A Case for Protecting Computer Games With SGX

Erick Bauman and Zhiqiang Lin
System and Software Security (S³) Lab
The University of Texas at Dallas

December 12th, 2016
Outline

1. Background
2. Overview
3. Detailed Design
4. Case Study
5. Conclusion
Computer Games

- Large industry, market value of tens of billions
- Popular games have millions of players
Cheat Prevention

- **Cheating in multiplayer games** serious concern for developers
- Small percentage of players can ruin experience for majority
Cheat Prevention

- A million-dollar industry
- Difficult to defend against
  - Cannot trust client machines
  - Server-side integrity checks often have high overhead
- Easy data duplication makes sharing applications trivial
- Many companies have strong interests in copy protection
- Piracy often costs billions in lost sales

**VOLUME AND COST OF HACKED GAMES IS GROWING**

Hacked game releases found between Jan. 2012 and Mar. 2015

<table>
<thead>
<tr>
<th>Year</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012</td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td></td>
</tr>
<tr>
<td>2014</td>
<td></td>
</tr>
<tr>
<td>2015*</td>
<td></td>
</tr>
</tbody>
</table>

**IN 2014, THE UNMONETIZED VALUE OF PIRATED GAMES REACHED $74,000,000,000**

Source: The Cheetah, 2014 Research
DRM: preventing circumvention of protection is hard

- Usually requires a trusted component on user’s machine
- Trusted component is protected by complex obfuscation, often quickly reverse-engineered
- Secrets are often too easily extracted without a way to truly secure them
Background

Ubisoft: DRM Can't Stop Piracy
VP of digital publishing says, "I don't want us in a position where we're punishing a paying player for what a pirate can get around."

Last updated by Eddie Makuch on June 20, 2014

Hacks! An investigation into the million-dollar business of video game cheating
By Emanuel Maiberg - April 30, 2014

Denuvo, the strongest game DRM available, has allegedly been cracked
By Tim Schiesser on Aug 10, 2016, 5:30 AM | 23 comments

Another Denuvo-protected game cracked just weeks after release
Quick Inside crack shows that industry's best DRM is no longer safe.
KYLE ORLAND - 8/28/2016, 10:05 AM
Intel SGX

- SGX’s secure enclaves provide strong guarantees to protect applications
  - Isolated execution environment
  - Contents unreadable by machine owner
  - Protection enforced by hardware
Why Intel SGX

Operating Systems

Hardware
Why Intel SGX

Operating Systems

Hardware
Why Intel SGX
Why Intel SGX

Operating Systems

Hardware
Why Intel SGX

Operating Systems

Hardware
Why Intel SGX

Operating Systems

Hardware
Why Intel SGX

Virtualization

Hardware

Operating Systems

Linux Kernel

Virtualization

Operating Systems

Linux Kernel

Hardware
Why Intel SGX

Virtualization

Operating Systems

Linux Kernel

Hardware
Why Intel SGX

Virtualization

Operating Systems

Linux Kernel

Hardware
Why Intel SGX

Virtualization
- Operating Systems
- Linux Kernel

Hardware
Why Intel SGX

Virtualization

Operating Systems

Linux Kernel

Hardware

SGX
Why Intel SGX

- Operating Systems
- Virtualization
- Hardware
  - SGX
Why Intel SGX
Key SGX Features of Interest
Outline

1. Background
2. Overview
3. Detailed Design
4. Case Study
5. Conclusion
Scope and Assumptions

**Scope: Computer Games**

- Multiplayer games for **cheat prevention**
- Single and multiplayer games for **DRM**
Scope and Assumptions

**Scope: Computer Games**
- Multiplayer games for **cheat prevention**
- Single and multiplayer games for **DRM**

**Assumptions and Threat Model**
- An attacker may have **full control over all software except for trusted enclaves**
- Attacker may access all memory, but not the processor
- We assume SGX itself is secure
Protection Model

Integrity: Crucial for Cheat Prevention

**Data Integrity**
- Prevent disallowed modifications to data
- Protect code that does modify data
- Provide limited interface for modifying data

**Code Integrity**
- Prevent modifications to crucial code, e.g. validation code
- Move necessary code to enclave
Protection Model
Confidentiality: Crucial for DRM

Data Confidentiality
- Any data decrypted inside enclave remains hidden
- If data must be shown to user, it may potentially be extracted from memory without secure I/O
- If code that touches data can reside entirely inside enclave, data can remain hidden

Code Confidentiality
- More challenging than code integrity
- Enclave code can be read before enclave is instantiated
- Code must be dynamically decrypted in enclave at runtime
- Can result in complete black box for user
### Protection Model

#### Examples

<table>
<thead>
<tr>
<th><strong>Integrity</strong></th>
<th><strong>Confidentiality</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data</strong></td>
<td></td>
</tr>
<tr>
<td>Game State:</td>
<td>Media Content:</td>
</tr>
<tr>
<td>Score, lives,</td>
<td>sounds, textures</td>
</tr>
<tr>
<td>orientation,</td>
<td>3D models</td>
</tr>
<tr>
<td>map inventory</td>
<td>configuration data</td>
</tr>
<tr>
<td>items</td>
<td></td>
</tr>
<tr>
<td>player position</td>
<td></td>
</tr>
<tr>
<td><strong>Code</strong></td>
<td></td>
</tr>
<tr>
<td>Integrity Checks:</td>
<td>Game Logic:</td>
</tr>
<tr>
<td>Velocity Checks</td>
<td>Algorithms</td>
</tr>
<tr>
<td>Collision Detection</td>
<td>Scripts</td>
</tr>
</tbody>
</table>
## Desired Properties for Protected Content

### Isolated
- Enclaves prohibit certain instructions, e.g. system calls
- Enclave code must be isolated from the application code
- Data sent across enclave boundary must be copied
- Presents a challenge to port existing applications to SGX!
Desired Properties for Protected Content

**Isolated**
- Enclaves prohibit certain instructions, e.g. system calls
- Enclave code must be isolated from the application code
- Data sent across enclave boundary must be copied
- Presents a challenge to port existing applications to SGX!

**Crucial**
- Enclaves have a limited amount of memory available
- An enclave too large for EPC will hurt performance
- The larger the code in enclave, the greater the risk of vulnerability or side channel
Outline

1. Background
2. Overview
3. Detailed Design
4. Case Study
5. Conclusion
Protecting Integrity

Key Ideas

- Multiplayer games must have one or more game servers
- Server-side integrity checks may be expensive
- SGX allows a single, one-time check of enclave integrity
- After attestation, all signed or encrypted messages from the enclave can be trusted without further checks
- Code and data inside enclave can therefore be trusted
Protecting Integrity

User Platform

Application

Enclave

Authentication Server

Game Server
Protecting Integrity

User Platform

Application

Enclave

Authentication Server

Game Server

1

2
Protecting Integrity
Protecting Integrity

User Platform
  Application
    Enclave

Authentication Server
  1
  2

Game Server
  3
  4
Protecting Integrity: Recap

Detailed Steps

1. Start Remote Attestation
2. Verify Enclave
3. Share Credentials
4. Enclave Communicates with Game Server
Protecting Confidentiality

Key Ideas

- Content can be protected by encryption
- All data decrypted inside enclave is secure
- Key to decrypt content can be withheld until proof of purchase is given
- Authentication server gives decryption key only after successful attestation and license key is given
- After initial license check, enclave can seal key to allow resource decryption without contacting server
Protecting Confidentiality

User Platform

Application

User Interface

Enclave

Encrypted Resources

File Systems

Sealed key

Encrypted Resource Files

Authentication Server

Overview

Detailed Design

Case Study

Conclusion
Protecting Confidentiality

User Platform

Application

User Interface

Enclave

Encrypted Resources

File Systems

Sealed key

Encrypted Resource Files

Authentication Server
Protecting Confidentiality

User Platform
- Application
  - User Interface
- Enclave
  - Encrypted Resources

Authentication Server

File Systems
- Sealed key
- Encrypted Resource Files

1. Connection from User Platform to Authentication Server
2. Connection from Authentication Server to File Systems
3. Connection from User Interface to Enclave
Protecting Confidentiality

User Platform

Application

User Interface

Enclave

Encrypted Resources

File Systems

Sealed key

Encrypted Resource Files

Authentication Server

1 2 4 5 3
Protecting Confidentiality

User Platform

Application

User Interface

Enclave

Encrypted Resources

File Systems

Sealed key

Encrypted Resource Files

Authentication Server

1 2 4 5 6

Sealed key Encrypted Resource Files

File Systems

Encrypted Resources

Enclave

User Interface

Application

User Platform

Authentication Server
Protecting Confidentiality

User Platform

Application

User Interface

Enclave

Encrypted Resources

File Systems

Sealed key

Encrypted Resource Files

Authentication Server
Protecting Confidentiality

User Platform

Application

User Interface

Enclave

Encrypted Resources

File Systems

Sealed key

Encrypted Resource Files

Authentication Server

1. [Authentication Server]
2. [User Platform]
3. [Application]
4. [Enclave]
5. [File Systems]
6. [Sealed key]
7. [Encrypted Resources]
8. [User Interface]
Protecting Confidentiality: Recap

Detailed Steps

1. Start Remote Attestation
2. Verify Enclave
3. Retrieve License Key
4. Send License Key
5. Receive Decryption Key
6. Retrieve Encrypted Assets
7. Decrypt Assets
8. Seal Decryption Key
Challenges

- Each game requires protection of different content (i.e., Protection is game specific)
- Partitioning is difficult
  - Existing games not designed with isolated component
  - Many code dependencies
  - Can lead to too much code in enclave
  - Difficult to balance enclave size with securing enough code and data
- Many assets will be leaked due to lack of secure I/O
Objectives

Port Real Game to SGX

Open-source game Biniax2, consisting of over 3500 lines of C
Objectives

Applying Our Framework
Focus on **DRM protection mechanisms** since the game does not support networked multiplayer

Protecting Assets
Prevent assets from being loaded until the encryption key is provided

Operating Systems
Virtualization
Hardware
SGX
Modifications

- Partitioned application into trusted and untrusted components
- Modified asset handling code to load encrypted assets
  - 923KB of images
  - 160KB of sound effects
  - 14KB of text
- Provided proof-of-concept confidentiality protection for assets
## Performance

Table: Comparison of several metrics between the original Biniax2 game and our modified version that we ported to SGX.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Biniax2</th>
<th>SGX-Biniax2</th>
<th>Increase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lines of Code</td>
<td>3540</td>
<td>4326</td>
<td>22.20%</td>
</tr>
<tr>
<td>Initialization Time (ms)</td>
<td>141.58±4.23</td>
<td>243.59±4.11</td>
<td>72.05%</td>
</tr>
<tr>
<td>Binary Size (bytes)</td>
<td>35038</td>
<td>38353</td>
<td>9.46%</td>
</tr>
<tr>
<td>Asset Size (bytes)</td>
<td>1084486</td>
<td>1097259</td>
<td>1.18%</td>
</tr>
</tbody>
</table>
## Performance

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lines of Code in Enclave</td>
<td>580</td>
</tr>
<tr>
<td>Enclave Size (bytes)</td>
<td>100425</td>
</tr>
<tr>
<td>Enclave Initialization (ms)</td>
<td>53.22±4.21</td>
</tr>
<tr>
<td>Assets Encrypted</td>
<td>29</td>
</tr>
</tbody>
</table>

**Table:** Statistics for our modified SGX-Biniax2.
Future Work

- Encrypt secrets that never need to leave enclave
- Fully demonstrate attestation, sealing, and unsealing
- Perform case study for cheat prevention
- Further analyze security implications of enclave applications and how to prevent implementation vulnerabilities
SGX provides an excellent opportunity for protecting games and applications.

We demonstrated a general framework that takes a first step in using SGX for DRM and cheat prevention.

We performed a case study showing the feasibility of our approach.
Thank You