

HØGSKOLEN I GJØVIK



Process Tracking for Forensic Readiness

Yi-Ching Liao

Norwegian Information Security Laboratory



yi-ching.liao@hig.no

COINS Ph.D. student seminar 2014

ABSTRACT

- **Summarize the research on process tracking for forensic readiness**
 - the state-changing activities of processes
 - cost-benefit analysis of process tracking
 - the architecture for process tracking

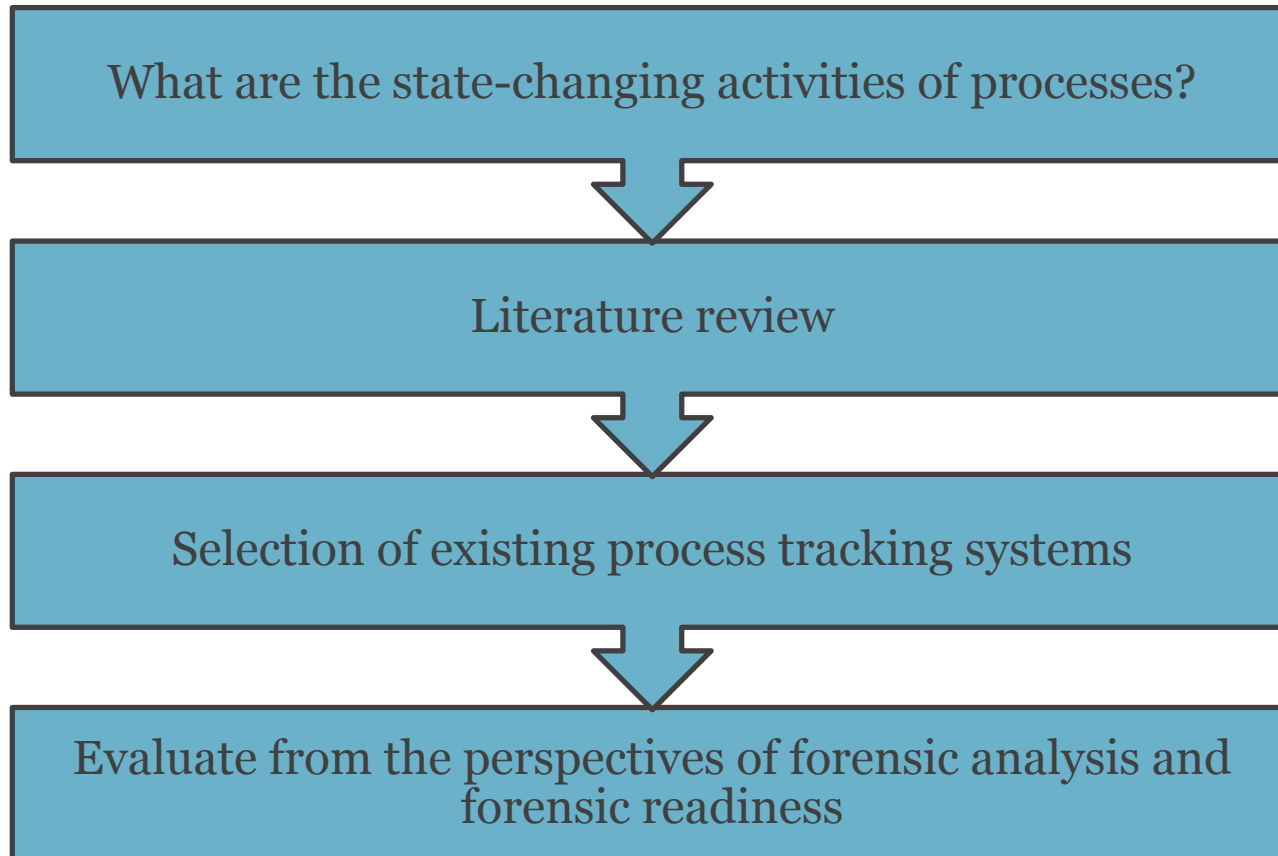
PROBLEM STATEMENT

- **Forensic analysis**
 - suffers from insufficient logging of events
- **Current system loggers**
 - do not record enough information for incident analysis and replay
- **Comprehensive process tracking**
 - provides precise, timely, complete, and dependable information for incident investigation and replay
 - recovers the traceability links between the incident and the person or action accountable for the incident

RESEARCH QUESTIONS

1. What are the state-changing activities of processes?
2. How effective, efficient, and expensive is comprehensive process activity tracking?
3. Which hardware/software architecture facilitates process activity tracking?
4. What are privacy implications for users of systems that support comprehensive traceability?
5. How does comprehensive traceability affect evidence gathering and the legal process?

STATE-CHANGING ACTIVITIES OF PROCESSES OVERVIEW



STATE-CHANGING ACTIVITIES OF PROCESSES

LITERATURE REVIEW

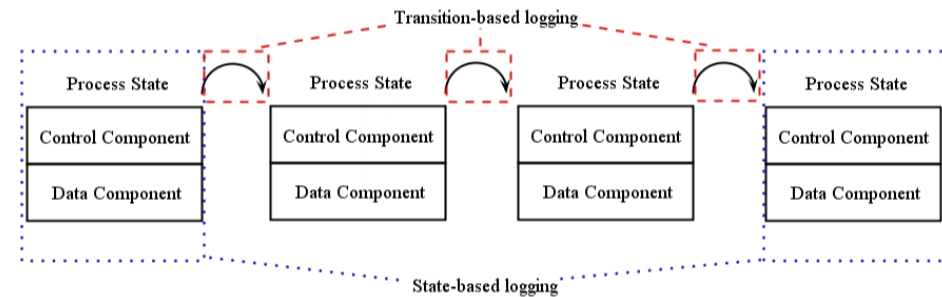
- **Program execution monitors**
 - completeness
 - record all possible process states of a program
 - pertinence
 - log no data outside the state space of a program
- **Program comprehension through dynamic analysis**
- **Execution replay systems**

STATE-CHANGING ACTIVITIES OF PROCESSES CANDIDATES FOR EVALUATION

- **Selection of existing process tracking systems**
 - for security
 - 9 process activity tracking systems
 - for debugging
 - 11 process activity tracking systems

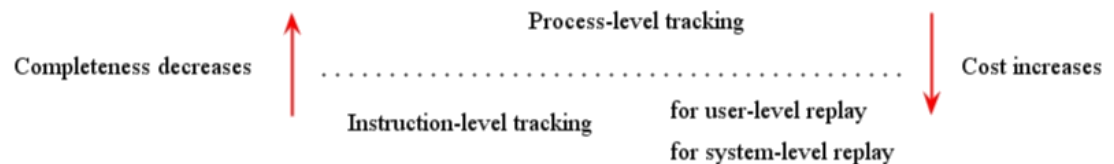
STATE-CHANGING ACTIVITIES OF PROCESSES EVALUATION

- Evaluate from the perspectives of forensic analysis and forensic readiness
 - the logging method
 - state-based
 - transition-based
 - the implementation method
 - user-space
 - kernel-space



STATE-CHANGING ACTIVITIES OF PROCESSES EVALUATION (CONTINUED)

- Evaluate from the perspectives of forensic analysis and forensic readiness
 - the tracing granularity
 - instruction-level
 - process-level
 - the replay boundary
 - user-level
 - system-level



STATE-CHANGING ACTIVITIES OF PROCESSES

SUMMARY

System name	Logging method ¹	Tracing granularity ²	Replay boundary ³	Design purpose ⁴	Implementation method ⁵	OS ⁶
ReVirt [17]	SB	IL	SL	S	VM	L
ExecRecorder [15]	SB & TB	IL	SL	S	Emulator	Any
AskStrider [50]	TB	PL	N/A	S	U	W
Capture [45]	TB	PL	N/A	S	K	W
XenLR [28]	TB	IL	SL	S	Hypervisor	L
Process Hacker [42]	TB	PL	N/A	S	K	W
Process Monitor [31]	TB	PL	N/A	S	K	W
Carbon Black [10]	TB	PL	N/A	S	N/A	W
FileSure [9]	TB	PL	N/A	S	N/A	W
Tornado [13]	TB	IL	UL	D	K & U	L
Jockey [43]	TB	IL	UL	D	B & U	L
liblog [20]	TB	IL	UL	D	B & U	L
Flashback [46]	SB	IL	UL	D	K	L
iDNA [5]	TB	IL	UL	D	B	W
ODR [1]	TB	IL	UL	D	B & K	L
Respec [26]	SB & TB	IL	UL	D	K	L
DoublePlay [48]	SB & TB	IL	UL	D	K	L
FDR [51]	SB & TB	IL	SL	D	H	Any
BugNet [35]	SB & TB	IL	UL	D	B & H	L
QuickRec [40]	TB	IL	SL	D	H & K	L

¹ SB=State-based; TB=Transition-based

² IL=Instruction-level; PL=Process-level

³ SL=System-level; UL=User-level

⁴ D=Debugging; S=Security

⁵ B=Binary patching; H=Hardware; K=Kernel-space; U=User-space

⁶ L=Linux; W=Windows

STATE-CHANGING ACTIVITIES OF PROCESSES

CONCLUSION

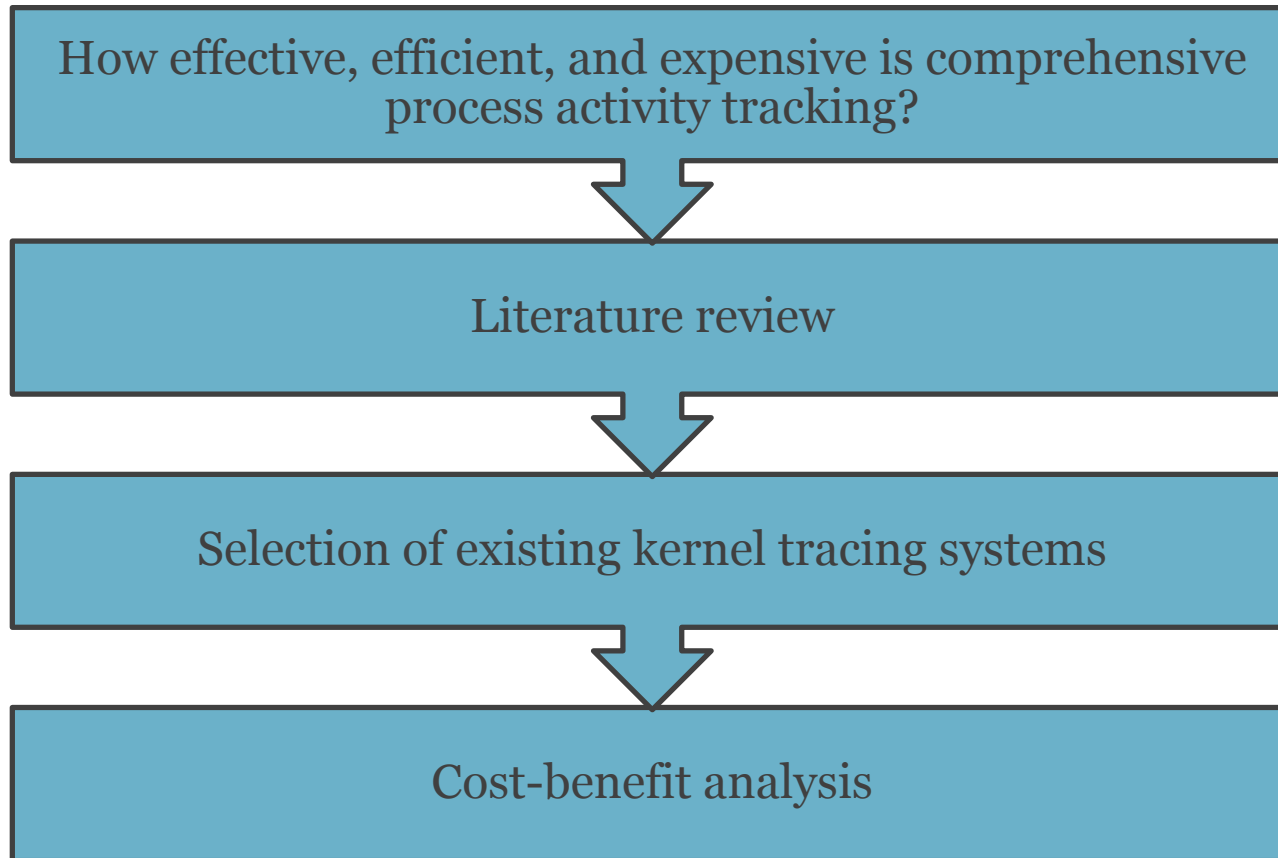
- **Process activity tracking can provide sufficient evidence for investigation if the tracking is**
 - transition-based
 - system-level
 - kernel-space implementation
- **Forensic analysis requires**
 - better data
 - not overwhelming data

STATE-CHANGING ACTIVITIES OF PROCESSES

QUESTION RAISED

- **To strike a balance between the forensic effectiveness and efficiency, we need to**
 - evaluate the soundness, completeness, and cost of process activity tracking

COST-BENEFIT ANALYSIS OF PROCESS TRACKING OVERVIEW



COST-BENEFIT ANALYSIS OF PROCESS TRACKING LITERATURE REVIEW

- **Kernel tracing for**
 - forensics
 - reverse engineering
 - system behavior analysis
- **Kernel traces**
 - low-level execution logs of operating systems
 - should be considered as one of the potential evidence sources used in forensic readiness

COST-BENEFIT ANALYSIS OF PROCESS TRACKING CANDIDATES FOR EVALUATION

- **strace**
 - monitors and intercepts the system calls that a program utilizes and all the signals a program receives
 - only for program-specific tracing
- **SystemTap**
 - utilizes dynamic instrumentation through a simple scripting language
 - for program-specific tracing and system-wide tracing
- **LTTng**
 - utilizes static instrumentation
 - only for system-wide tracing

COST-BENEFIT ANALYSIS OF PROCESS TRACKING EVALUATION STRATEGY

- **System call fuzzer, Trinity, to exercise system calls**
 - 292 32-bit and 277 64-bit system calls
 - 100 times for each system call
- **Kernel tracing systems to trace the exercised system calls**
 - program-specific tracing
 - trace the fuzzer only
 - system-wide tracing
 - stop tracing after the fuzzer exits
 - record operating system noise

COST-BENEFIT ANALYSIS OF PROCESS TRACKING SYSTEM CALL CATEGORIZATION

- According to the accessed system resource

Architecture	Resource	Number of system calls	
32-bit	File system	134	} 292
	Inter-process communication	7	
	Kernel	122	
	Key control	3	
	Memory management	23	
	Network	3	
64-bit	File system	122	} 277
	Inter-process communication	18	
	Kernel	94	
	Key control	3	
	Memory management	22	
	Network	18	

COST-BENEFIT ANALYSIS OF PROCESS TRACKING

BENEFIT ANALYSIS

- **Comprehensive coverage**

- percentage of successfully traced system calls

- $$\frac{\textit{count of successfully traced system calls}}{\textit{actual count of system calls exercised by fuzzer}}$$

- system-wide tracing

- exclude other system calls except the system call we intend to trace

COST-BENEFIT ANALYSIS OF PROCESS TRACKING

COST ANALYSIS

- **Performance overhead**

- $$\frac{(u+s)_{tracing} - (u+s)_{fuzzer}}{c}$$

- u: user process time

- s: sys process time

- c: count of successfully traced system calls

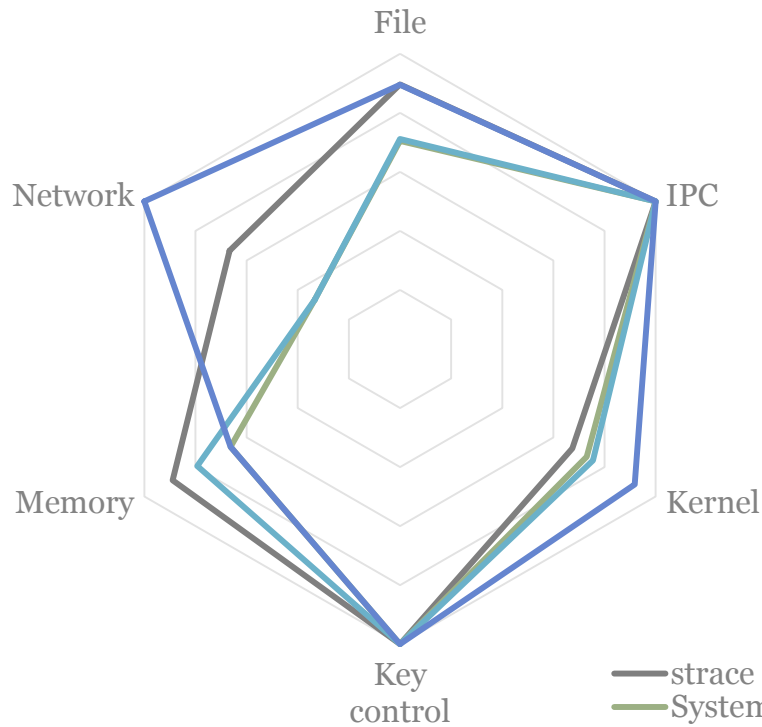
- **Storage overhead**

- $$\frac{\text{trace output size}}{c}$$

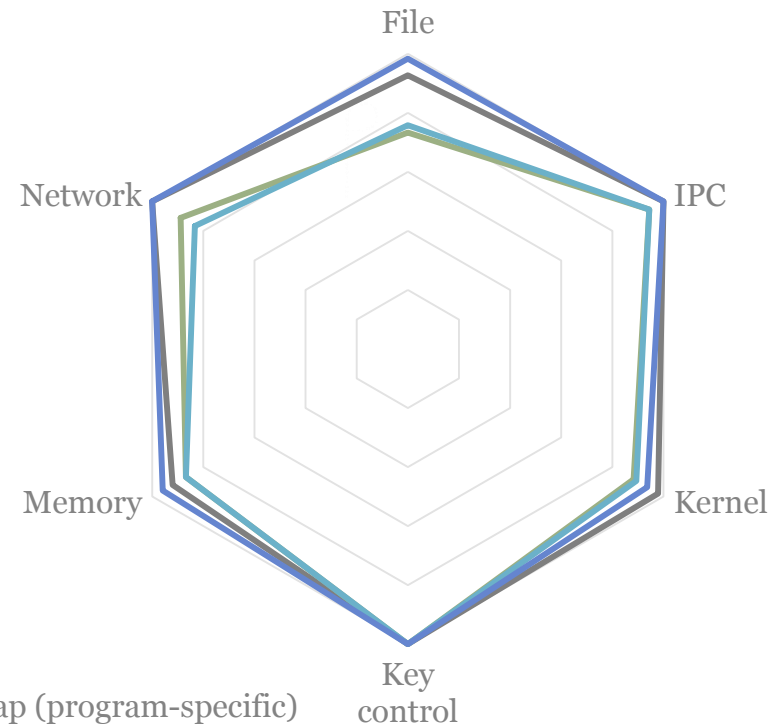
COST-BENEFIT ANALYSIS OF PROCESS TRACKING

BENEFIT ANALYSIS RESULTS

Comprehensive coverage in
32-bit architecture



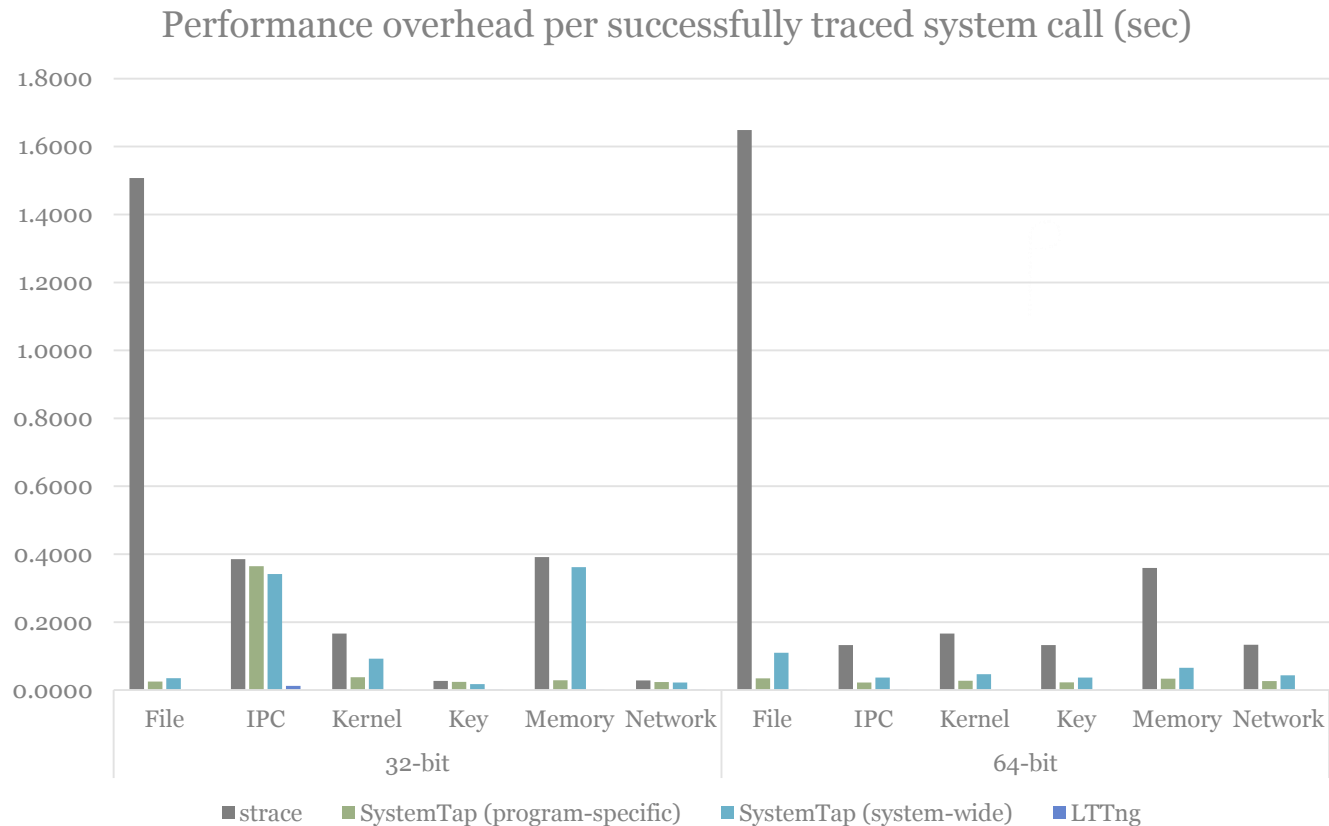
Comprehensive coverage in
64-bit architecture



— strace
 — SystemTap (program-specific)
 — SystemTap (system-wide)
 — LTTng

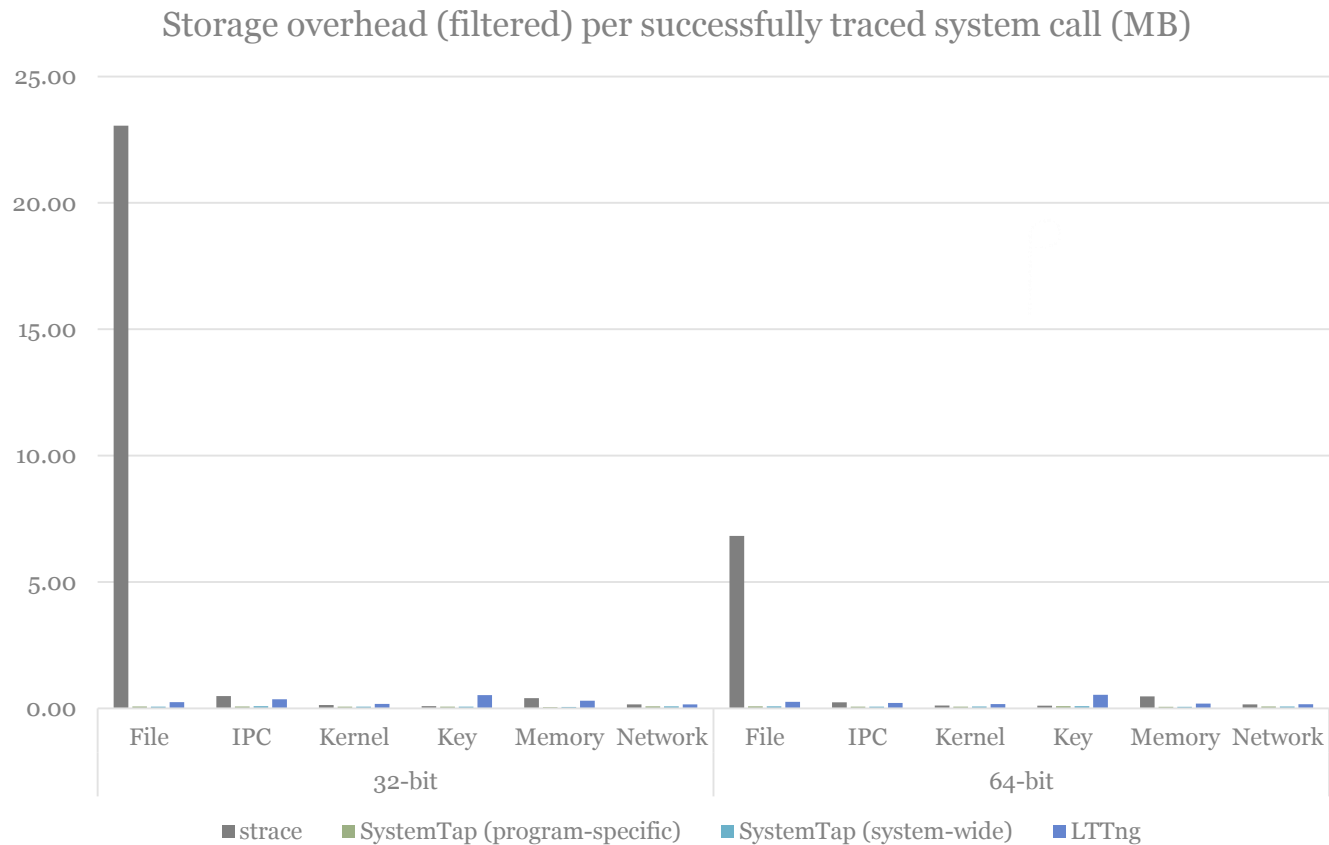
COST-BENEFIT ANALYSIS OF PROCESS TRACKING

COST ANALYSIS RESULTS: PERFORMANCE OVERHEAD



COST-BENEFIT ANALYSIS OF PROCESS TRACKING

COST ANALYSIS RESULTS: STORAGE OVERHEAD



COST-BENEFIT ANALYSIS OF PROCESS TRACKING

CONCLUSION

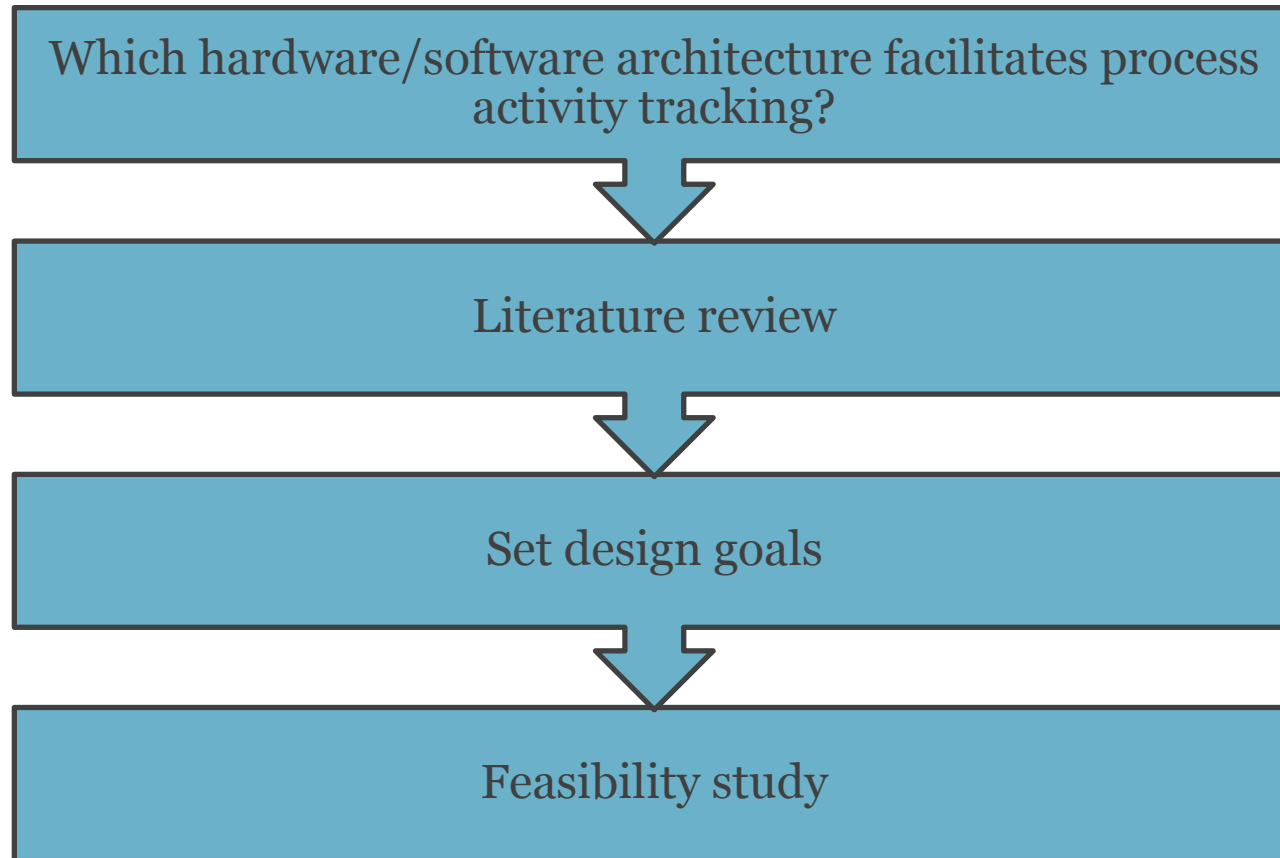
- **Kernel tracing systems can meet the two objectives of forensic readiness (Tan, 2001)**
 - maximize the capability of collecting credible digital evidence
 - minimize the cost of investigation
- **However**
 - high performance and storage overheads caused by dynamic instrumentation

COST-BENEFIT ANALYSIS OF PROCESS TRACKING

QUESTION RAISED

- **For cost-benefit trade-off, we need to**
 - design the architecture for flexible and adjustable process tracking
 - baseline monitoring through static instrumentation
 - » lower the impact on daily operations
 - malicious activities or policy violations discovered
 - » trace in detail through dynamic instrumentation

ARCHITECTURE FOR PROCESS TRACKING OVERVIEW



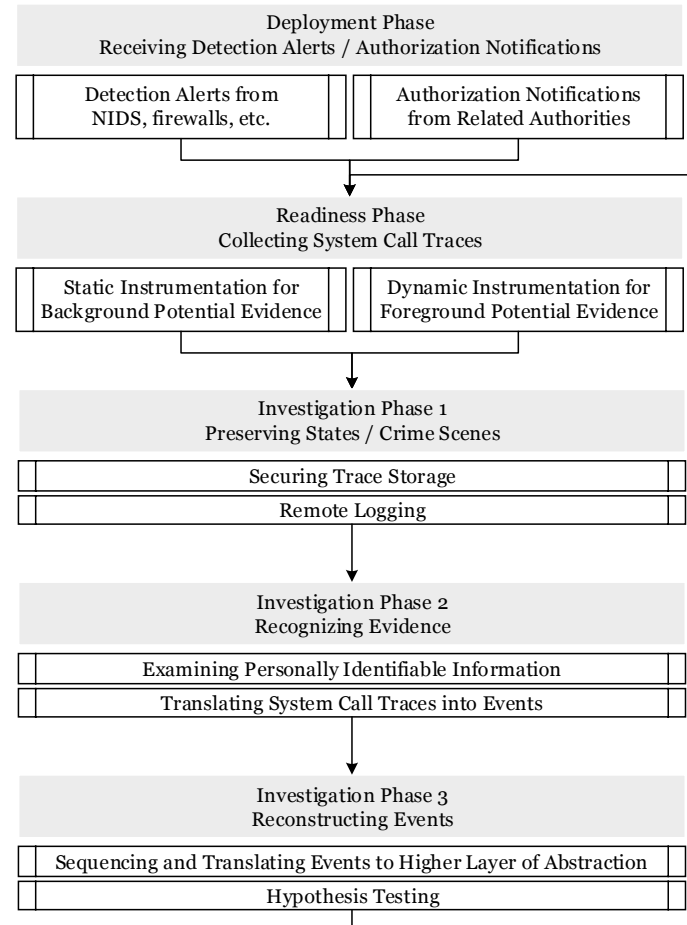
ARCHITECTURE FOR PROCESS TRACKING LITERATURE REVIEW

- **Digital event reconstruction systems**
 - insufficient coverage of system resource usage
 - process management, file system, and network
- **Digital forensics frameworks**
 - limited feasibility studies
 - two case studies about server intrusion and child pornography
 - lack of demonstration of implementation

ARCHITECTURE FOR PROCESS TRACKING DESIGN GOALS

- **Completeness**
 - able to trace anything regardless of expected incidents or crimes
- **Pertinence**
 - enable analysts and investigators to adjust and decide what to trace
- **Reliability**
 - ensure the potential evidence is readily available in a legal manner for admissibility
- **Privacy preservation**
 - protect the confidentiality of personally identifiable information from unauthorized access, use, and disclosure

ARCHITECTURE FOR PROCESS TRACKING PROTOTYPE FRAMEWORK



ARCHITECTURE FOR PROCESS TRACKING FEASIBILITY STUDY: READINESS PHASE

- **Collecting system call traces**
 - background potential evidence
 - LTTng
 - » static instrumentation
 - » lower the performance impact
 - » reduce storage overhead
 - foreground potential evidence
 - strace or SystemTap
 - » dynamic instrumentation
 - » for flexible and adjustable tracing

ARCHITECTURE FOR PROCESS TRACKING FEASIBILITY STUDY: DEPLOYMENT PHASE

- **Receiving detection alerts / authorization notifications**
 - strace or SystemTap
 - dynamic instrumentation
 - for flexible and adjustable tracing
 - future plan
 - automatic tracing level adjustment

ARCHITECTURE FOR PROCESS TRACKING FEASIBILITY STUDY: INVESTIGATION PHASE 1

- **Preserving states / crime scenes**
 - secure trace storage
 - access control mechanisms
 - remote logging
 - the GRR rapid response framework
 - » Advanced Encryption Standard (AES) 256-bit encryption
 - » authenticate the transmitter through X.509 certificate

ARCHITECTURE FOR PROCESS TRACKING FEASIBILITY STUDY: INVESTIGATION PHASE 2

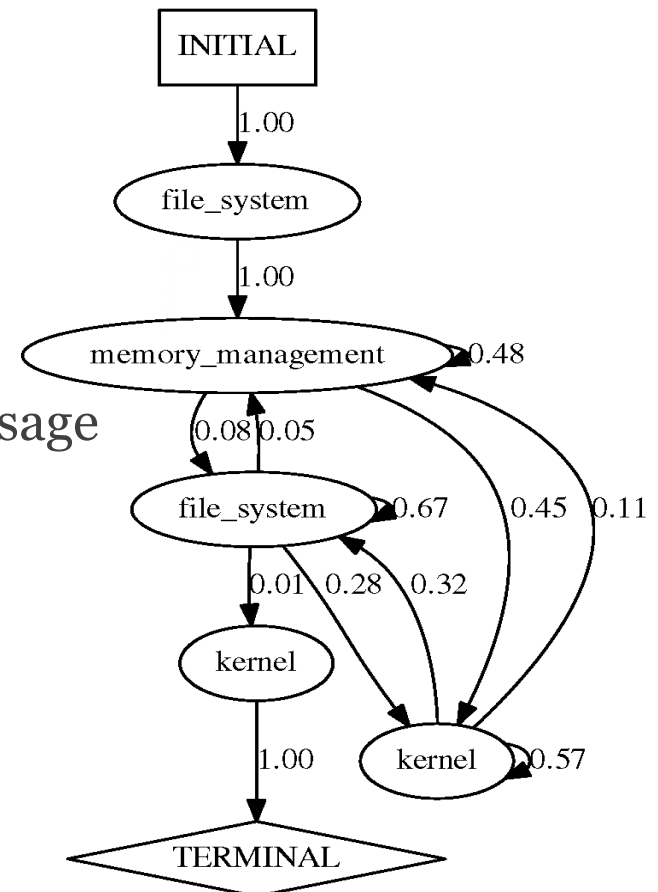
- **Recognizing evidence**

- examine personally identifiable information
 - host name and user name
- categorize system call traces according to the system resource they access
 - file system, inter-process communication, kernel, network, memory management, key (security), and drivers
- produce a system model similar to a finite state machine
 - Synoptic

ARCHITECTURE FOR PROCESS TRACKING FEASIBILITY STUDY: SYSTEM MODEL

- **Generated system model**

- ls command
- each edge label
 - transition probability
- the state transitions of resource usage



ARCHITECTURE FOR PROCESS TRACKING FEASIBILITY STUDY: INVESTIGATION PHASE 3

- **Reconstructing events**

- play back the system activity history as an animation
 - Gourc
 - » generates a dynamic tree to animate the software development history
 - » user who commits the update floating near the files
 - » color the update actions (add, modify, and delete)
 - » animate the history by the timelines

ARCHITECTURE FOR PROCESS TRACKING FEASIBILITY STUDY: ANIMATION

- **Generated animation**

- ls command
- user: process identifier
- file: resource
- update actions
 - extract verbs from manuals by natural language toolkit
 - » reconnaissance: green
 - » gaining access: red
 - » covering tracks: blue

Friday, 25 July, 2014 00:00:00

ARCHITECTURE FOR PROCESS TRACKING

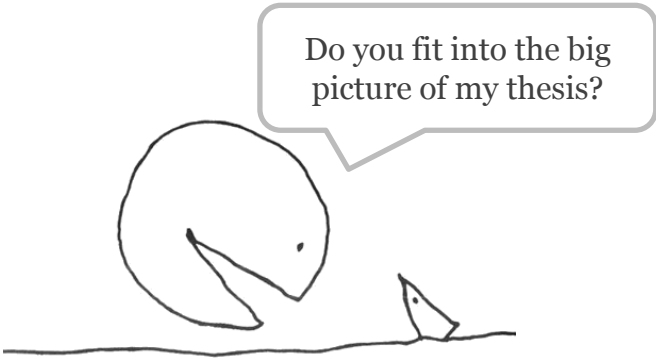
CONCLUSION

- **Employing kernel tracing systems in readiness phase of digital forensics frameworks can**
 - ensure the potential evidence is readily available in an acceptable form when an incident or a crime occurs

ARCHITECTURE FOR PROCESS TRACKING

QUESTION RAISED

- **Interpret the meaning of digital events**
 - cause and effect analysis
 - layers of abstraction
- **Support automatic tracing level adjustment**
- **Effectiveness evaluations**
 - legal and regulatory compliance
 - law enforcement investigation facilitation



Do you fit into the big picture of my thesis?

FUTURE WORK

- **Privacy Implications of Process Tracking**
 - What are privacy implications for users of systems that support comprehensive traceability?
 - privacy impact assessment
- **Admissibility of Process Tracking**
 - How does comprehensive traceability affect evidence gathering and the legal process?
 - vulnerability analysis
 - measurement of credibility

HØGSKOLEN I GJØVIK



Thank you

Yi-Ching Liao

Norwegian Information Security Laboratory



yi-ching.liao@hig.no

COINS Ph.D. student seminar 2014