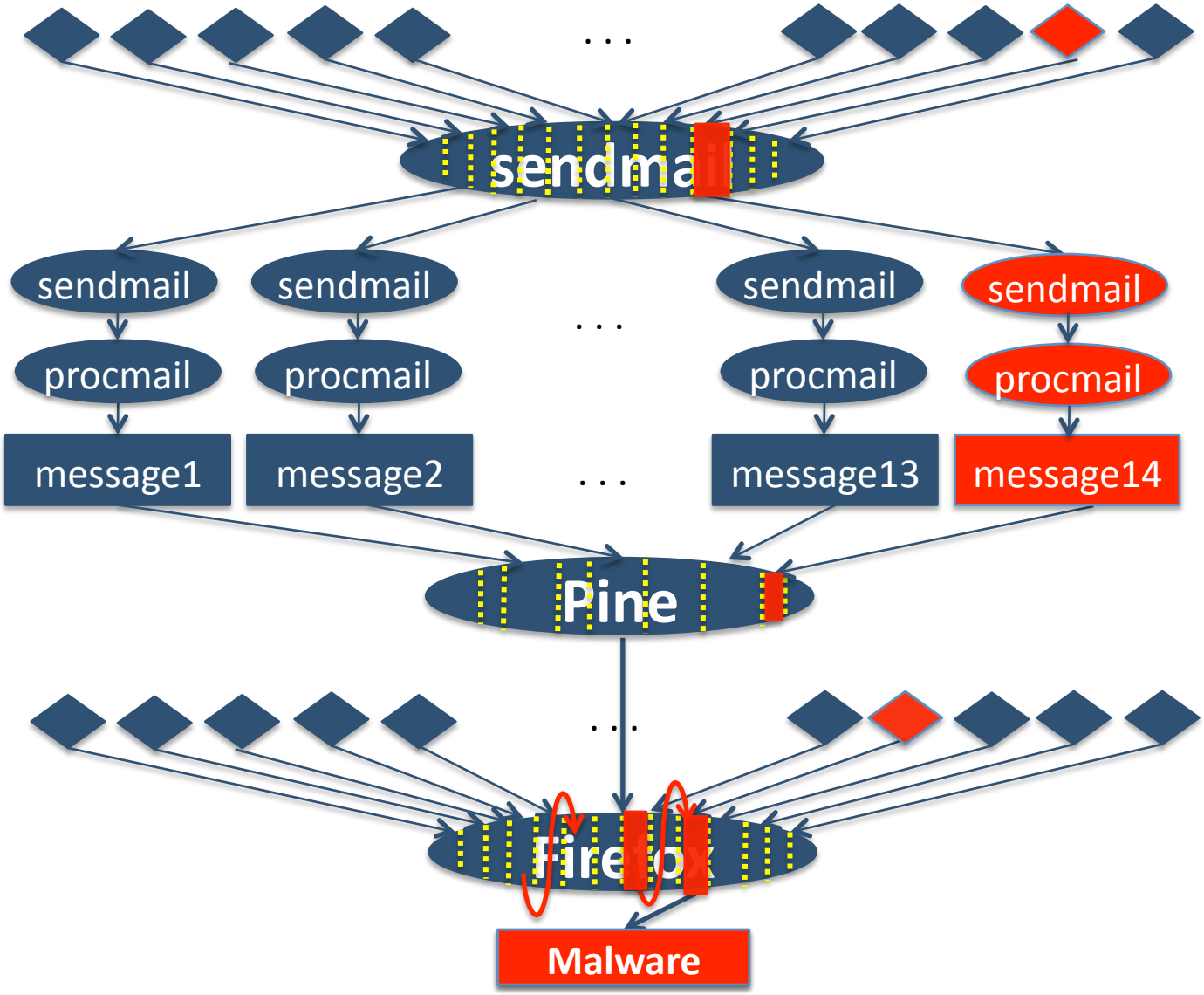
PURDUE UNIVERSITY

CERIAS



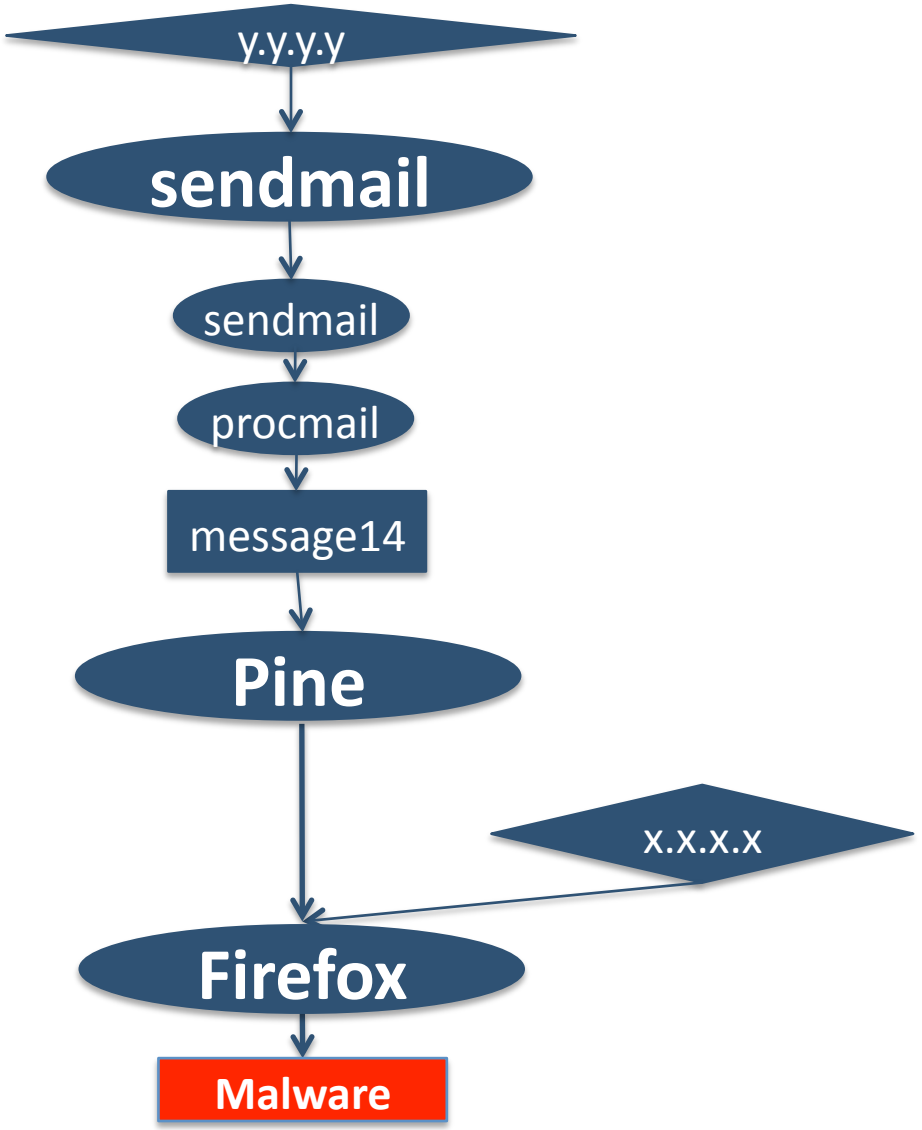
PURDUE UNIVERSITY

CERIAS



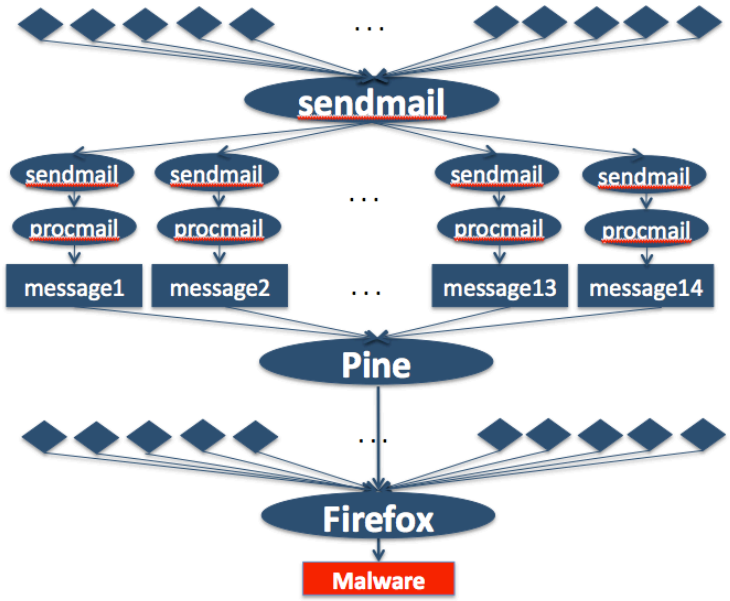
PURDUE UNIVERSITY

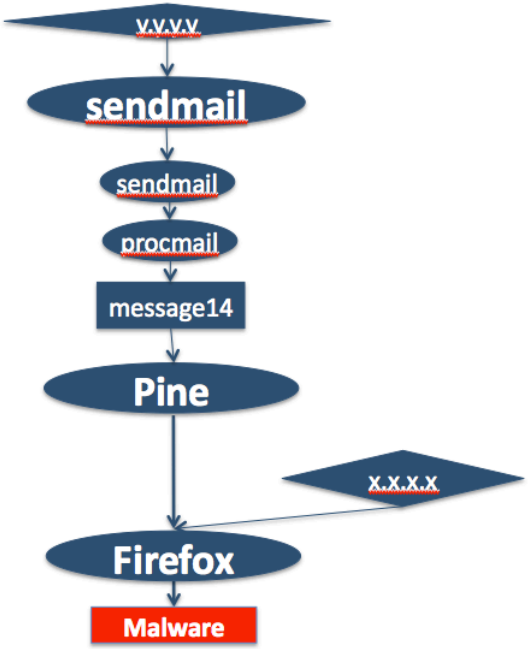
CERIAS



PURDUE UNIVERSITY

CERIAS





Previous Approach :

51 Processes, **15** Files,

251 Network addresses, **351** Edges

16.3 times smaller

BEEP :

10 Processes, **2** Files,

6 Network addresses, **23** Edges

PURDUE UNIVERSITY

CERIAS

BEEP : Our Approach

- **BEEP** : **B**inary-based **E**x**E**cution **P**artition
  - Without source code
  - Reverse engineer units and unit dependences
  - Negligible runtime overhead (1.4% max.)
  - Low space overhead (12.28% avg.)
  - Effective (40.93 times smaller causal graph)

PURDUE UNIVERSITY

CERIAS

Outline

- Identifying UNITs
- Inter-UNIT Dependences
- How BEEP Works
- Experimental Results
- Conclusion

PURDUE UNIVERSITY

CERIAS

Outline

- **Identifying UNITs**
- Inter-UNIT Dependences
- How BEEP Works
- Experimental Results
- Conclusion

PURDUE UNIVERSITY

CERIAS

Identifying Units

	Category	Total Applications	Loop structured Applications
Servers	Web server	13	13
	Mail server	8	8
	FTP server	6	6
	SSHD server	2	2
	DNS server	9	9
	Database server	4	4
	Proxy server	2	2
	Media server	5	5
	Directory server	3	3
	Version control server	2	2
	Remote desktop server	2	2
UI Programs	Web browser	5	5
	E-mail client	5	5
	FTP client	5	5
	Office	2	2
	Text Editor	3	3
	Image tool	4	4
	Audio player	2	2
	Video player	4	4
	P2P program	6	6
	Messanger	2	2
	File manager	2	2
	Shell program	3	3

Identifying Units

	Category	Total Applications	Loop structured Applications
	Web server	13	13
	Mail server	8	8
	FTP server	6	6
	SSHD server	2	2
	DNS server	0	0
	Proxy server	2	2
	Media server	5	5
	Version control server	3	3
	Remote desktop server	2	2
	E-mail client	5	5
	FTP client	5	5
	Office	2	2
UI Programs	Text Editor	3	3
	Image tool	4	4
	Audio player	2	2
	Video player	4	4
	P2P program	6	6
	Messanger	2	2
	File manager	2	2
	Shell program	3	3

Characteristics of long running programs

- ✔ Driven by **external requests**
- ✔ Dominated by **event processing loops**

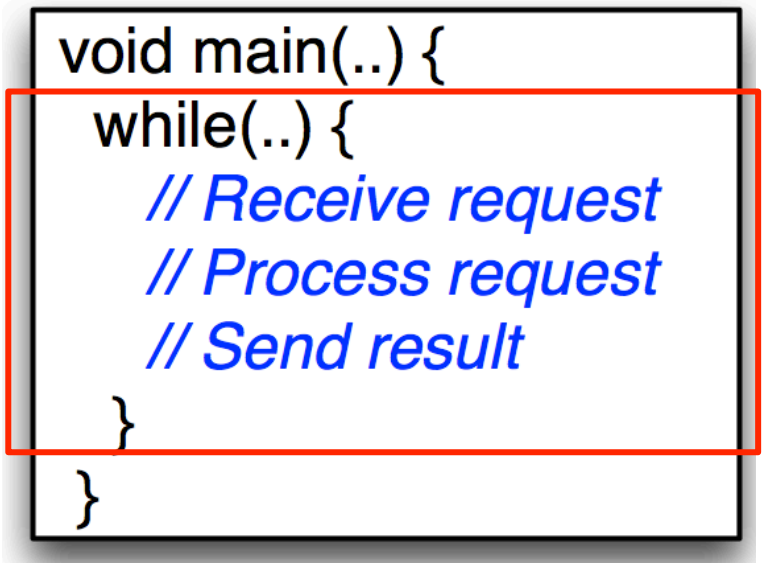
Identifying Units

- Event processing loop

```
void main(..) {
  while(..) {
     // Receive request
     // Process request
     // Send result
   }
}
```

Identifying Units

- Event processing loop



Execution Unit :

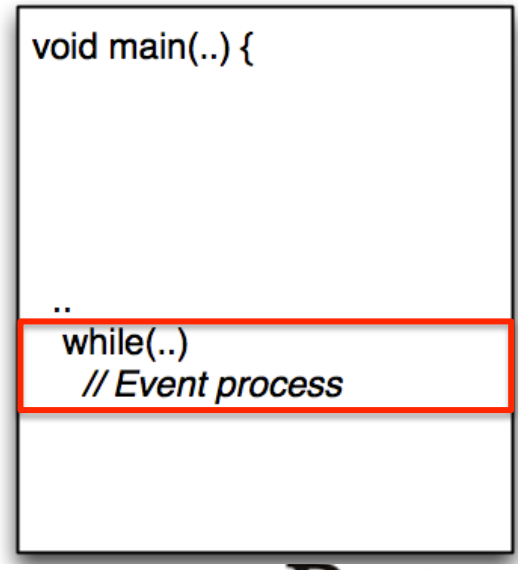
Iteration of event processing loop

PURDUE UNIVERSITY

CERIAS

Identifying Units

- Detecting Unit Loops
  - High level loop



PURDUE UNIVERSITY

CERIAS

Identifying Units

- Detecting Unit Loops
  - High level loop

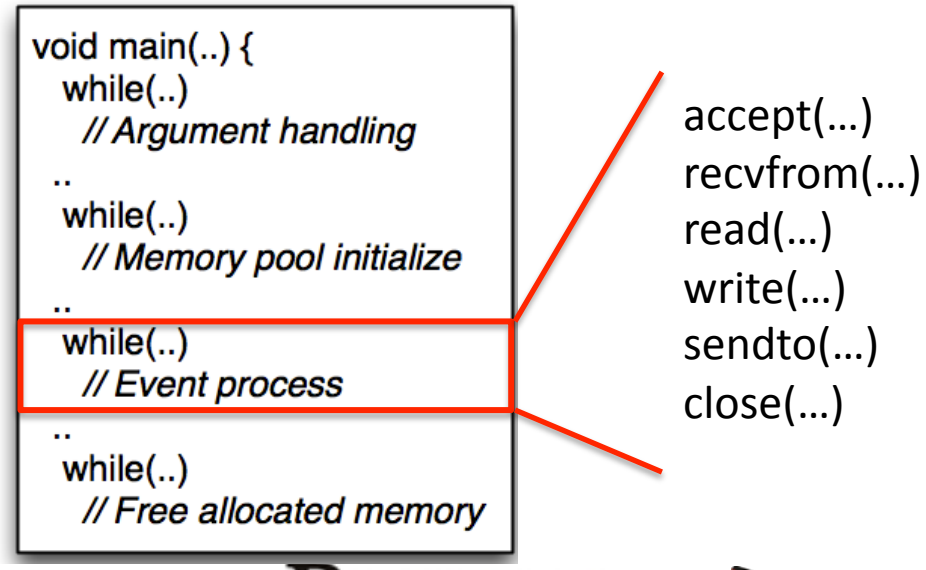
```
void main(..) {
  while(..)
    // Argument handling
  ..
  while(..)
    // Memory pool initialize
  ..
  while(..)
    // Event process

  while(..)
    // Free allocated memory
```

PURDUE UNIVERSITY

CERIAS

Identifying Units

- Detecting Unit Loops
  - High level loop
  - Receive inputs and produce outputs



PURDUE UNIVERSITY

CERIAS

Outline

- Identifying UNITs
- **Inter-UNIT Dependences**
- How BEEP Works
- Experimental Results
- Conclusion

PURDUE UNIVERSITY

CERIAS

Inter-Unit Dependences

- A unit alone may not correspond to a semantically independent sub-execution

Inter-Unit Dependences

```
593   void *listener_thread(..) {
...
631     while(1) {
...
742       req=accept_func(..);
...
768       ap_queue_push(req);
...
798     } // while end
...
810   } // listener_thread end
```

```
820   void *worker_thread(..) {
...
842     while(!worker_may_exit) {
...
862       req=ap_queue_pop();
...
894       process_request(req);
...
899     } // while end
...
906   } // worker_thread end
```

<Apache-2.2.21> - **Multi-thread**

PURDUE UNIVERSITY

CERIAS

Inter-Unit Dependences

```
593   void *listener_thread(..) {
...
631     while(1) {
...
742       req=accept_func(..);
...
768       ap_queue_push(req);
...
798     } // while end
...
810   } // listener_thread end
```

```
820   void *worker_thread(..) {
...
842     while(!worker_may_exit) {
...
862       req=ap_queue_pop();
...
894       process_request(req);
...
899     } // while end
...
906   } // worker_thread end
```

<Apache-2.2.21> - **Multi-thread**

PURDUE UNIVERSITY

CERIAS

Inter-Unit Dependences

```
593   void *listener_thread(..) {
...
631     while(1) {
...
742       req=accept_func(..);
...
768       ap_queue_push(req);
...
798     } // while end
...
810   } // listener_thread end
```

```
820   void *worker_thread(..) {
...
842     while(!worker_may_exit) {
...
862       req=ap_queue_pop();
...
894       process_request(req);
...
899     } // while end
...
906   } // worker_thread end
```

<Apache-2.2.21> - **Multi-thread**

PURDUE UNIVERSITY

CERIAS

Inter-Unit Dependences

```
593   void *listener_thread(..) {
...
631     while(1) {
...
742       req=accept_func(..);
...
768       ap_queue_push(req);
...
798     } // while end
...
810   } // listener_thread end
```

```
820   void *worker_thread(..) {
...
842     while(!worker_may_exit) {
...
862       req=ap_queue_pop();
...
894       process_request(req);
...
899     } // while end
...
906   } // worker_thread end
```

<Apache-2.2.21> - **Multi-thread**

PURDUE UNIVERSITY

CERIAS

Inter-Unit Dependences

```
593   void *listener_thread(..) {
...
631     while(1) {
...
742       req=accept_func(..);
...
768       ap_queue_push(req);
...
798     } // while end
...
810   } // listener_thread end
```

```
820   void *worker_thread(..) {
...
842     while(!worker_may_exit) {
...
862       req=ap_queue_pop();
...
894       process_request(req);
...
899     } // while end
...
906   } // worker_thread end
```

<Apache-2.2.21> - **Multi-thread**

PURDUE UNIVERSITY

CERIAS

Inter-Unit Dependences

```
593   void *listener_thread(..) {
...
631     while(1) {
...
742       req=accept_func(..);
...
768       ap_queue_push(req);
...
798     } // while end
...
810   } // listener_thread end
```

```
820   void *worker_thread(..) {
...
842     while(!worker_may_exit) {
...
862       req=ap_queue_pop();
...
894       process_request(req);
...
899     } // while end
...
906   } // worker_thread end
```

<Apache-2.2.21> - **Multi-thread**

PURDUE UNIVERSITY

CERIAS

Inter-Unit Dependences

```
593   void *listener_thread(..) {
...
631     while(1) {
...
742       req=accept_func(..);
...
768       ap_queue_push(req);
...
798     } // while end
...
810   } // listener_thread end
```

```
820   void *worker_thread(..) {
...
842     while(!worker_may_exit) {
...
862       req=ap_queue_pop();
...
894       process_request(req);
...
899     } // while end
...
906   } // worker_thread end
```

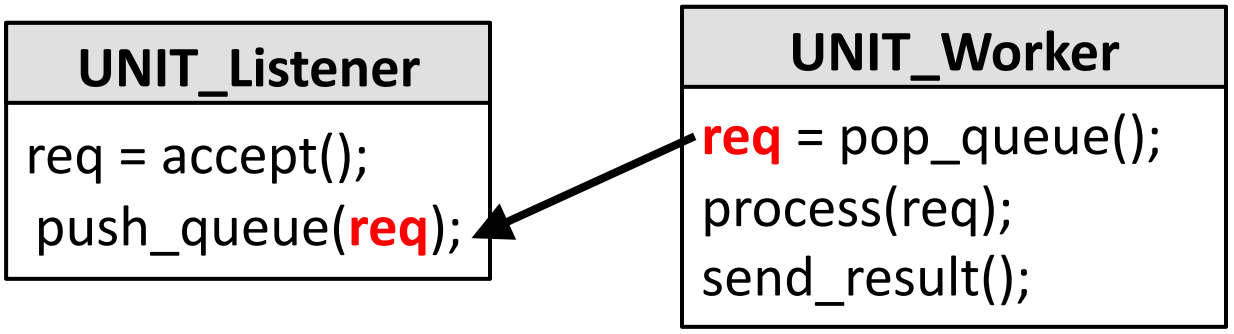
<Apache-2.2.21> - **Multi-thread**

PURDUE UNIVERSITY

CERIAS

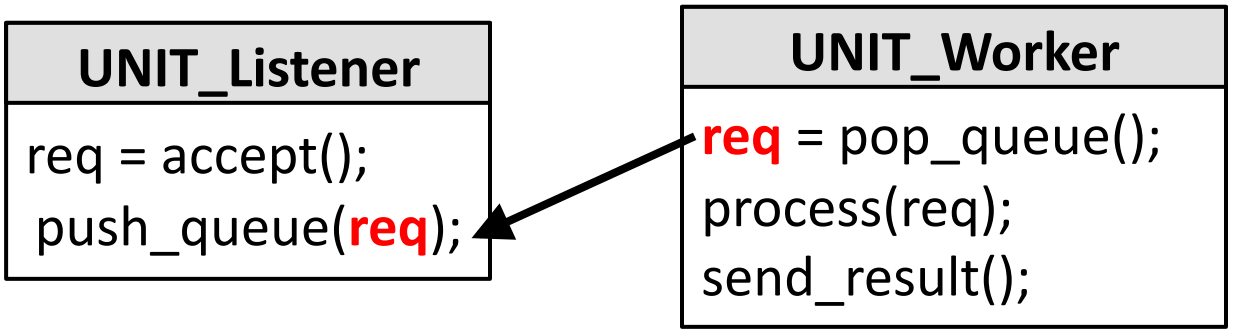
Inter-Unit Dependences

- A unit alone may not correspond to a semantically independent sub-execution
  - A few units together compose an autonomous sub-execution



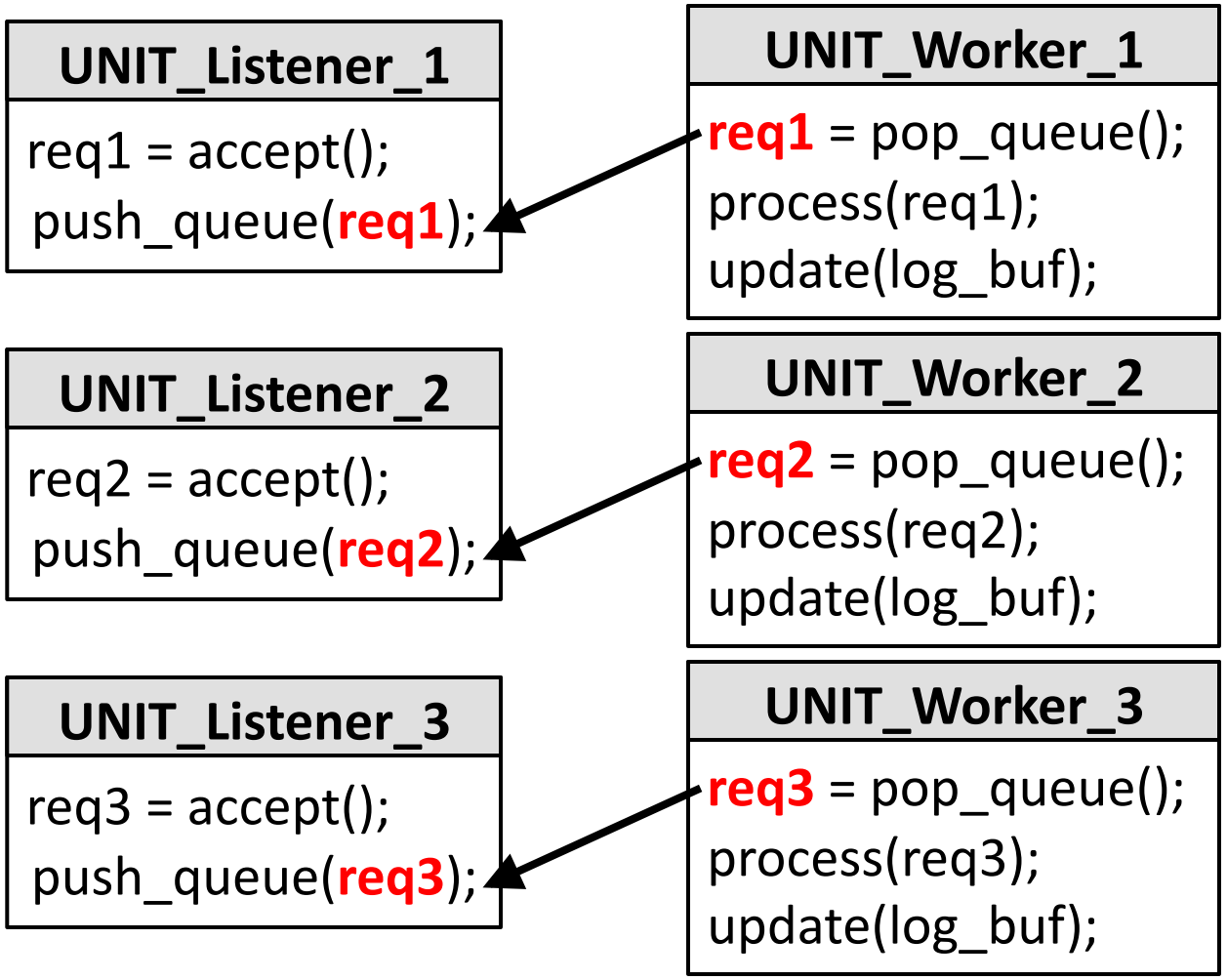
Inter-Unit Dependences

- A unit alone may not correspond to a semantically independent sub-execution
  - A few units together compose an autonomous sub-execution



  - Dependences through workflow object
    - Ex) queue - enqueue, dequeue

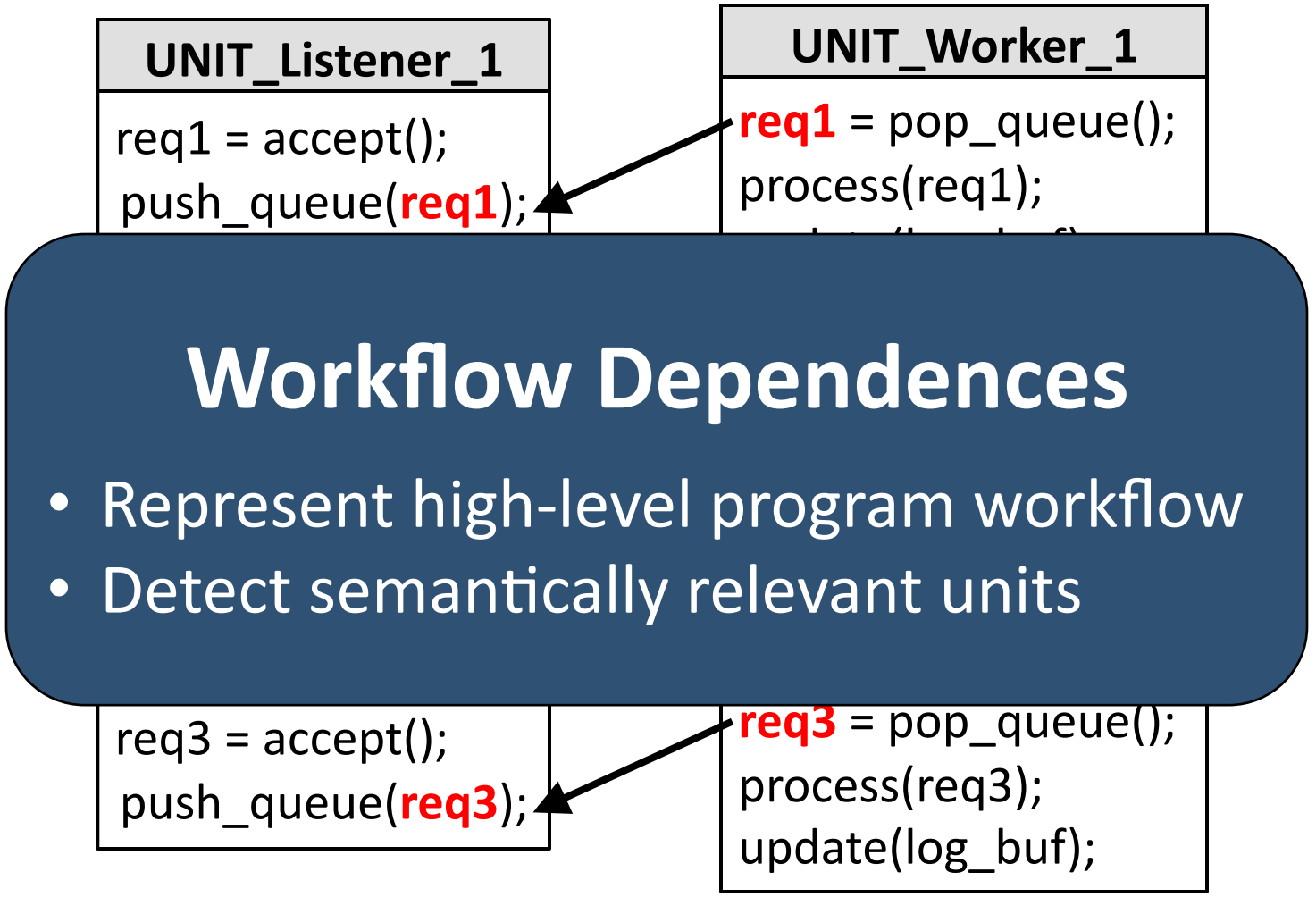
Inter-Unit Dependences



PURDUE UNIVERSITY

CERIAS

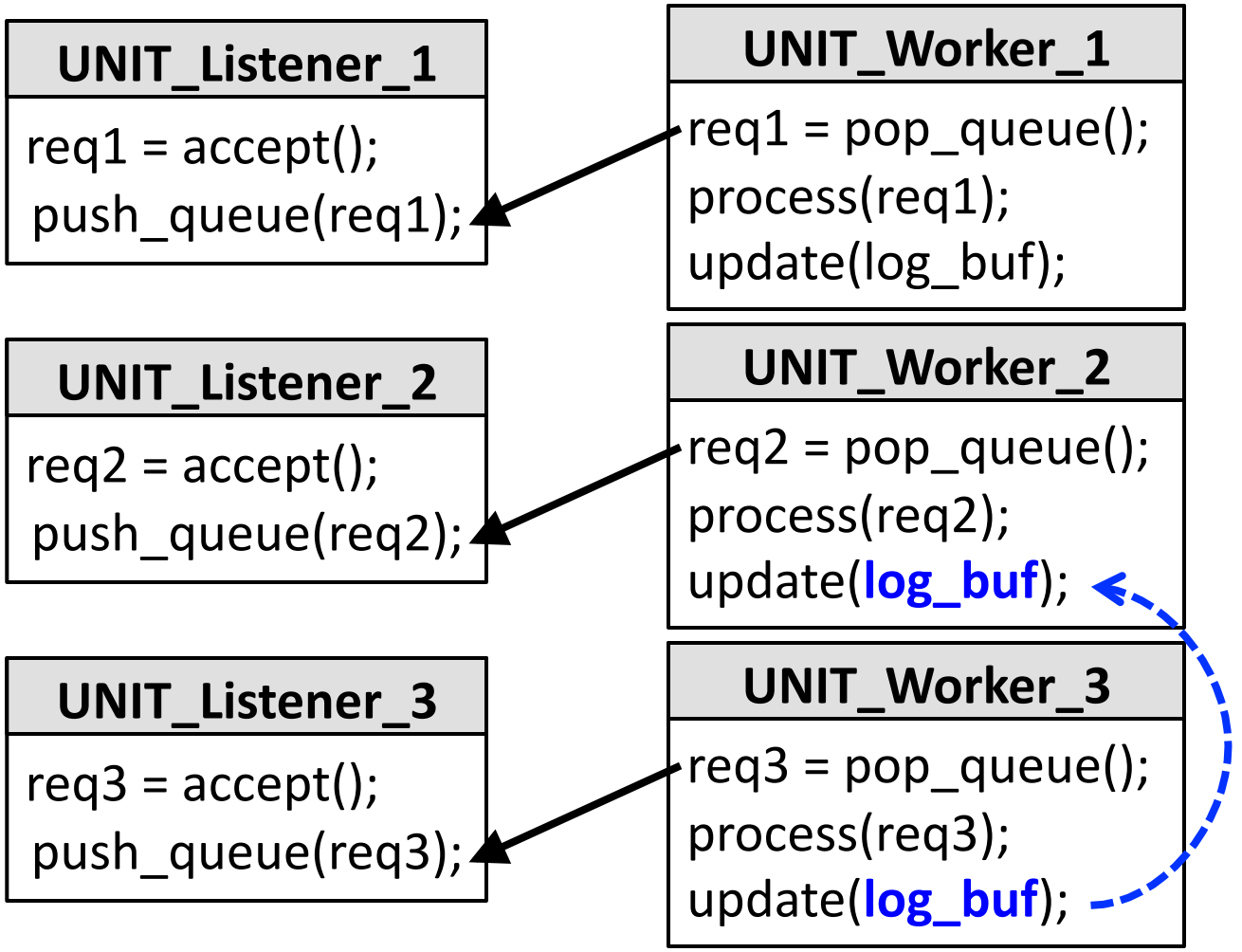
Inter-Unit Dependences



PURDUE UNIVERSITY

CERIAS

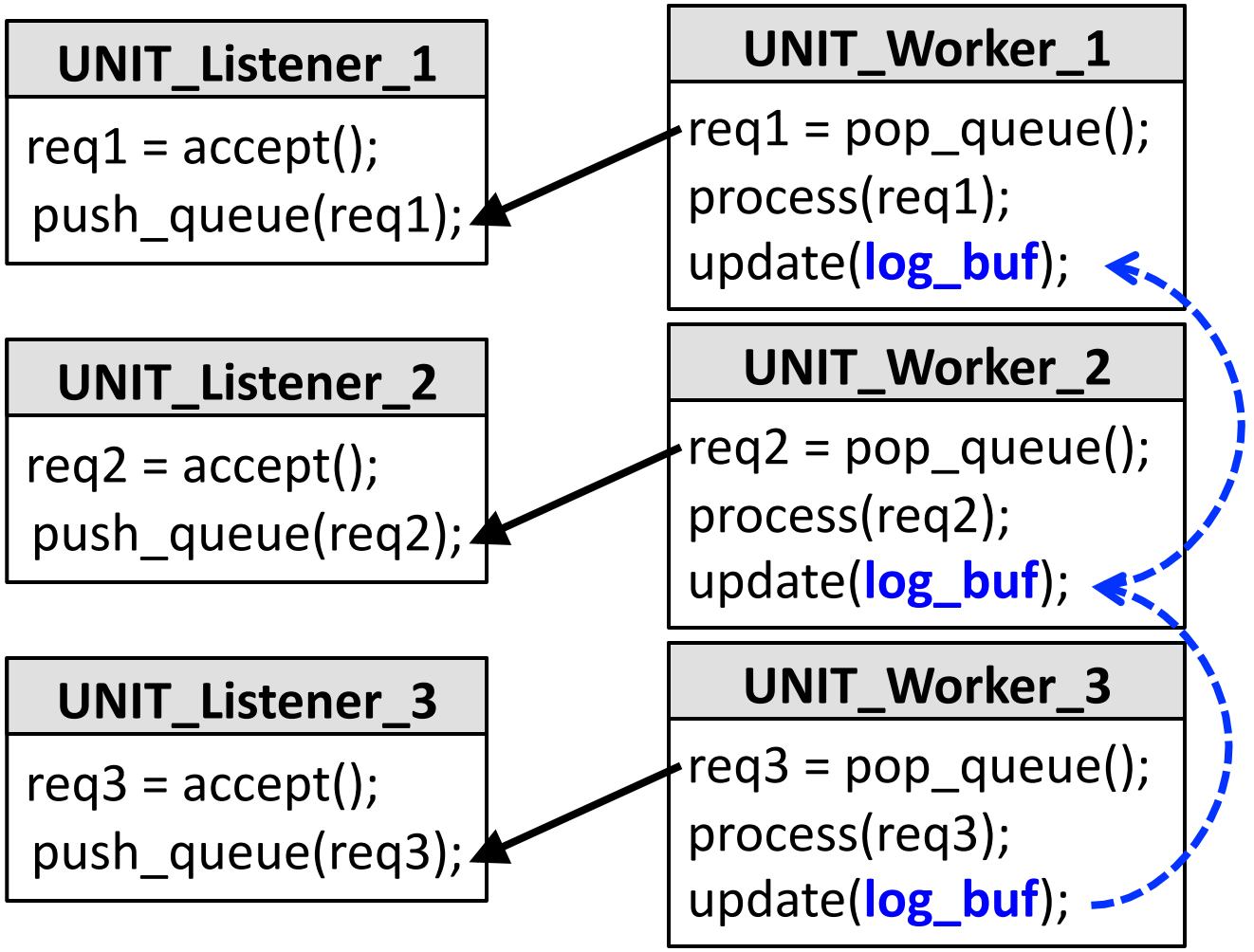
Inter-Unit Dependences



PURDUE UNIVERSITY

CERIAS

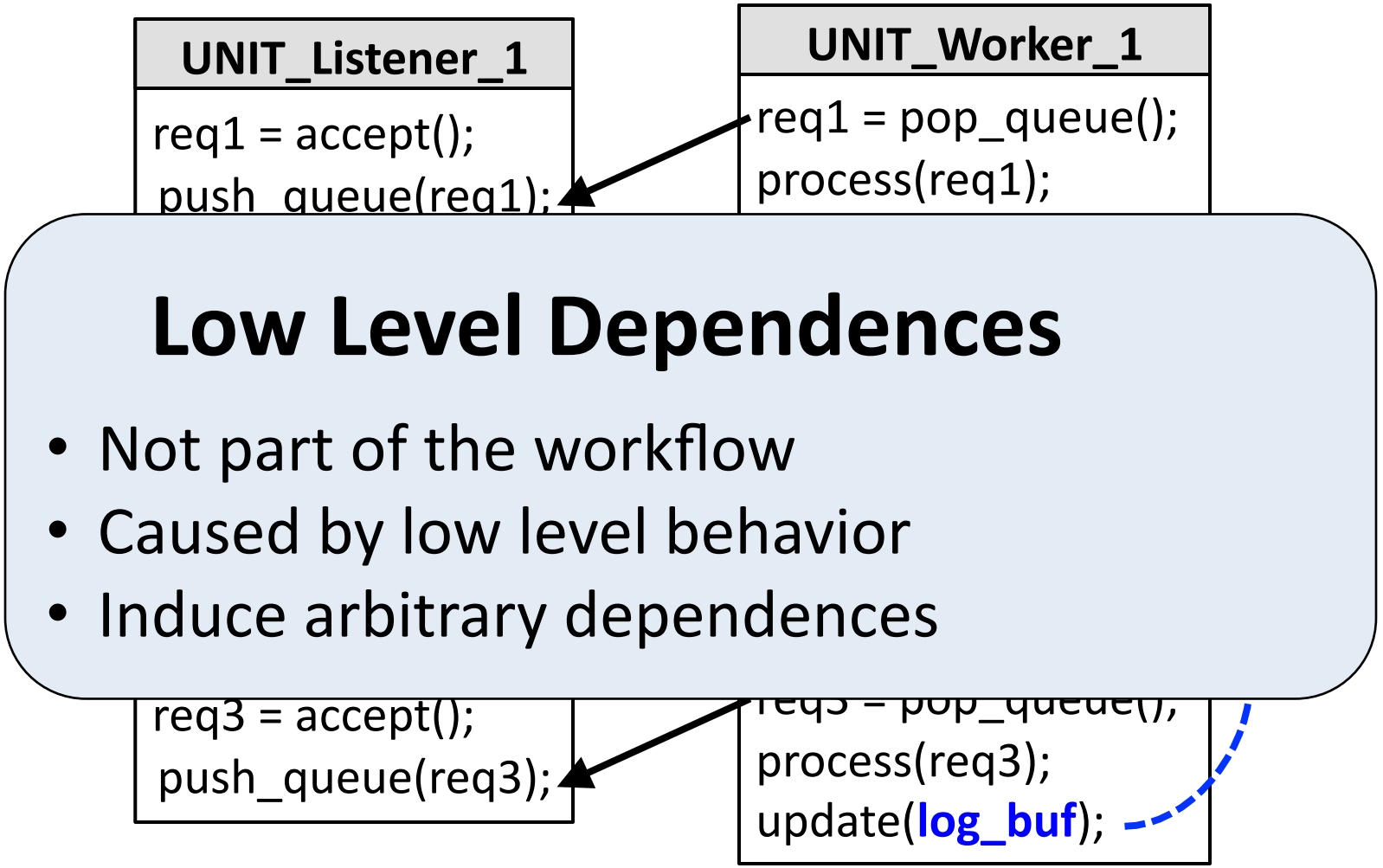
Inter-Unit Dependences



PURDUE UNIVERSITY

CERIAS

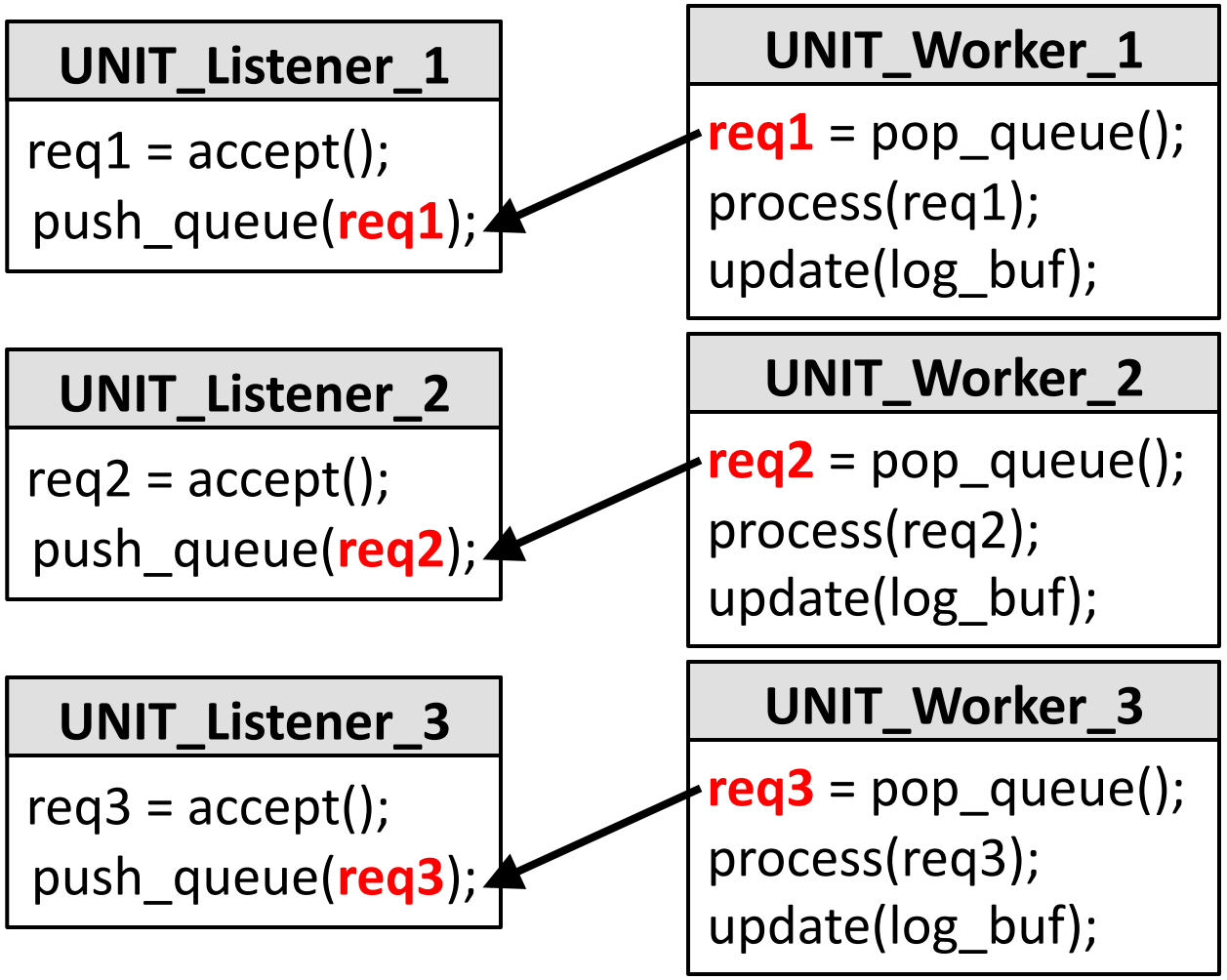
Inter-Unit Dependences



PURDUE UNIVERSITY

CERIAS

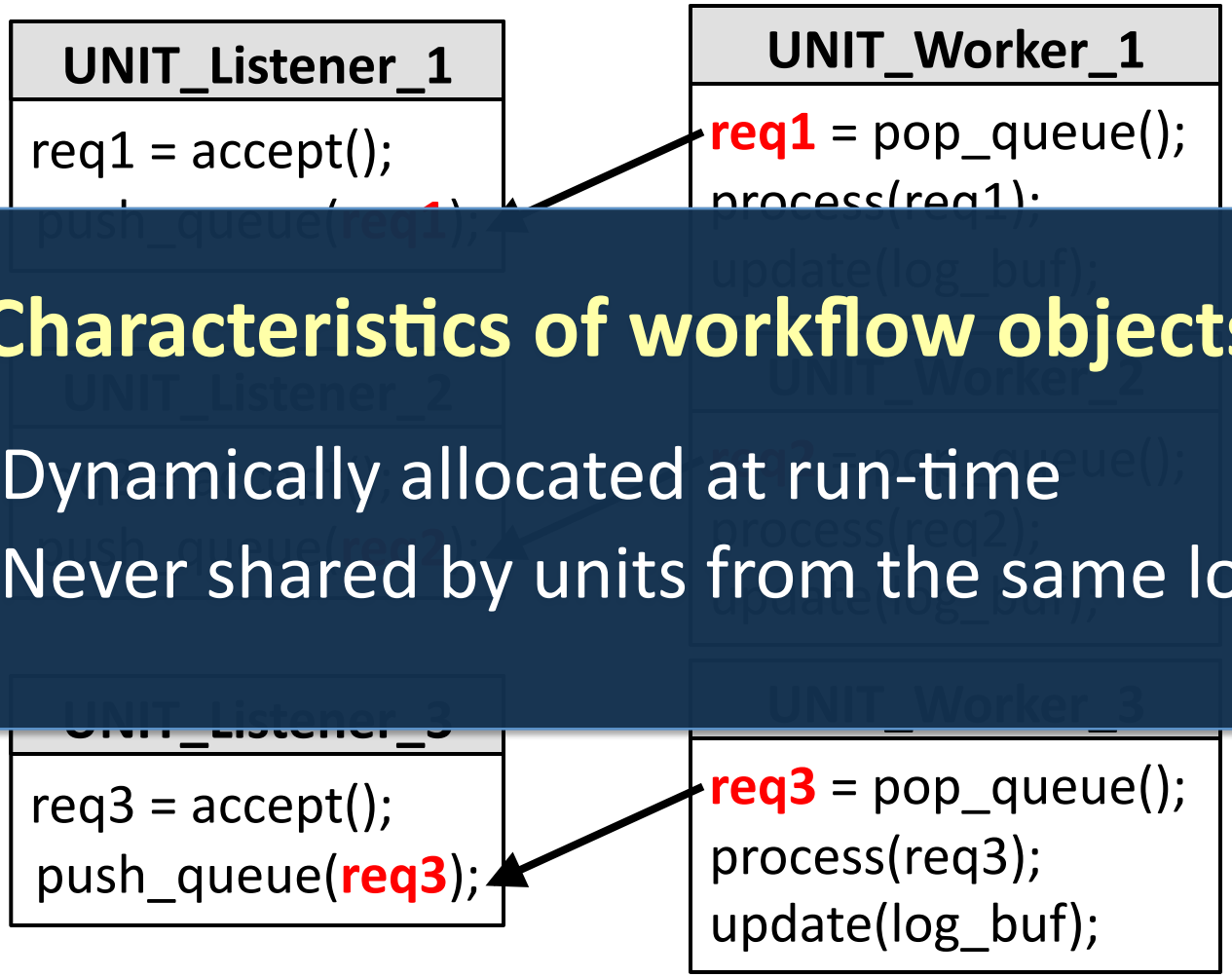
Inter-Unit Dependences



PURDUE UNIVERSITY

CERIAS

Inter-Unit Dependences



Characteristics of workflow objects

- ✔ Dynamically allocated at run-time
- ✔ Never shared by units from the same loop

PURDUE UNIVERSITY

CERIAS

Outline

- Identifying UNITs
- Inter-UNIT Dependences
- **How BEEP Works**
- Experimental Results
- Conclusion

PURDUE UNIVERSITY

CERIAS

How BEEP Works

Static Analysis

PURDUE UNIVERSITY

CERIAS

How BEEP Works

Static Analysis

Training Runs

PURDUE UNIVERSITY

CERIAS

How BEEP Works

Static Analysis

Training Runs

Instrumentation

PURDUE UNIVERSITY

CERIAS

How BEEP Works

PURDUE UNIVERSITY

CERIAS

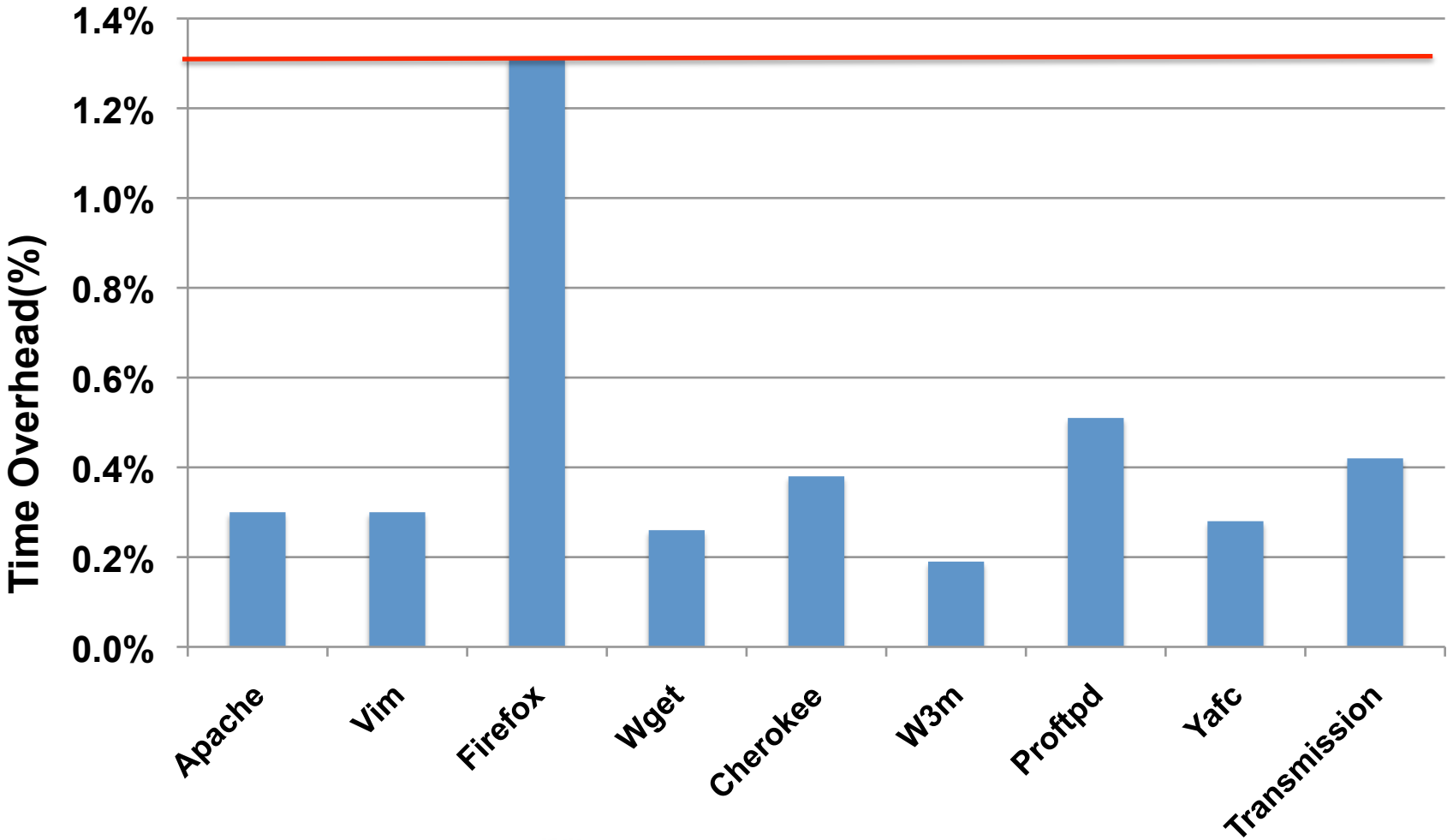
Outline

- Identifying UNITs
- Inter-UNIT Dependences
- How BEEP Works
- **Experimental Results**
- Conclusion

PURDUE UNIVERSITY

CERIAS

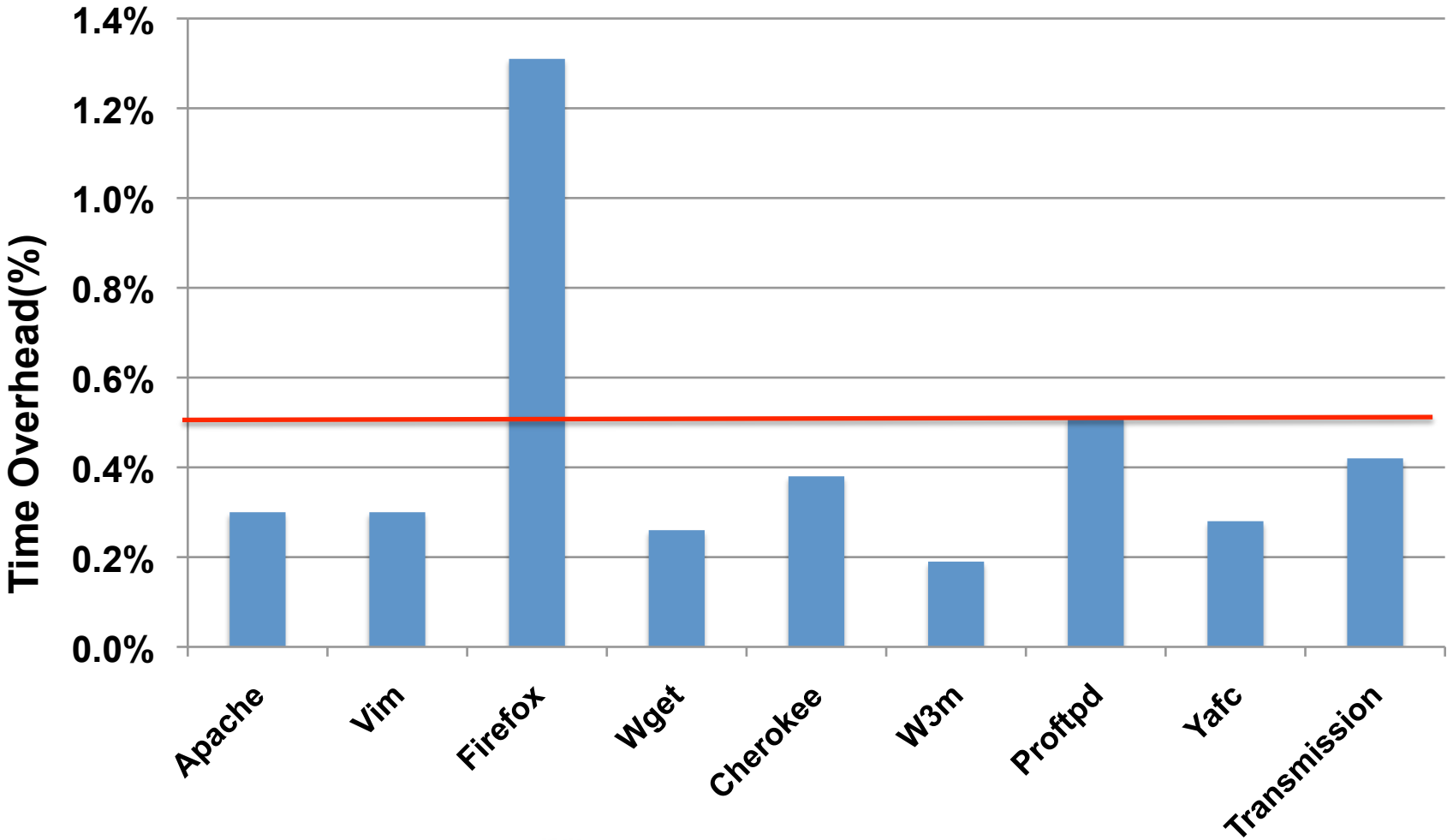
Logging Overhead (Time)



PURDUE UNIVERSITY

CERIAS

Logging Overhead (Time)



PURDUE UNIVERSITY

CERIAS

Logging Overhead (Space)

App	Log Size (MB)		Space Overhead (%)
	Original	Instrumented	
Apache	0.227	0.278	22.4%
Vim	85.75	89.4	4.2%
Firefox	46.12	63.25	37.1%
Cherokee	1.1	1.126	2.2%
W3M	26.88	28.39	5.6%
Proftpd	0.54	0.56	2.3%
Wget	17.47	17.48	0.6%
Yafc	7.29	7.41	1.6%
Transmission	4.92	5.37	9.1%
Pine	5.62	6.21	10.5%
Bash	0.63	0.64	0.6%
MC	1.56	1.67	6.9%
Sshd	2.2	2.22	0.8%
Sendmail	0.214	0.22	2.8%

Logging Overhead (Space)

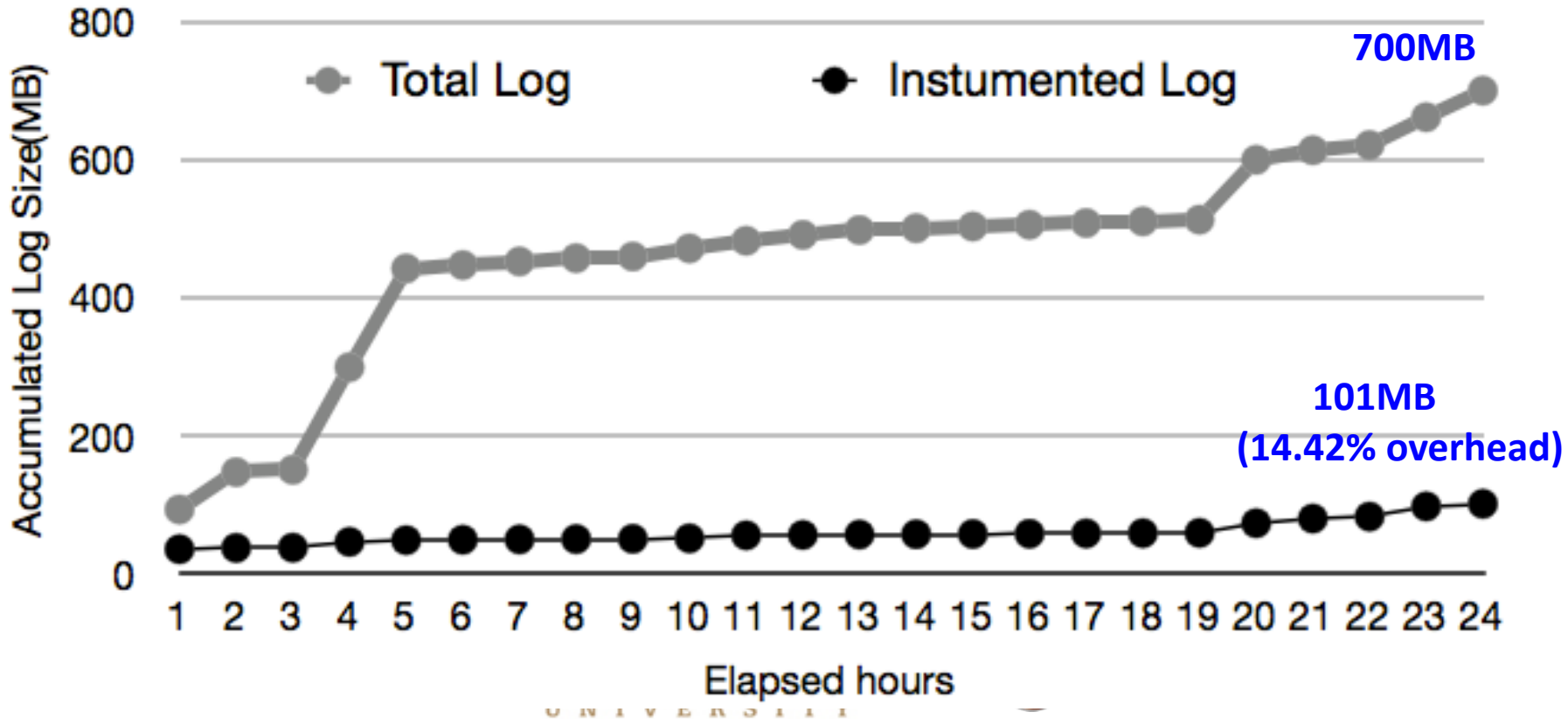
App	Log Size (MB)		Space Overhead (%)
	Original	Instrumented	
Apache	0.227	0.278	22.4%
Vim	85.75	89.4	4.2%
Firefox	46.12	63.25	37.1%
Cherokee	1.1	1.126	2.2%
W3M	26.88	28.39	5.6%
Proftpd	0.54	0.56	2.3%
Wget	17.47	17.48	0.6%
Yafc	7.29	7.41	1.6%
Transmission	4.92	5.37	9.1%
Pine	5.62	6.21	10.5%
Bash	0.63	0.64	0.6%
MC	1.56	1.67	6.9%
Sshd	2.2	2.22	0.8%
Sendmail	0.214	0.22	2.8%

Logging Overhead (Space)

App	Log Size (MB)		Space Overhead (%)
	Original	Instrumented	
Apache	0.227	0.278	22.4%
Vim	85.75	89.4	4.2%
Firefox	46.12	63.25	37.1%
Cherokee	1.1	1.126	2.2%
W3M	26.88	28.39	5.6%
Proftpd	0.54	0.56	2.3%
Wget	17.47	17.48	0.6%
Yafc	7.29	7.41	1.6%
Transmission	4.92	5.37	9.1%
Pine	5.62	6.21	10.5%
Bash	0.63	0.64	0.6%
MC	1.56	1.67	6.9%
Sshd	2.2	2.22	0.8%
Sendmail	0.214	0.22	2.8%

Logging Overhead (One day)

Server programs	UI programs
Sshd-5.9, Sendmail-8.12.11 Proftpd-1.3.4, Apache-2.2.21 Cherokee-1.2.1	Wget-1.13, W3m-0.5.2, Pine-4.64, MC-4.6.1, Vim-7.3, Bash-4.2, Firefox-11, Yafc-1.1.1, Transmission-2.6



Effectiveness

- Attack Ramification
  - 332 times smaller

	# of Processes	# of Files	# of Sockets	# of Edges
Previous approach	348	512	0	2,135
BEEP	4	1	0	4

- Information Stealing
  - 7.8 times smaller

	# of Processes	# of Files	# of Sockets	# of Edges
Previous approach	2	11	38	51
BEEP	2	4	1	6

PURDUE UNIVERSITY

CERIAS

Related Works

- Coarse-grained analysis

  [S. T. King et al. '03], [A. Goel et al. '05]

- Whole system replay

  [T. Kim et al. '10], [R. Chandra et al. '11]

- Coarse-grained + instruction logging

  [X. Wang et al. '09], [P. Barham et al. '04]

- Instruction level information flow

  [H. Yin et al. '07], [B. C. Tak et al. '09]

PURDUE UNIVERSITY

CERIAS

Conclusion

- **BEEP** : **B**inary-based **E**x**E**cution **P**artition
  - Highly accurate provenance tracing technique
    - Fine-grained autonomous units
    - Reverse engineer units and unit dependences
    - Without source code
  - Negligible runtime overhead (1.4% max.)
  - Low space overhead (12.28% avg.)
  - Effective (40.93 times smaller causal graph)

PURDUE UNIVERSITY

CERIAS

Q & A

Unit Loop Detection

- Missing unit loops
  - Undesirable unification of dependences
    - –May cause false dependences
    - –Does not affect correctness
- Including bogus loops
  - Unit dependence analysis will detect the causal dependences between two loops
    - –Cause additional overhead
    - –Does not affect correctness

PURDUE UNIVERSITY CERIAS

Unit Dependence Detection

- Missing unit dependences
  - BEEP detects the workflow dependences
    - As long as trainings cover "some" of instructions
- Including low level dependences
  - May lead to bogus inter-unit dependences
    - Does not affect correctness

PURDUE UNIVERSITY

CERIAS