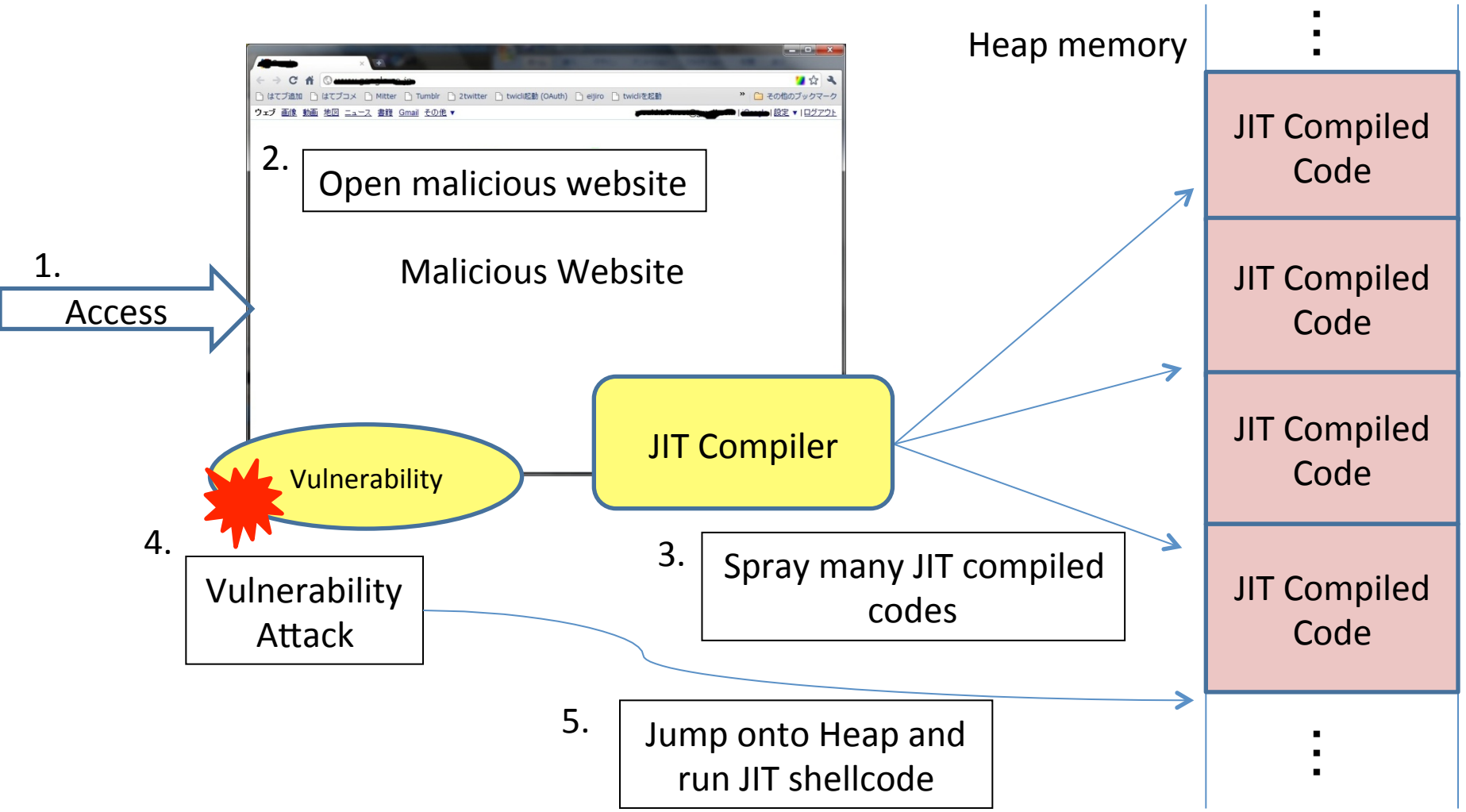
# Preventing execution of JIT shellcode by isolating running process

Ken Ichikawa, Kanta Matsuura
University of Tokyo

Works in Progress
ACSAC 27

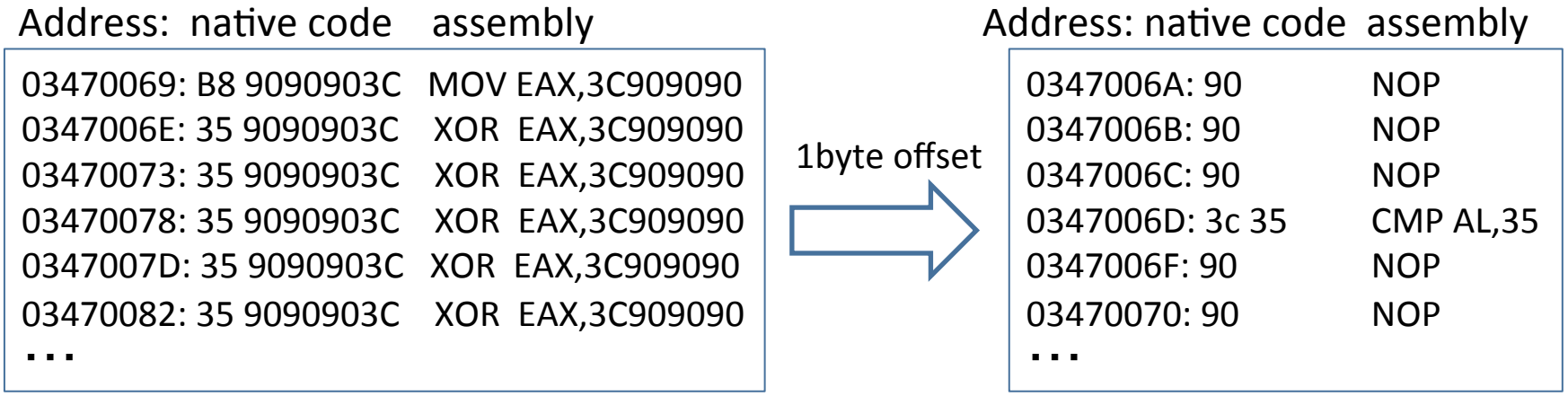
# JIT Compilation causes JIT Spraying




Bypass DEP(Data Execution Prevention ) and ASLR(Address Space Layout Randomization)

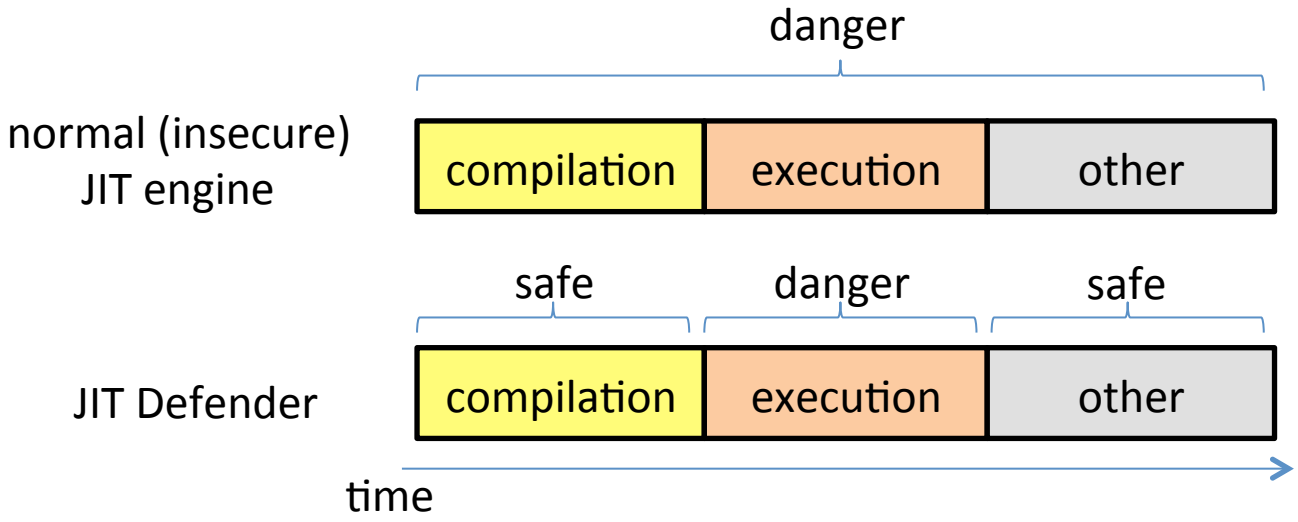
# JIT shellcode generation mechanism

- JIT compiler generates only safe codes seemingly but they can become shellcodes



# One of the Countermeasures

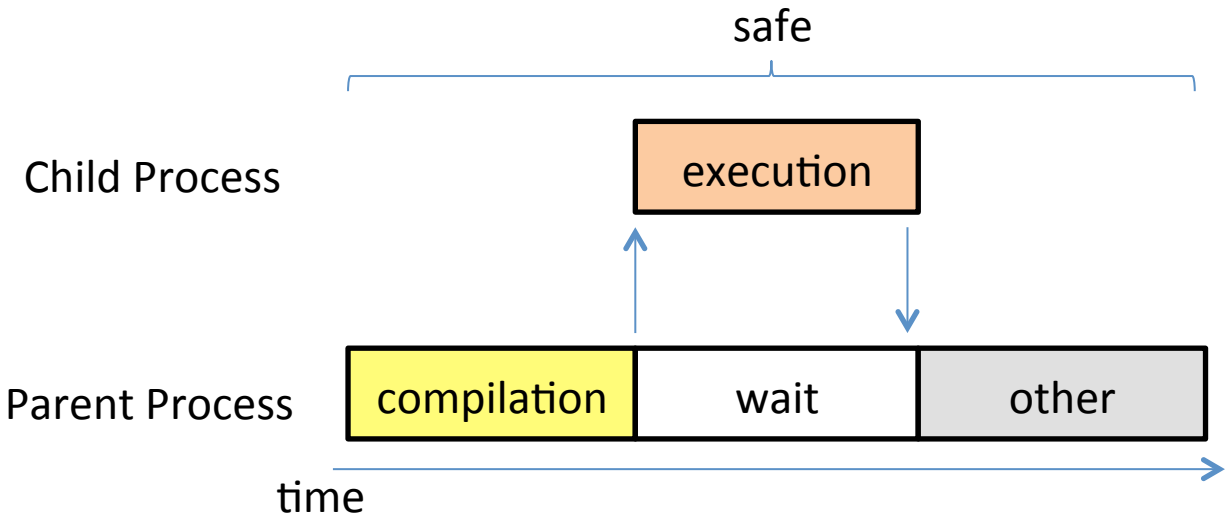
- JIT Defender[Ping Chen et al. 2011]
  - Adds executable attributes at only execution time
  - Very simple idea and very easy to implement
  - Has a problem
    - It's probable to be attacked at execution time




**Reference:** P. Chen, Y. Fang, B. Mao, and L. Xie, "JITDefender: A defense against jit spraying attacks," in Future Challenges in Security and Privacy for Academia and Industry, vol. 354 of IFIP Advances in Information and Communication Technology, pp. 142–153, Springer Boston, 2011.

# Our Proposal

- Process Isolation
  - Generates a child process just before execution of generated codes.
  - Parent process don't need to add executable attributes to the data area of the memory.



# Check for work of our proposal

- Attack success or failure

| Condition | Success or Fail |
|---|---|
| normal v8 (after execution) | Success |
| Normal v8 (while execution) | Success |
| process isolated v8 (after execution) | Failure |
| process isolated v8 (while execution) | Failure |

The checking method

1. Created JavaScript code that can become JIT shellcode after JIT compilation.
2. Modified a program using V8 to jump onto JIT shellcode location as a vulnerable program.

All conditions are turned ASLR off to be the program more vulnerable.

# Performance Evaluation

Don't trust this data. Our implementation has not completed yet.
However, process isolation itself can work.

- V8 benchmarks total score average

| Normal V8 | Process isolated V8 |
|---|---|
| 7067.8 | 7068.2 |

approximately-same 

- V8 benchmarks and V8 its own running time average

| Normal V8 | Process isolated V8 |
|---|---|
| 9.6002s | 9.6310s |

Only 0.3% overhead 

# Conclusion

- Our proposal, process isolation, can prevent JIT Spraying Attacks at any time.
- It seems that our proposal's running time overhead is marginal, so it's practical.

# Future Work

- Make certain our implementation.
- Consider the situation that multi threads of a parent call execution of generated codes.

# Thank you for your attention!

If you would like to contact us, please send E-mail to
ichik@iis.u-tokyo.ac.jp