Flush+Reload

L3 Cache Side-Channel Attack

Aman Jain - 160050019
Meet Kathiriya - 160050001
Phuntsog Wangchuk - 160050109
Shreyas Pimpalgaonkar - 160050024
Shubham Anand - 160050060

CS 305 Course Project

Objectives

- Implement a side-channel attack on L3 cache of Intel x86 processors

Objectives

- Implement a side-channel attack on L3 cache of Intel x86 processors
- Attack based on information gained from the implementation of a computer system, rather than weaknesses in the implemented algorithm

Objectives

- Implement a side-channel attack on L3 cache of Intel x86 processors
- Attack based on information gained from the implementation of a computer system, rather than weaknesses in the implemented algorithm
- Exploit the weakness by monitoring access to memory lines in shared pages by Flush+Reload technique

Objectives

- Implement a side-channel attack on L3 cache of Intel x86 processors
- Attack based on information gained from the implementation of a computer system, rather than weaknesses in the implemented algorithm
- Exploit the weakness by monitoring access to memory lines in shared pages by Flush+Reload technique
- Extract private encryption keys by attacking the implementation of RSA algorithm in GnuPG 1.4.13 (an open-source implementation of a widely used encryption standard)

Objectives

- Implement a side-channel attack on L3 cache of Intel x86 processors
- Attack based on information gained from the implementation of a computer system, rather than weaknesses in the implemented algorithm
- Exploit the weakness by monitoring access to memory lines in shared pages by Flush+Reload technique
- Extract private encryption keys by attacking the implementation of RSA algorithm in GnuPG 1.4.13 (an open-source implementation of a widely used encryption standard)
- Explore measures to combat this attack

Objectives

- Implement a side-channel attack on L3 cache of Intel x86 processors
- Attack based on information gained from the implementation of a computer system, rather than weaknesses in the implemented algorithm
- Exploit the weakness by monitoring access to memory lines in shared pages by Flush+Reload technique
- Extract private encryption keys by attacking the implementation of RSA algorithm in GnuPG 1.4.13 (an open-source implementation of a widely used encryption standard)
- Explore measures to combat this attack
- Discuss possible implications

Flush + Reload Technique

- De-duplication of unrelated identical pages, copy-on-write mechanism and eviction of read-only pages from LLC

Flush + Reload Technique

- De-duplication of unrelated identical pages, copy-on-write mechanism and eviction of read-only pages from LLC
- Flush specific memory locations in L3 that correspond to the Square, Reduce and Multiply commands used by GnuPG,

Flush + Reload Technique

- De-duplication of unrelated identical pages, copy-on-write mechanism and eviction of read-only pages from LLC
- Flush specific memory locations in L3 that correspond to the Square, Reduce and Multiply commands used by GnuPG, wait until victim accesses that memory location,

Flush + Reload Technique

- De-duplication of unrelated identical pages, copy-on-write mechanism and eviction of read-only pages from LLC
- Flush specific memory locations in L3 that correspond to the Square, Reduce and Multiply commands used by GnuPG, wait until victim accesses that memory location, reload and exploit time difference

Flush + Reload Technique

- De-duplication of unrelated identical pages, copy-on-write mechanism and eviction of read-only pages from LLC
- Flush specific memory locations in L3 that correspond to the Square, Reduce and Multiply commands used by GnuPG, wait until victim accesses that memory location, reload and exploit time difference
- Need to choose an optimal wait time between two probes to minimize noise and slot misses, processor dependent.

Flush + Reload Technique

- De-duplication of unrelated identical pages, copy-on-write mechanism and eviction of read-only pages from LLC
- Flush specific memory locations in L3 that correspond to the Square, Reduce and Multiply commands used by GnuPG, wait until victim accesses that memory location, reload and exploit time difference
- Need to choose an optimal wait time between two probes to minimize noise and slot misses, processor dependent.
- Recovers 96.7 percent of encryption key in single run through

Implementation of attack on GnuPG

```
function exponent(b, e, m)
begin
  x ← 1
  for i ← |e| − 1 downto 0 do
    x ← x^2
    x ← x mod m
    if (e_i = 1) then
      x ← xb
      x ← x mod m
    endif
  done
  return x
end
```

Mitigation Techniques

Mitigation Techniques

- The use of *clflush* should be a privileged operation and should be limited to either programs with write permissions, or be paired with copy-on-write functionality.

Mitigation Techniques

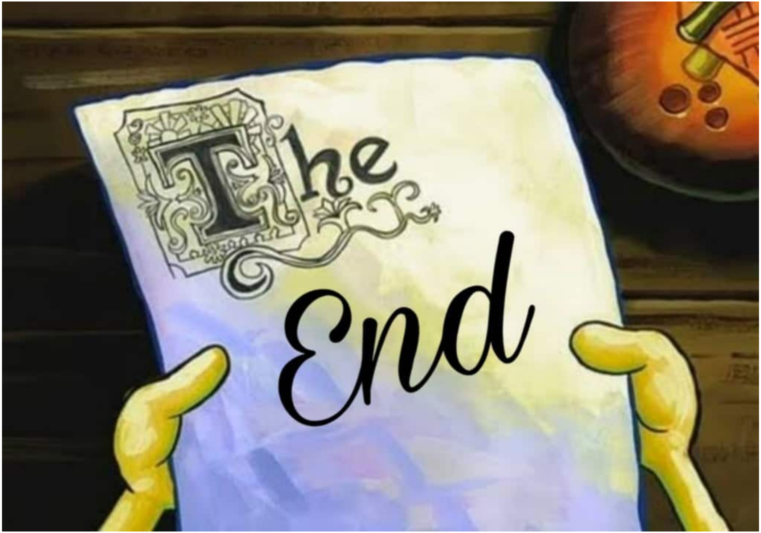
- The use of *clflush* should be a privileged operation and should be limited to either programs with write permissions, or be paired with copy-on-write functionality.
- Disallowing one process to flush a memory line from another process's cache region, or adopting non-inclusive caching will prevent possible attacks.

Mitigation Techniques

- The use of *clflush* should be a privileged operation and should be limited to either programs with write permissions, or be paired with copy-on-write functionality.
- Disallowing one process to flush a memory line from another process's cache region, or adopting non-inclusive caching will prevent possible attacks.
- Execution order of a process shouldn't be indicative of any sensitive information.

Possible Applications

- SSH, to extract the secret keys used for public-key authentication
- SSL, determining other users' private keys
- VIM, to determine what the victim is typing into a document
- Terminal commands (e.g. ls, cat, cd)


The
End