Cache时间侧信道模拟实验

尹凌峰

清華大學
Tsinghua University

清華大學
Tsinghua University

Cache时间侧信道

- 当访存指令的目标地址在cache中时，可直接从cache中获取数据，减少指令执行时间。
- 从访存指令执行时间可以反推出目标地址是否在cache中。

- 利用cache时间侧信道，进程可以推断出cache的状态，从而了解到其它进程的访存情况。

Cache时间侧信道攻击

- 信息发送方（受害者）：可访问秘密信息m，将其编码为内存地址u，利用cache时间侧信道泄露u。
- 信息接收方（攻击者）：利用cache时间侧信道接收内存地址u，解码得到秘密信息m。

- Ex. 攻击者进程和受害者进程共享一个数组a。

Flush + Reload 攻击

1、攻击者使用flush指令从cache中清除a数组元素

2、受害者进程获取m后，访问a[m*512]。（u=m*512）

3、攻击者进程通过遍历u所有可能的位置（a[i*512]，i=0,1,…）。发生cache命中时就得到了u，解码得到m（m=u>>8）

实验目的

- 理解各类Cache侧信道攻击的工作机理
- 在实验框架内，实现Cache侧信道攻击，获取目标信息

- 在真实硬件上实现侧信道攻击
  - 受硬件环境、系统环境影响较大，难以形成统一评测标准
  - 测量单步访存操作延迟可能容易有误差，影响结果准确性
  - 易受到第三方进程的干扰

- 使用模拟Cache实现侧信道攻击
  - 统一的模拟环境
  - 接口直接返回准确的操作延迟
  - 不会受到第三方进程的干扰

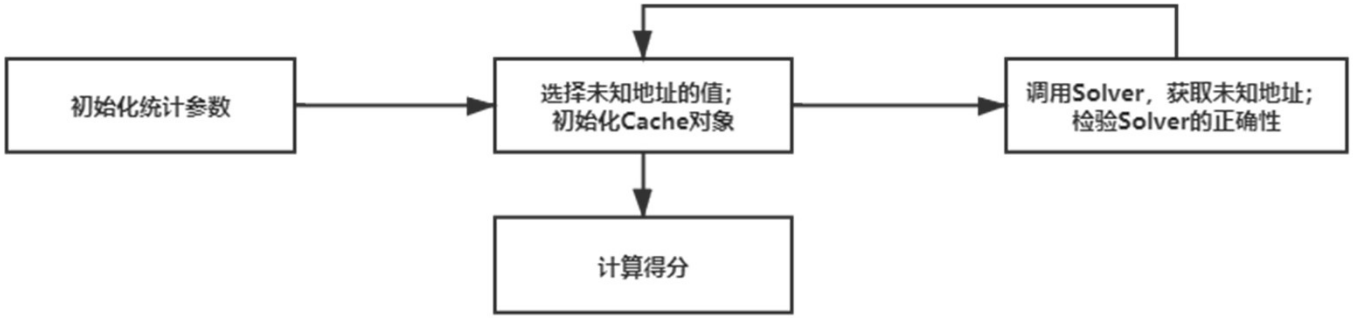
实验模型

- 攻击目标：一个用户无法直接获取的未知地址u
- 攻击者可行的操作
  - 操纵受害者进程访问、flush未知地址u
  - 操纵受害者进程访问、flush选定的地址
  - 操作受害者进程flush整个cache
  - 攻击者访问、flush选定地址
  - 攻击者flush整个cache

- 在模拟cache中，操纵受害者访问和攻击者直接操作从结果上没有区别，因此在实验中不区分操作主体

实验框架

- main.cpp
  - 调用solver、cache，测试各种情况下solver的正确性，计算得分

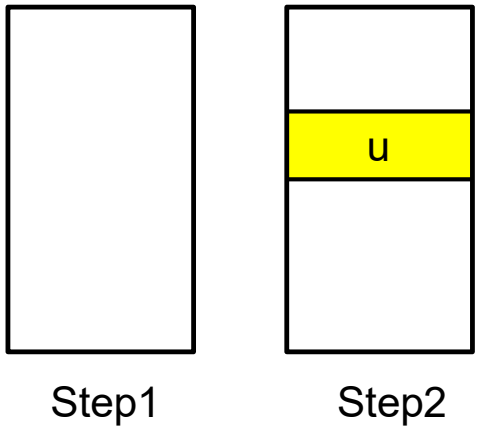


- cache.h、cache.cpp
  - 模拟cache，采用直接映射方式
  - 接口：access、flush、flushany、accessu、flushu
- solver.h、solver.cpp：
  - getAns接口，参数是模拟Cache、未知地址范围、Cache line数量。返回值是未知地址
  - getAns接口需实现

- Cache参数：直接映射Cache，32条Cache Line。
- 一个Cache Line只存储一个物理地址的数据。
- 物理地址范围[0,1600)，其中未知地址范围是[0,800)

实验任务

- 实现Solver类的getAns接口
  - 利用Cache时间侧信道，获取未知地址
  - 参考思路：Flush + Reload
    - 目标：判断未知地址u是否是某个地址a
    - 步骤1：从cache中清除a
    - 步骤2：访问未知地址u
    - 步骤3：访问a，测量时间延迟
    - 如果u=a，那么最后一步延迟低，否则延迟高



- 一些Cache时间侧信道攻击的论文，可供参考

[1] Yarom Y, Falkner K. FLUSH+ RELOAD: a high resolution, low noise, L3 cache side-channel attack

[2] Gruss D, Maurice C, Wagner K, et al. Flush+ Flush: a fast and stealthy cache attack

[3] Osvik D A, Shamir A, Tromer E. Cache attacks and countermeasures: the case of AES

[4] Neve M, Seifert J P. Advances on access-driven cache attacks on AES

- 思考题
  - 如何采取一些措施，防御你使用的cache时间侧信道攻击，并简述措施的有效性
  - 参考设计
    - PL cache、RP cache
      - New Cache Designs for Thwarting Software Cache-based Side Channel Attacks
    - Random Fill cache
      - Random Fill Cache Architecture
    - DAWG cache
      - DAWG: A Defense Against Cache Timing Attacks in Speculative Execution Processors
    - CEASER cache
      - CEASER: Mitigating Conflict-Based Cache Attacks via Encrypted-Address and Remapping
    - …

清華大學
Tsinghua University

提交文件

- 代码：solver.h/solver.cpp
  - 如果还需要其他文件，一并提交，包括修改后的makefile文件
- 实验报告
  - 攻击原理
  - 运行结果
  - 思考题

清華大學
Tsinghua University

注意事项

- 实验结果以助教测试结果为准
- main.cpp/cache.h/cache.cpp文件不会被覆盖，因此对于这三个文件的修改是无意义的
- 不建议使用与特定操作系统（linux、windows等）相关的库，否则在最终测试时可能无法编译

评价指标

- 10 = 7 + 1 + 2
  - 实验结果（7’）：成功获取到目标地址（3’）；访存次数（4’），见下表。
  - 代码（1’）：风格、注释等。基本给全。
  - 文档（2’）：攻击原理、运行结果截图；思考题。

总访存次数	得分
<=100,000	4
<=150,000	3.5
<=300,000	3
<=500,000	2.5
<=750,000	2
<=1,000,000	1

如何减少访存次数

- 利用多个cache line，同时开展多个攻击

- 使用多种侧信道攻击，减少未知地址的可能范围
  - Ex. Prime + Probe attack
- 利用上一次攻击后的状态，减少初始化步骤

- 如有疑问，可联系助教
- 邮箱：ylf17@mails.tsinghua.edu.cn