Practical Memory Deduplication Attacks in Sandboxed JavaScript

Daniel Gruss, David Bidner, and Stefan Mangard
IAIK, Graz University of Technology

September 23, 2015
Overview

- Page deduplication not only a problem in the cloud
- Can be used to eavesdrop on browser usage
Overview

- Page deduplication not only a problem in the cloud
- Can be used to eavesdrop on browser usage

The first page deduplication attack,

- in sandboxed JavaScript,
- on personal computers and smartphones,
- through malicious websites.
Overview

- Page deduplication not only a problem in the cloud
- Can be used to eavesdrop on browser usage

The first page deduplication attack,

- in sandboxed JavaScript,
- on personal computers and smartphones,
- through malicious websites.

Large scale remote attacks possible
Copy-on-Write

Virtual Address Space

Physical Address Space
Copy-on-Write

Virtual Address Space

Physical Address Space
Copy-on-Write

Virtual Address Space

Process A

Physical Address Space
Copy-on-Write

Process A

Virtual Address Space

Physical Address Space
Copy-on-Write

Virtual Address Space

Process A

Physical Address Space
Copy-on-Write

Virtual Address Space

Process A

Physical Address Space
Copy-on-Write

Virtual Address Space

Process A

fork

Process B

Physical Address Space
Copy-on-Write

Virtual Address Space

Process A

Process B

Physical Address Space
Copy-on-Write
Copy-on-Write

Virtual Address Space

Process A

Physical Address Space

Process B
Copy-on-Write

Virtual Address Space

Process A

Physical Address Space

Process B
Copy-on-Write

Process A

Virtual Address Space

Process B tries to write

Physical Address Space
Copy-on-Write

Process A

Virtual Address Space

Process B tries to write

copy

Physical Address Space
Copy-on-Write

Virtual Address Space

Process A

Process B

Physical Address Space

write
Write vs. Copy-on-Write

- Regular write access  $< 0.2\mu s$
- Write access with copy-on-write pagefault  $> 3.0\mu s$
- Clearly distinguishable
Page Deduplication

Virtual Address Space

Process A

Process B

Physical Address Space

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication

Virtual Address Space

Processes started independently

Physical Address Space
Page Deduplication

Virtual Address Space

Physical Address Space

Process A

Process B
Page Deduplication

Virtual Address Space

Process A

Physical Address Space

Process B
Page Deduplication

Virtual Address Space

Process A

Physical Address Space

Process B
Page Deduplication

Virtual Address Space

Process A

Process B

Physical Address Space
Page Deduplication

Virtual Address Space

Deduplication Thread

Physical Address Space

Process A

Process B
Page Deduplication

Virtual Address Space

Deduplication Thread

Physical Address Space
Page Deduplication

Virtual Address Space

Deduplication Thread

Physical Address Space

Process A

Process B
Page Deduplication

Virtual Address Space

Process A

Deduplication Thread

\(\neq\)

Process B

Physical Address Space

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication

Virtual Address Space

Deduplication Thread

Physical Address Space
Page Deduplication

Virtual Address Space

Process A

Process B

Physical Address Space

Deduplication Thread

≠
Page Deduplication

Virtual Address Space

Deduplication Thread

Process A

Process B

Physical Address Space
Page Deduplication

Virtual Address Space

Deduplication Thread

Physical Address Space
Page Deduplication

Virtual Address Space

Deduplication Thread

Physical Address Space
Page Deduplication

Virtual Address Space

Deduplication Thread

Physical Address Space

Process A

Process B
Page Deduplication

Virtual Address Space

Process A

Deduplication Thread

Process B

Physical Address Space

Done!
Page Deduplication

Virtual Address Space

Deduplication Thread

Physical Address Space
Page Deduplication

- Deduplication between processes:
  1. in same OS instance (Android, Windows)
  2. in different VMs (KVM, VMWare, ...)
- Code pages, data pages - even kernel pages
- Time until deduplication 2-45 minutes
  - depends on system configuration
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space
Page Deduplication Attack

Attacker generates a page suspected in process B.
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space
Page Deduplication Attack

Attacker waits for deduplication

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space
Page Deduplication Attack

Attacker waits for deduplication

t = time();
p[0] = p[0];
\Delta = time() - t;
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

$\Delta$ in $\mu$s

Time

measure $\Delta$
Page Deduplication Attack

Virtual Address Space

Attacker Process A

Benign Process B

Physical Address Space

$\Delta$ in $\mu$s

Time

$\Delta$ measured

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Attacker Process A

Virtual Address Space

\[ \Delta \text{ in } \mu s \]

Time

Benign Process B

Physical Address Space

\[ \text{measure } \Delta \]
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

$\Delta$ in $\mu$s

Time

measure $\Delta$
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

$\Delta$ in $\mu$s

Time

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

$\Delta$ in $\mu$s

Time

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Virtual Address Space

Attacker Process A

Benign Process B

Physical Address Space

\[ \Delta \text{ in } \mu s \]

Time

measure \( \Delta \)
Page Deduplication Attack

![Diagram showing page deduplication attack between Attacker Process A and Benign Process B.]

- **Attacker Process A**
- **Virtual Address Space**
  - Δ in µs
  - Time
  - Measure Δ
- **Physical Address Space**
- **Benign Process B**
Page Deduplication Attack

Virtual Address Space

Attacker Process A

Benign Process B

Physical Address Space

\[ \Delta \text{ in } \mu s \]

Time

\( \Delta \)

measure \( \Delta \)
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

\[ \Delta \text{ in } \mu s \]

Time

measure \( \Delta \)
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

\[ \Delta \text{ in } \mu s \]

Time

\[ \text{measure } \Delta \]
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

Δ in µs

Time

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

\[ \Delta \text{ in } \mu s \]

Time

\[ 0 \]

\[ 4 \]

measure \( \Delta \)
Page Deduplication Attack

![Diagram showing virtual and physical address spaces with attacker process A and benign process B.]
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

\[ \Delta \text{ in } \mu s \]

Time

\[ \text{measure } \Delta \]
Page Deduplication Attack

![Diagram showing virtual and physical address spaces, attacker process A, benign process B, and time difference measurement.]
Page Deduplication Attack

Virtual Address Space

Attacker Process A

Benign Process B

Physical Address Space

$\Delta$ in $\mu s$

Time

measure $\Delta$

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

Δ in µs

Time

measure ∆
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

$\Delta$ in $\mu s$

Time

$\neq$

measure $\Delta$

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Virtual Address Space

Attacker Process A

Physical Address Space

Benign Process B

\[ \Delta \text{ in } \mu s \]

Time

\[ \neq \]

measure \( \Delta \)

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

$\Delta$ in $\mu$s

0

4

Time

$\neq$

measure $\Delta$
Page Deduplication Attack

Virtual Address Space

Attacker Process A

Benign Process B

Physical Address Space

$\Delta$ in $\mu$s

Time

$\neq$

measure $\Delta$
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

Measure $\Delta$

Time

$\Delta$ in $\mu$s

$\neq$
Page Deduplication Attack

Attacker Process A

Virtual Address Space

\[ \Delta \text{ in } \mu s \]

Benign Process B

Physical Address Space

\[ \neq \]

Time

measure \( \Delta \)
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Δ in μs

Benign Process B

Physical Address Space

Time
Page Deduplication Attack

![Diagram showing the virtual and physical address spaces with attacker process A and benign process B.]
Page Deduplication Attack

Virtual Address Space

Attacker Process A

Benign Process B

Physical Address Space

$\Delta$ in $\mu s$

Time

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

- Attack Process A
- Virtual Address Space
  - Time
  - Write and measure $\Delta$
  - $\Delta$ in $\mu$s

- Physical Address Space

- Benign Process B
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

write and measure \( \Delta \)

copy

Time

\( \Delta \) in \( \mu s \)
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Attacker Process A

Virtual Address Space

\[ \Delta \text{ in } \mu s \]

Time

Benign Process B

Attacker learns that another process had an identical page

Physical Address Space
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Time

Attack learns that another process had an identical page

Physical Address Space

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

Attacker learns that another process had an identical page
Page Deduplication Attack

Attacker learns that another process had an identical page

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

Time

Attacker learns that another process had an identical page

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Attacker Process A

Virtual Address Space

Benign Process B

Physical Address Space

 attackers learns that another process had an identical page

$\Delta$ in $\mu$s

Time

0 4

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Page Deduplication Attack

Attacker Process A

Virtual Address Space

\[\Delta \text{ in } \mu s\]

Time

Attacker learns that another process had an identical page

Benign Process B

Physical Address Space
What can be attacked?

Existing Attacks:

- Detect binary versions in co-located VMs
- Detect downloaded image in Firefox under certain conditions

→ Attacks on hypervisors

- Native code only

*Suzaki et. al. 2011, Owens et. al. 2011, Xiao et. al. 2012, 2013*
What can be attacked?

Our Contribution:

- Detect CSS files and images of opened websites
  - Chrome, Firefox and Internet Explorer
- Perform the attack in JavaScript
  → Attacks on KVM, Windows 8.1 and Android
Attacking Browsers

- Images and CSS files are page-aligned in memory
- Load them into memory for all websites of interest
- Detect deduplication

→ Malicious ad networks: alternative to tracking pixels?
Detect Image (Native, Cross-VM, KVM)

- Image not loaded
- Image loaded

Cycles vs Page

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Challenges of JavaScript-based attacks

- No cycle counting (rdtsc)
- No access to virtual addresses
Page Deduplication Attacks in JavaScript

- Only require microsecond accuracy
  - `performance.now()` is accurate enough
  - Can even work with millisecond accuracy
    - Accumulate time difference
    - Only possible with enough image/CSS data
- Large typed arrays are allocated page-aligned
Detect Image (JavaScript, Cross-VM, KVM)

- Image not loaded
- Image loaded

Nanoseconds

Page

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Detect Image (JavaScript, Windows 8.1)

- Image not loaded
- Image loaded

Nanoseconds

Page

Image not loaded

Image loaded

Daniel Gruss, IAIK, Graz University of Technology
September 23, 2015
Detect Image (JavaScript, Android 4.4.4)

- Image not loaded
- Image loaded

![Graph showing image load times](image-url)
Detection of Open Websites

- Attacker chosen set of websites
- Load website images and CSS files into arrays
  - Served image and CSS files depend on: Browser, OS, resolution, etc.
  - → Reuse HTTP headers of system under attack
- Reduce noise by measuring write accesses to several pages
Detection of Open Websites

- Compare with:
  - zero pages (always deduplicated)
  - random pages (never deduplicated)

- Top 10 websites:
  Amazon, Baidu, Facebook, Google, QQ, Taobao, Twitter, Wikipedia, Yahoo, Youtube

- Examples for different platforms
Example: JavaScript, Cross-VM, KVM

Nanoseconds

Open Websites: Amazon, QQ, Yahoo
Example: JavaScript, Windows 8.1

Open Websites: Baidu, Google, Wikipedia, Yahoo
Example: JavaScript, Android 4.4.4

Open Websites: Google, QQ, Youtube
Countermeasures

JavaScript:

- Reduce timer accuracy?
- Prevent page-aligned arrays?
- Website diversification?
- Prevent control over full pages
  - Every $n$-th byte not part of JavaScript array
Countermeasures

JavaScript:

- Reduce timer accuracy?
- Prevent page-aligned arrays?
- Website diversification?
- Prevent control over full pages
  - Every $n$-th byte not part of JavaScript array

Generic:

- Disable page deduplication (for writable pages)
Conclusion

- Page deduplication not only a problem on IaaS clouds
  → Privacy issue on personal computers and smartphones
- Remote attacks through malicious websites
- Same code for all platforms → large scale attacks
- Disable page deduplication if possible
Practical Memory Deduplication Attacks in Sandboxed JavaScript

Daniel Gruss, David Bidner, and Stefan Mangard
IAIK, Graz University of Technology
September 23, 2015