Now Playing in Plaintext

Extracting Audio from Hardware-Backed DRM

Björn Ruytenberg, Mohammad Sina Karvandi,

Herbert Bos, Erik van der Kouwe, Asia Slowinska

Vrije Universiteit Amsterdam

Security in Times of Surveillance – Eindhoven, The Netherlands
June 26, 2026

Who Am I

Björn Ruytenberg
@0Xiphorus@infosec.exchange

- PhD Candidate @ VUSec, Vrije Universiteit Amsterdam
- Security Researcher, HyperDbg lead maintainer
- x86-64 UEFI, hypervisor and PCI Express security
- Previous work: critical Intel Thunderbolt vulnerabilities (thunderspy.io)
- More info: bjornweb.nl

THUNDERSPY

Summary

Industry Assumptions

- DRM protects high value audio/video streaming content
- Analog capture and digital output interception requires special hardware; introduces noise and quality loss
- Hardware-backed DRM (e.g. Widevine L1, PlayReady SL3000) believed unbreakable due to TEE-based secure pipeline

Our Work – Key Insights

- Decrypted audio MUST eventually traverse the device I/O boundary to reach playback hardware
- DReaMcatcher intercepts device I/O, while hiding itself from DRM – no hardware needed
- Core DRM architecture never anticipated hypervisor-level device I/O interception

Software-based DRM

- 2010s: streaming media popularized by Spotify, Netflix, Hulu, Amazon Instant Video
- Previous work extensively studied software-based DRM, finding mostly design flaws and cryptographic weaknesses
- Examples: Steal This Movie (MovieStealer), youtube-dl (yt-dlp), WideLeak, Narrowbeer

Wang, Ruoyu; Shoshitaishvili, Yan; Kruegel, Christopher; Vigna, Giovanni (August 2013). "Steal This Movie: Automatically Bypassing DRM Protection in Streaming Media Services". *22nd USENIX Security Symposium (USENIX Security 13)*. pp. 687–702.

Hybrid SW/HW-based DRM

- Security-Explorations

Modern DRM Solutions

Low

Google Widevine L3

Software-only, in-browser decryption – history of public bypass tools

Medium

Microsoft PlayReady SL2000

Partial TEE-based decryption – one documented, but not publicly disclosed vulnerability [1]

High

Google Widevine L1

Full TEE-based decryption – **no known vulnerabilities**

Max

Microsoft PlayReady SL3000

Full TEE-based decryption; kernel, device drivers and hardware attestation – **no known vulnerabilities**

[1] Security-Explorations – Microsoft WarBird and PMP (2024)

Let’s bypass hardware-based DRM

Goal: extract lossless audio from commercial DRM systems operating at maximum security settings

- ✓ **Full DRM security protocol bypass**
- ✓ **All major streaming vendors impacted (Netflix, Amazon Prime Video, Spotify, …)**
- ✓ **Fully software-based attack**
- ✓ **No implementation flaws exploited, no licensing servers compromised**
- \- **Requires privileged adversary, but in-scope for DRM vendor threat model**

Modern DRM Solutions

Low

Google Widevine L3

Software-only, in-browser decryption – history of public bypass tools

Medium

Microsoft PlayReady SL2000

Partial TEE-based decryption – one documented, but not publicly disclosed vulnerability [1]

High

Google Widevine L1

Full TEE-based decryption – **no known vulnerabilities**

Max

Microsoft PlayReady SL3000

Full TEE-based decryption; kernel, device drivers and hardware attestation – **no known vulnerabilities**

[1] Security-Explorations – Microsoft WarBird and PMP (2024)

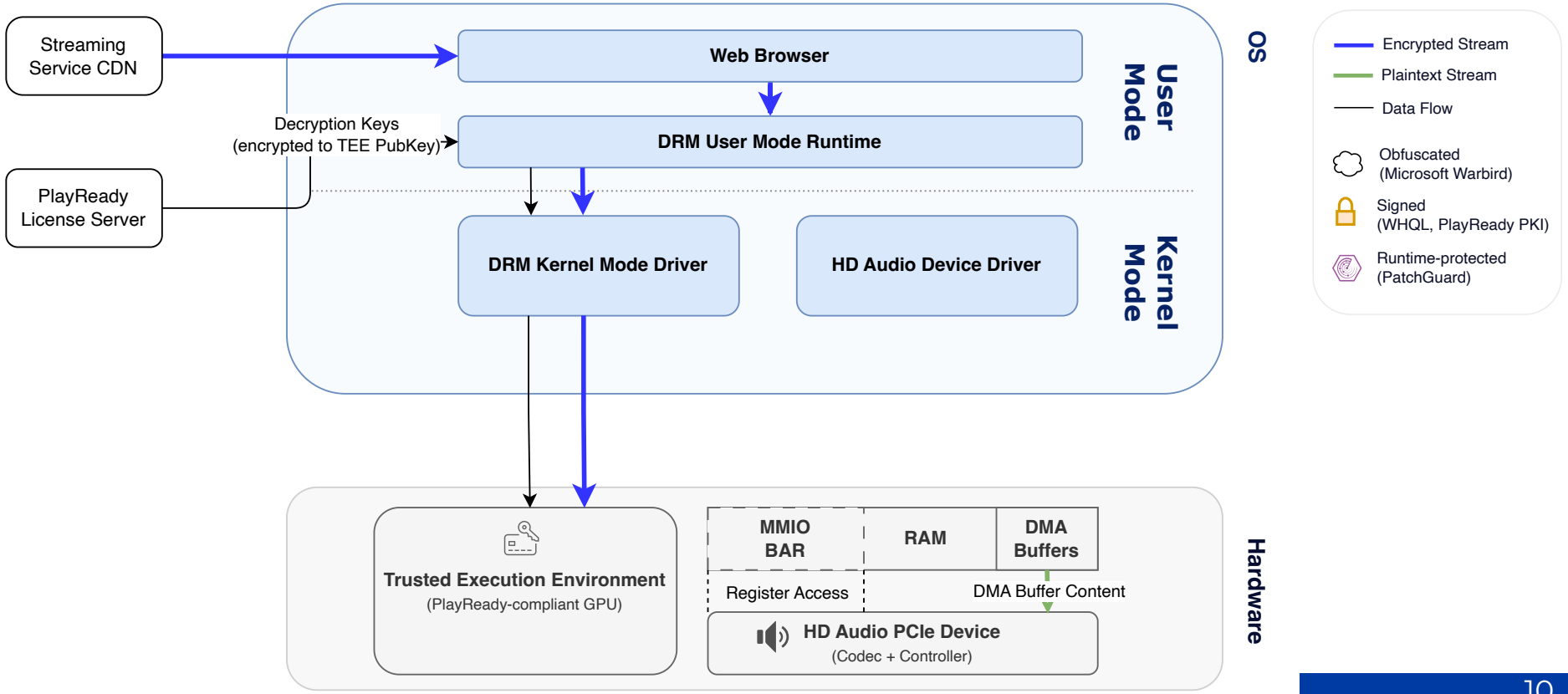
Modern DRM Solutions

DReaMcatcher: Full Bypass

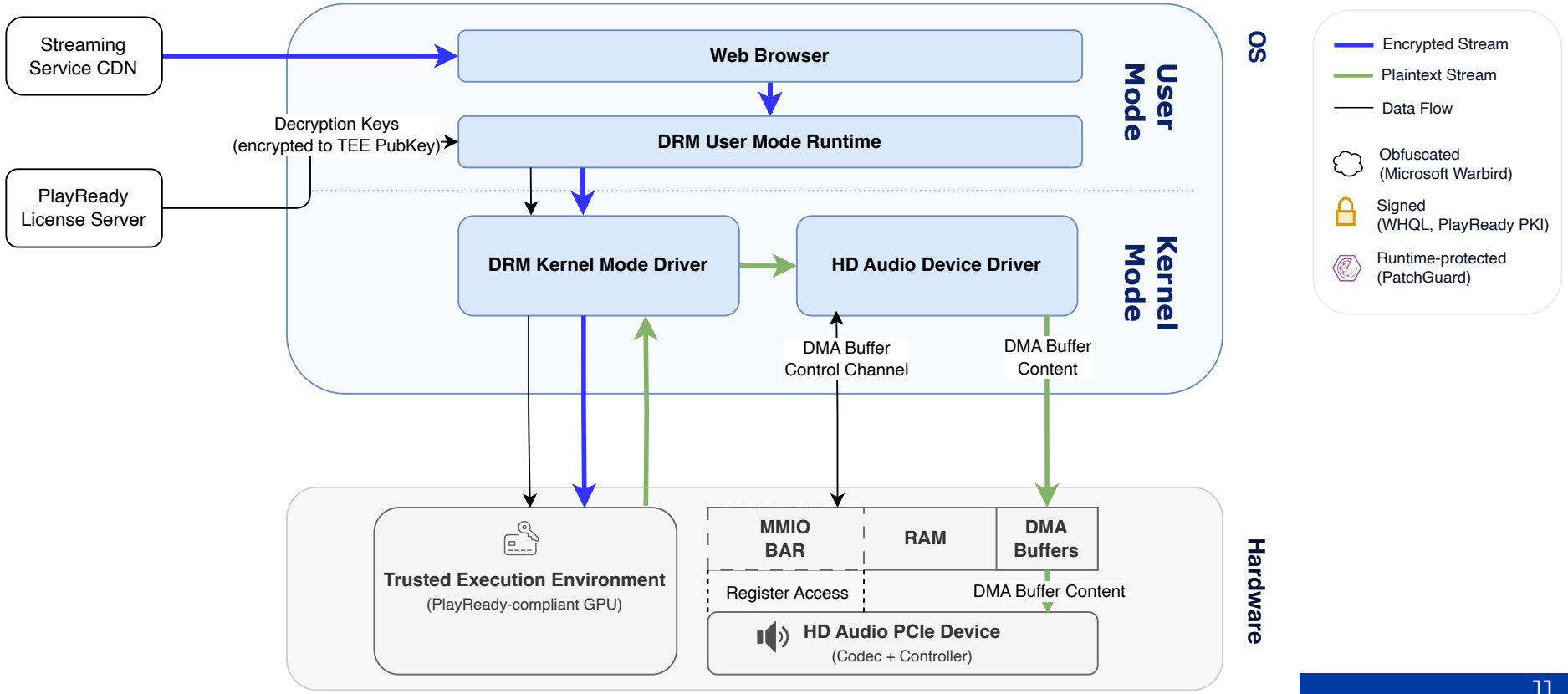
Low	**Google Widevine L3** Software-only, in-browser decryption – history of public bypass tools	✓
Medium	**Microsoft PlayReady SL2000** Partial TEE-based decryption – one documented, but not publicly disclosed vulnerability [1]	✓
High	**Google Widevine L1** Full TEE-based decryption – **no known vulnerabilities**	✓
Max	**Microsoft PlayReady SL3000** Full TEE-based decryption; kernel, device drivers and hardware attestation – **no known vulnerabilities**	✓

[1] Security-Explorations – Microsoft WarBird and PMP (2024)

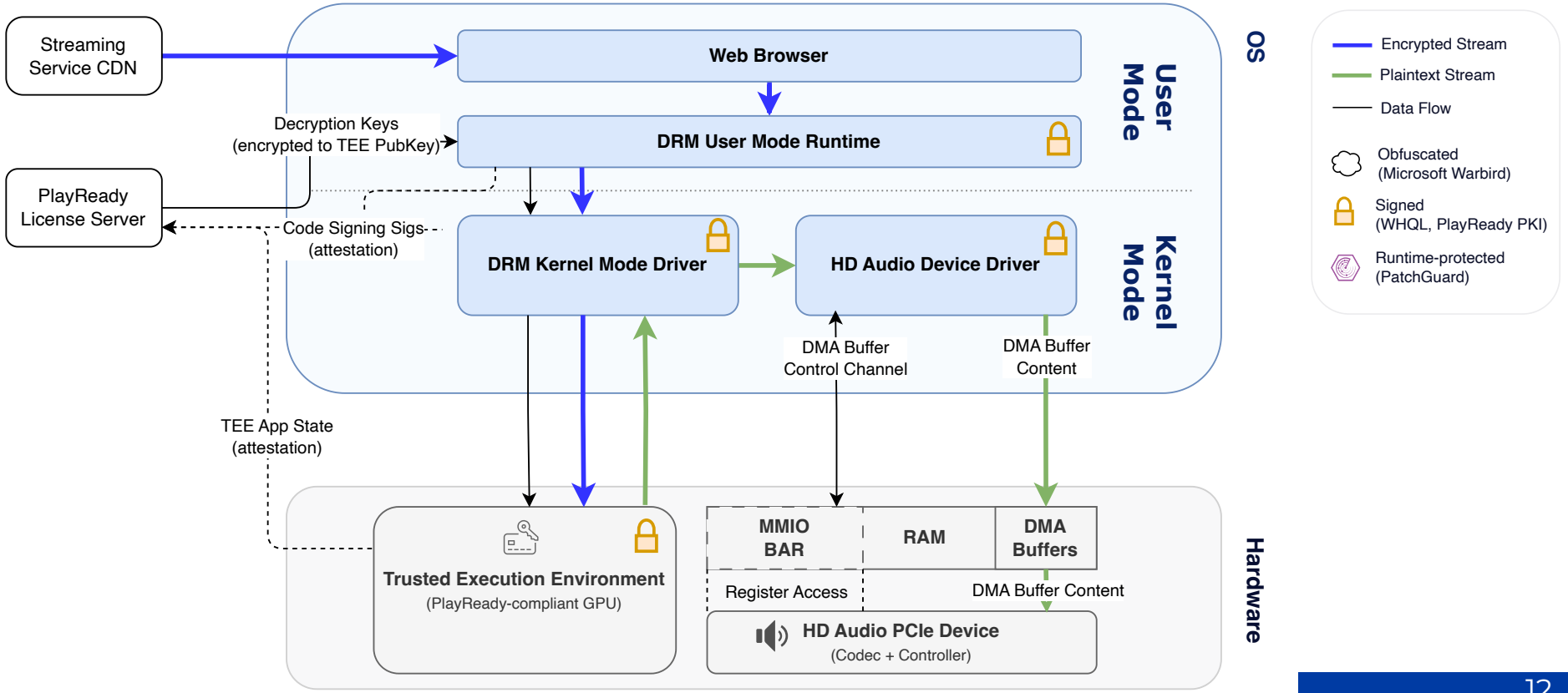
Modern DRM Architecture



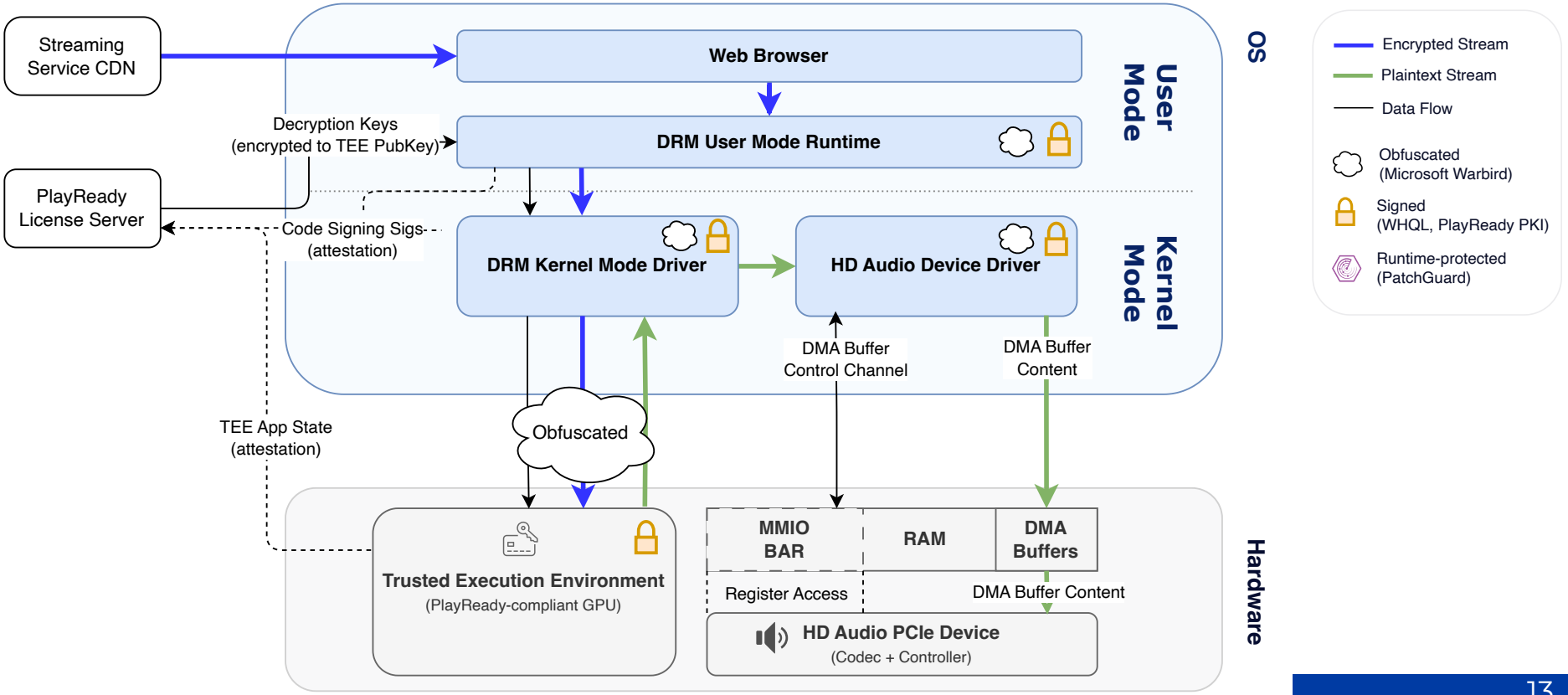
Modern DRM Architecture



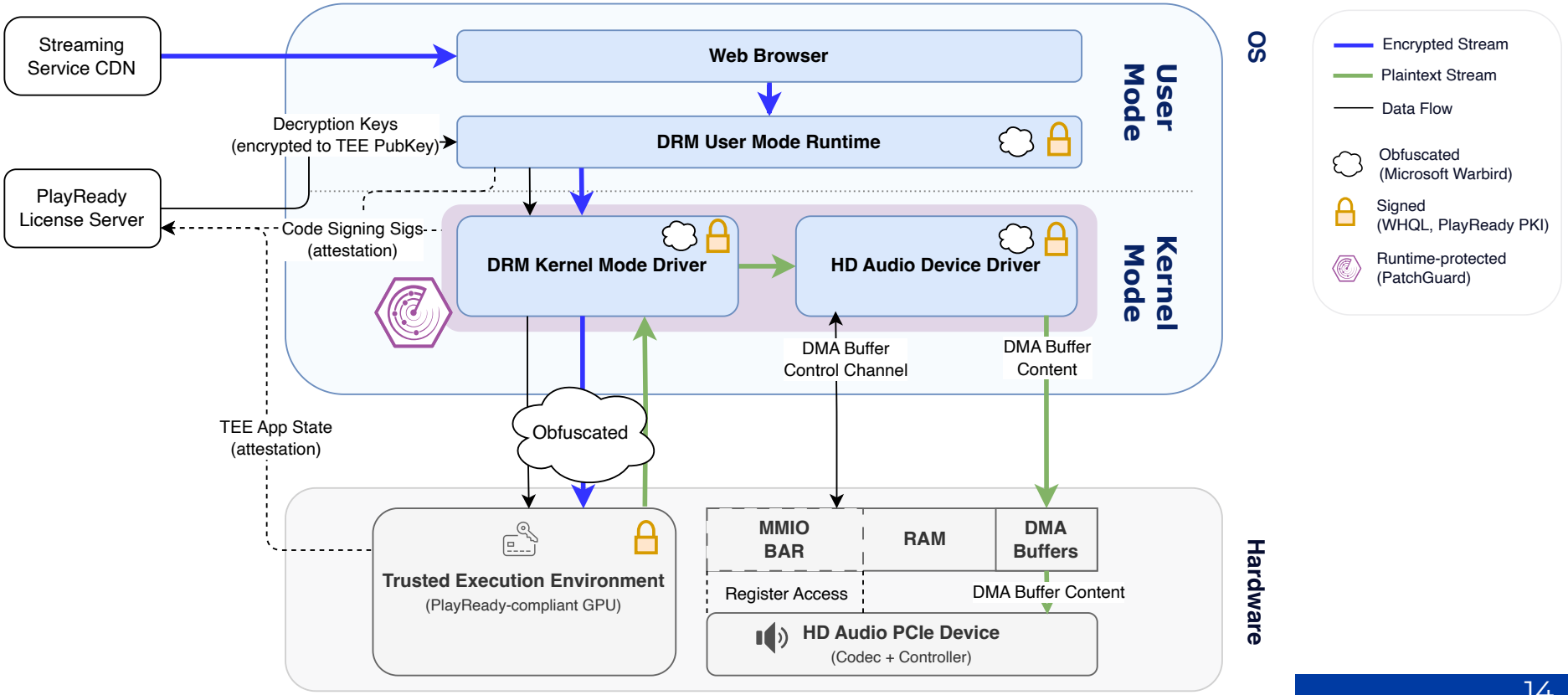
Modern DRM Architecture



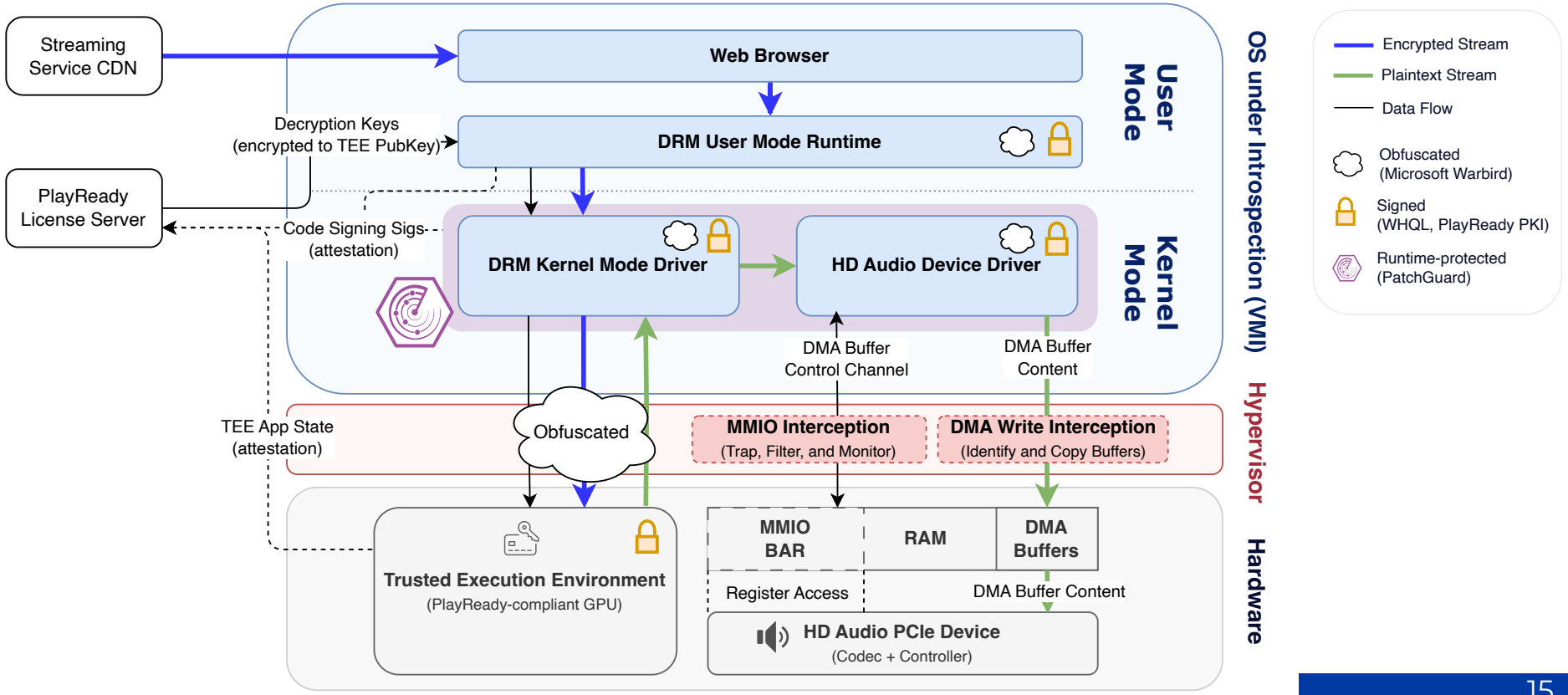
Modern DRM Architecture



Modern DRM Architecture



Modern DRM Architecture

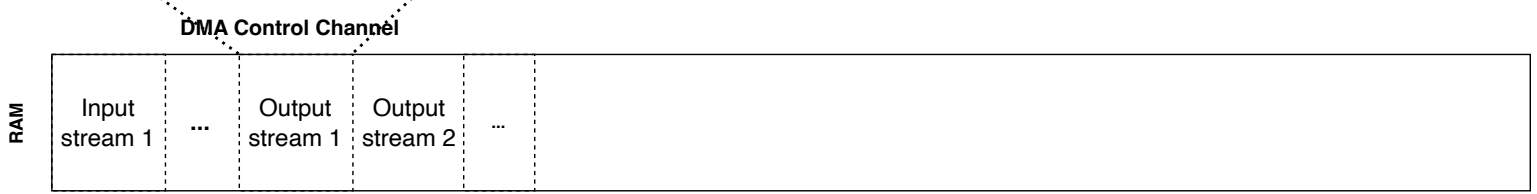


DReaMcatcher: Three-Phase Design

1

Discover DMA Buffers

Offset	Size	Description
0x00	1 byte	Stream Control byte 0
...	...	...
0x03	1 byte	Stream Status
...	...	
0x12	2 bytes	Stream Format
0x18	4 bytes	BDL Lower Base Addr
0x1C	4 bytes	BDL Upper Base Addr



HD Audio PCIe Device - MMIO BAR

DReaMcatcher: Three-Phase Design

1

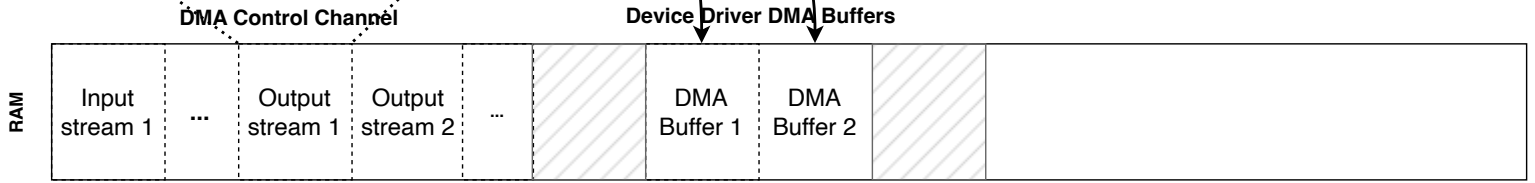
Discover DMA Buffers

2

Monitor Driver & Detect Buffer Completions

Offset	Size	Description
0x00	1 byte	Stream Control byte 0
...	...	...
0x03	1 byte	Stream Status
...	...	
0x12	2 bytes	Stream Format
0x18	4 bytes	BDL Lower Base Addr
0x1C	4 bytes	BDL Upper Base Addr

Hypervisor synchronizes on DMA Control Registers



HD Audio PCIe Device - MMIO BAR

HDA Device Driver

DReaMcatcher: Three-Phase Design

1

Discover DMA Buffers

Offset	Size	Description
0x00	1 byte	Stream Control byte 0
...	...	...
0x03	1 byte	Stream Status
...	...	
0x12	2 bytes	Stream Format
0x18	4 bytes	BDL Lower Base Addr
0x1C	4 bytes	BDL Upper Base Addr

2

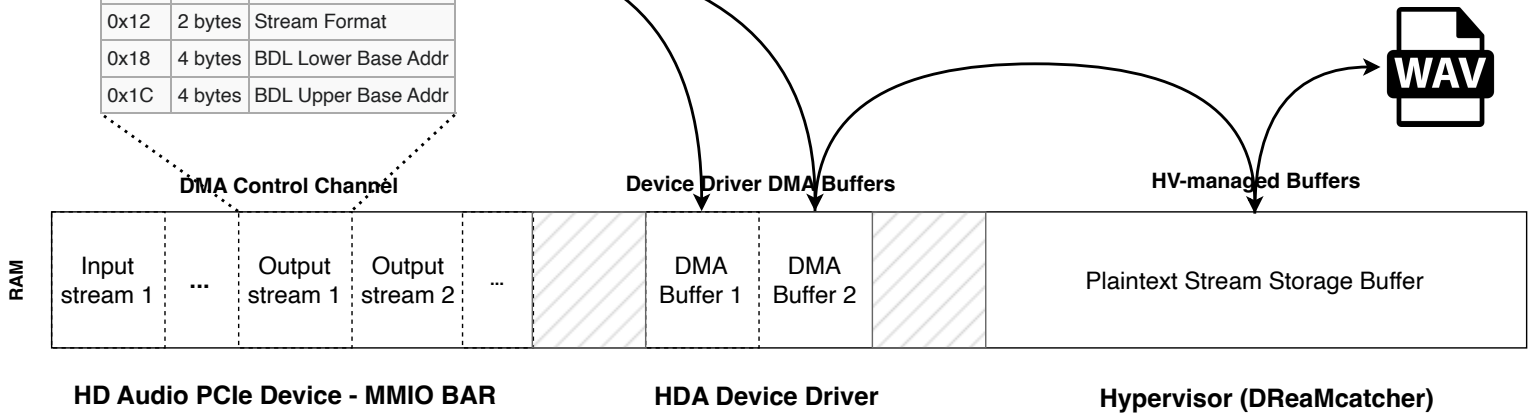
Monitor Driver & Detect Buffer Completions

Hypervisor synchronizes on DMA Control Registers

3

Extract & Reconstruct Stream

Hypervisor reconstructs PCM bitstream as WAV file



How EPT Hooks Enable DReaMcatcher

Extended Page Tables (EPT)

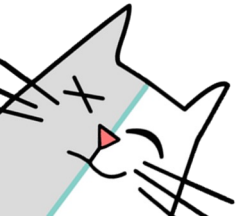
- Intel’s implementation for second-level address translation
- Marking EPT entries as non-readable enables hooking I/O

EPT Hook Mechanism

1. DReaMcatcher marks MMIO physical memory as non-readable
2. Upon MMIO write, DReaMcatcher freezes driver, inspects access, dumps DMA buffer
3. Temporarily re-enables driver execution
4. Restores physical memory non-readable state → ready for next transfer

HyperDbg Debugger

- FOSS (GPLv3) hypervisor-assisted debugger
- Leverages hardware virtualization controls to deliver advanced debugging capabilities (e.g., EPT-based memory monitoring, system call interception, PMIO/MMIO debugging)
- Operates independently of OS-level debugging APIs, providing higher transparency than traditional debuggers
- First released for Windows (2022), actively maintained since
  - Linux-based hypervisor agent work in progress – visit our website to track progress!



Get the source code:
github.com/HyperDbg/HyperDbg

Impacted platforms (selected)

Platform	DRM	Audio Format	Result
Netflix (Edge)	**PlayReady SL3000 (GPU)**	**xHE-AAC 48kHz stereo**	**✓ Success**
Netflix (Chrome)	**Widevine L1 (Intel SGX)**	**xHE-AAC 48kHz stereo**	**✓ Success**
Spotify	Widevine L3	AAC 44.1kHz 16-bit	✓ Success
Spotify Lossless	**Spotify Playplay**	**FLAC up to 16-bit/44.1kHz**	**✓ Success**
Apple Music Lossless (App)	**Unidentified**	**ALAC up to 24-bit/48kHz**	**✓ Success**
Amazon Prime Video	Widevine L3	AAC 48kHz 16-bit stereo	✓ Success
HBO Max	Widevine L3	AAC 48kHz stereo	✓ Success
Apple TV+ (App)	Unidentified	AAC / Dolby Atmos	✓ Success

Potential Mitigations and Challenges

Limit audio to HDMI HDCP-protected outputs only

Excludes devices without GPUs; incompatible with most speakers/headphones; HDCP subject to known vulnerabilities

HD Audio device: extend end-to-end secure pipeline to on-device TEE

Renders ALL in-market HD Audio devices obsolete

HD Audio device: to verify I/O path integrity, extend hardware attestation to device

Complex deployment, and equally renders all in-market HD Audio devices obsolete

Move decryption pipeline into confidential VM (Intel TDX / AMD-SEV)

Requires CPU support absent in most consumer devices; requires UEFI lockdown; TDX+SEV subject to known vulnerabilities

Employ hypervisor detection and anti-debugger techniques

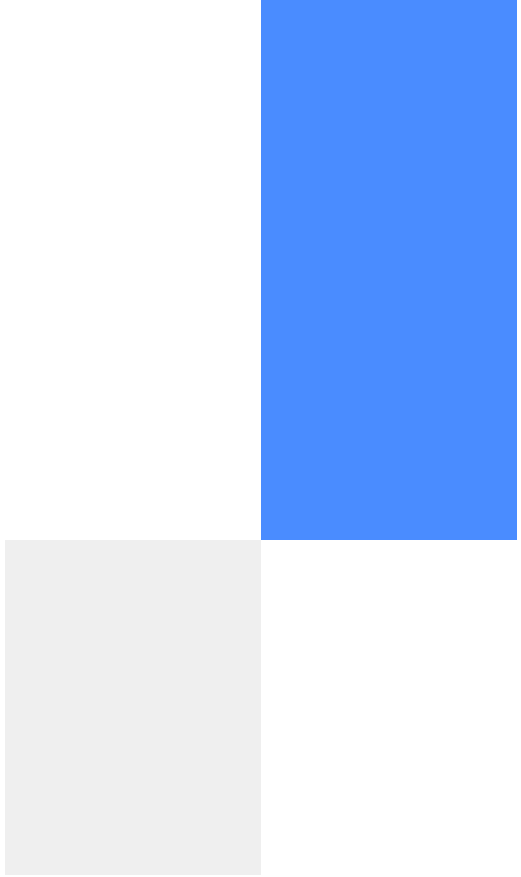
Skilled adversaries can bypass; security through obscurity; Sony rootkit precedent shows privacy and attack surface concerns

Demo

Proof-of-Concept

DReaMcatcher dumping audio from Netflix - Microsoft PlayReady SL3000

Demo



DRM Security & Privacy

A high-level overview

Sony Audio CD rootkit (2005)

- Windows rootkit shipped with Sony BMG audio CDs, installed without user consent
- Hijacks operating system internals to hide its presence
- Patches CD/DVD block device access – restricts audio CD access to Sony's own player application, returns garbage data otherwise
- Created attack surface for malware: hijacks userspace binaries to escalate privileges to Administrator, and
  - Inject itself into kernel space to compromise OS security
  - Hide its own presence, even from anti-virus solutions

Lessons from the Sony CD DRM Episode

J. Alex Halderman and Edward W. Felten
Center for Information Technology Policy
Department of Computer Science
Princeton University

Russinovich, Mark (31 October 2005). "Sony, Rootkits and Digital Rights Management Gone Too Far".
BBC News (11 November 2005). "Viruses use Sony anti-piracy CDs".
Halderman, J. Alex; Felten, Edward W. (August 2006). "Lessons from the Sony CD DRM Episode". *Proceedings of the 15th USENIX Security Symposium*. Vancouver, B.C., Canada: USENIX Association. pp. 77–92.

DRM trend: BYO ~~rootkit~~ vulnerable driver

Microsoft Security Bulletin MS07-067 - Important

Vulnerability in Macrovision Driver Could Allow Local Elevation of Privilege (944653)

Published: December 11, 2007 | Updated: December 11, 2007

EXPLOIT DATABASE

Macrovision SafeDisc - 'SecDRV.SYS' Method_Neither Privilege Escalation

EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
30680	2007-5587	ELIA FLORIO	LOCAL	WINDOWS	2007-10-18

EDB Verified: ✓

Exploit: / {}

Vulnerable App:

https://exploit-db.com/exploits/30680

NIST NATIONAL VULNERABILITY DATABASE NVD

CVE-2018-7249 Detail

Description

An issue was discovered in secdrv.sys as shipped in Microsoft Windows Vista, Windows 7, Windows 8, and Windows 8.1 before KB3086255, and as shipped in Macrovision SafeDisc. Two carefully timed calls to IOCTL 0xCA002813 can cause a race condition that leads to a use-after-free. When exploited, an unprivileged attacker can run arbitrary code in the kernel.

NotSecDrv - A PoC code for CVE-2018-7249

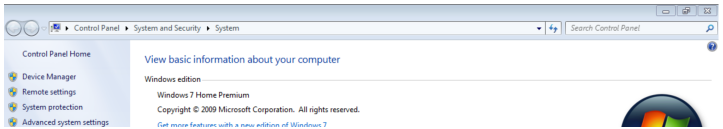
General Description

An issue was discovered in secdrv.sys as shipped in Microsoft Windows Vista, Windows 7, Windows 8, and Windows 8.1 before KB3086255, and as shipped in Macrovision SafeDisc. Two carefully timed calls to IOCTL 0xCA002813 can cause a race condition that leads to a use-after-free. When exploited, an unprivileged attacker can run arbitrary code in the kernel.

The vulnerability was reported to Microsoft, and since it does not affect an up-to-date Windows machine (only version prior to KB3086255), they will not take any action. Was tested and exploited successfully on Windows 7 x86.

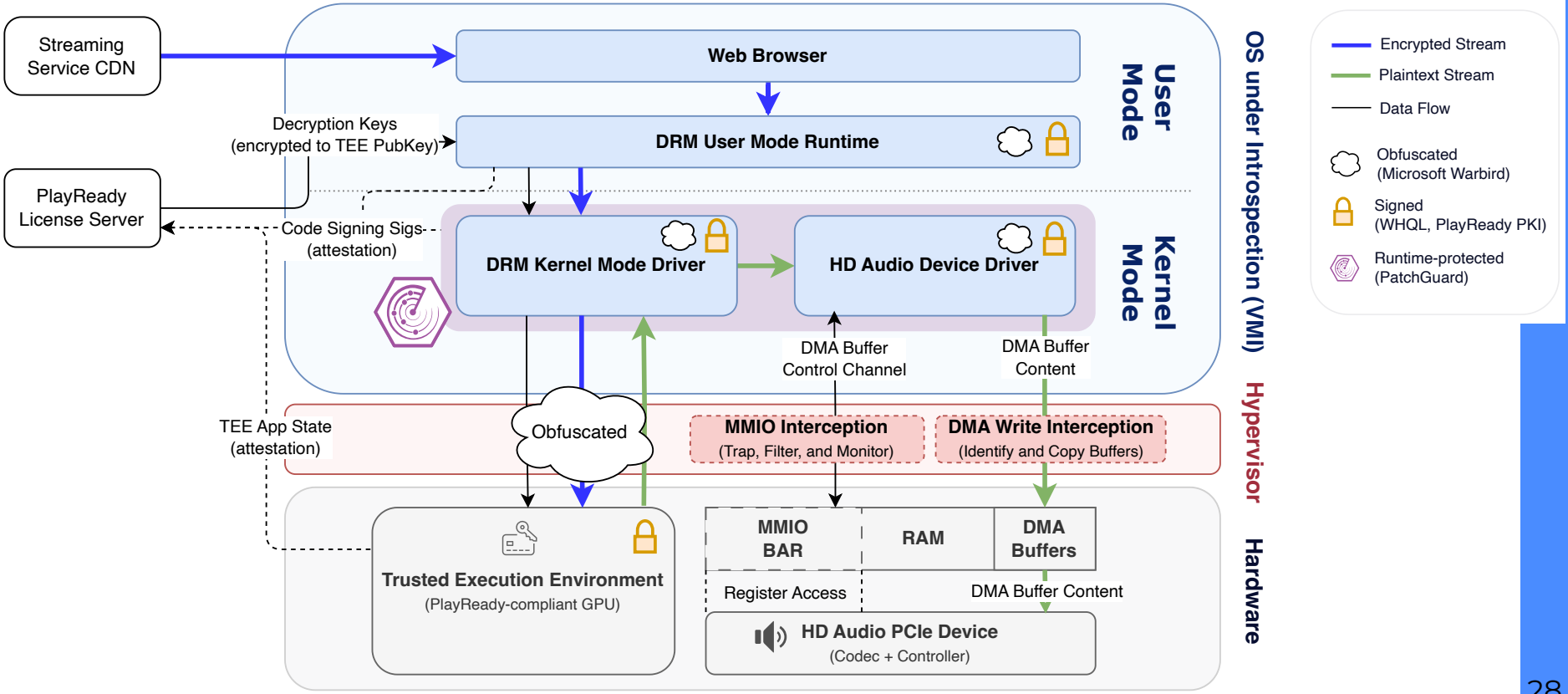
Also related to CVE-2018-7250.

Screenshot

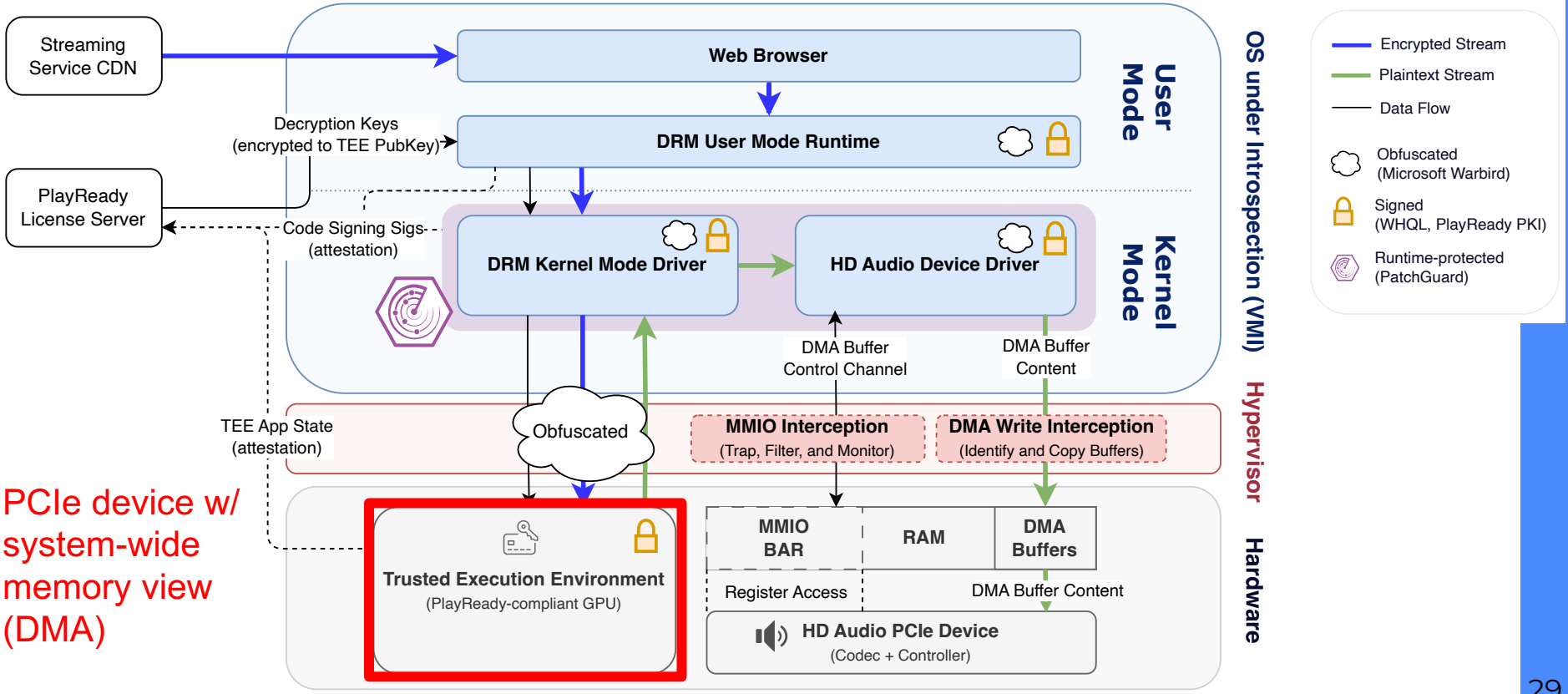


https://github.com/Elvin9/NotSecDrv

DRM trend: move into highly privileged TEE

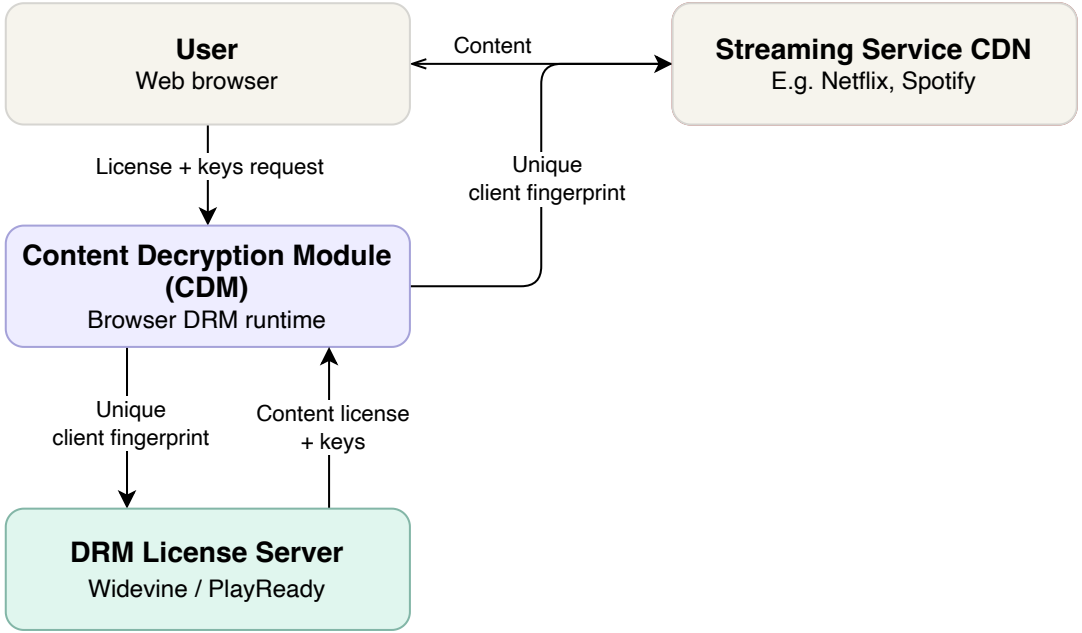


DRM trend: move into highly privileged TEE

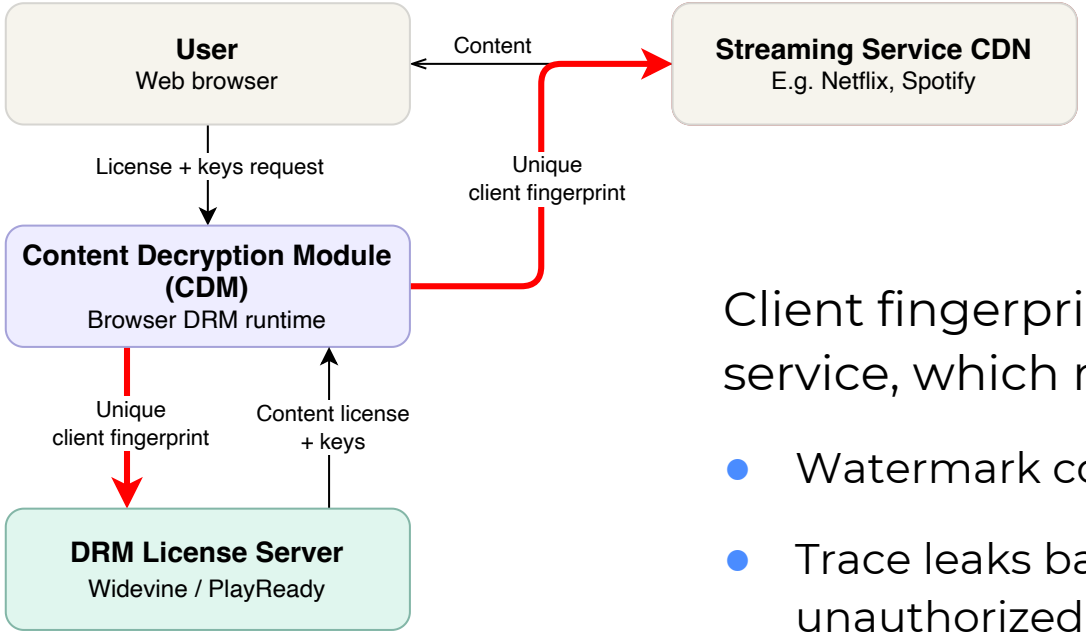


PCIe device w/ system-wide memory view (DMA)

DRM privacy implications



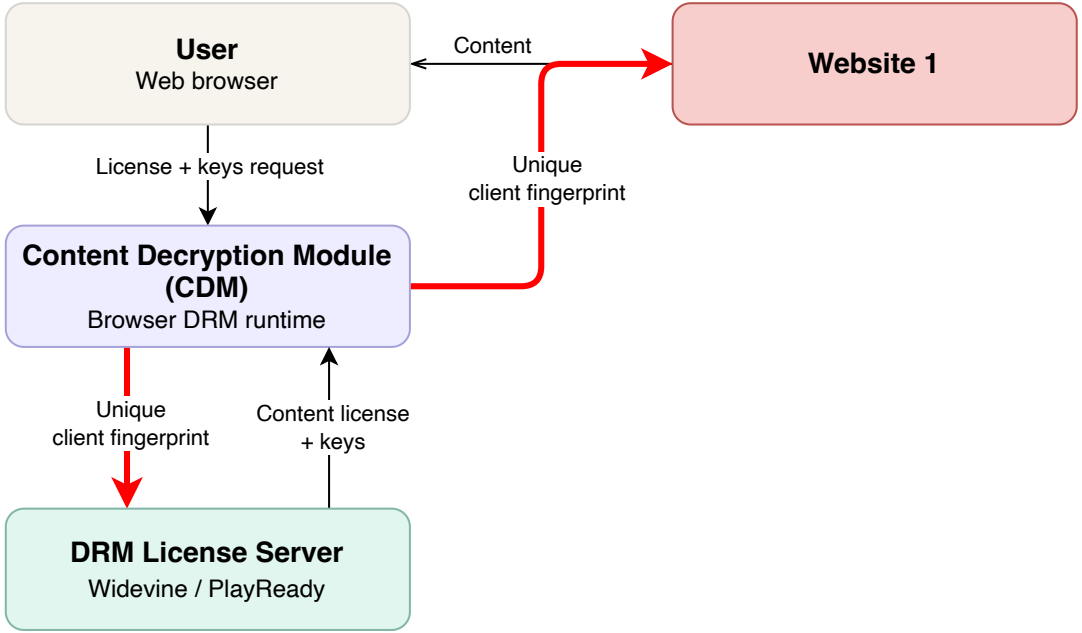
DRM privacy implications



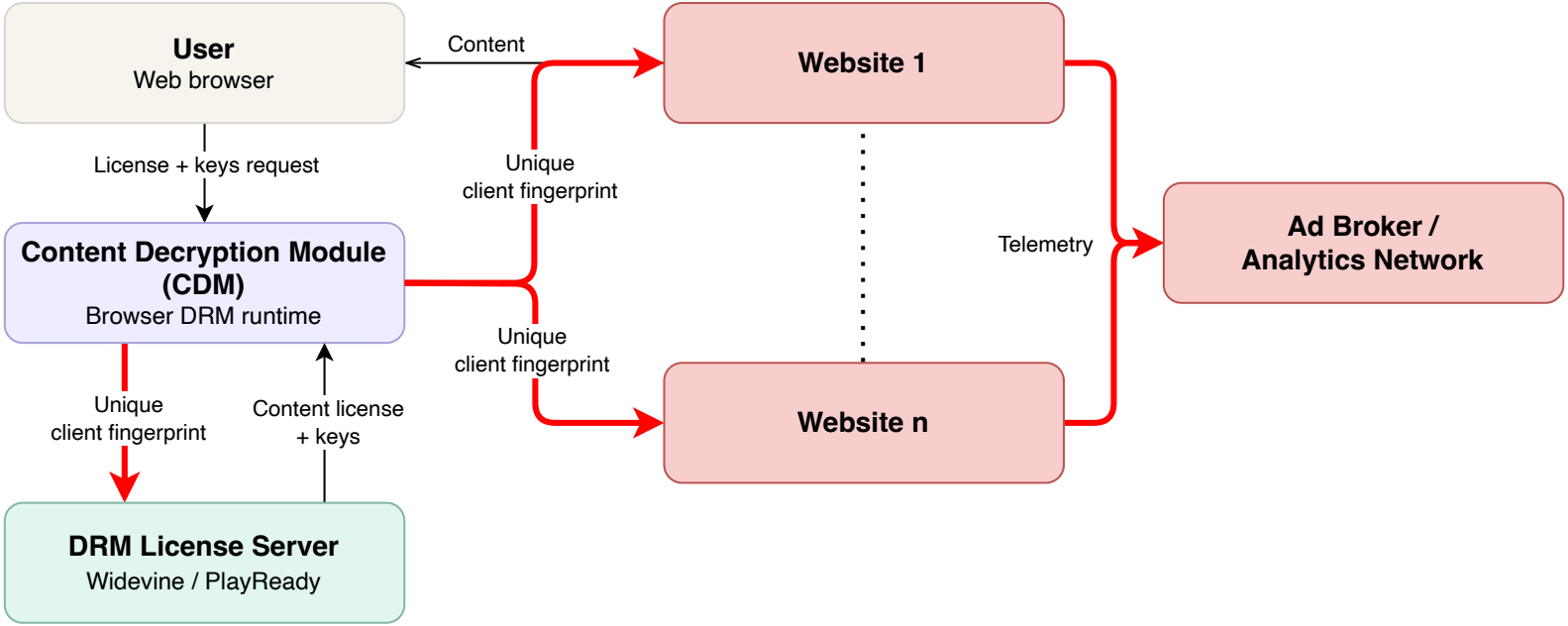
Client fingerprint shared with streaming service, which may be used to

- Watermark content
- Trace leaks back to devices involved in unauthorized content redistribution

DRM privacy implications



DRM privacy implications



DRM privacy implications

Your DRM Can Watch You Too: Exploring the Privacy Implications of Browsers (mis)Implementations of Widevine EME

Gwendal Patat
Univ Rennes, CNRS, IRISA
gwendal.patat@irisa.fr

Mohamed Sabt
Univ Rennes, CNRS, IRISA
mohamed.sabt@irisa.fr

Pierre-Alain Fouque
Univ Rennes, CNRS, IRISA
pierre-alain.fouque@irisa.fr

ABSTRACT

Thanks to HTML5, users can now view videos on Web browsers without installing plug-ins or relying on specific devices. In 2017, W3C published Encrypted Media Extensions (EME) as the first official Web standard for Digital Rights Management (DRM), with the overarching goal of allowing seamless integration of DRM systems on browsers. EME has prompted numerous voices of dissent with respect to the inadequate protection of users. Of particular interest, privacy concerns were articulated, especially that DRM systems inherently require uniquely identifying information on users' devices to control content distribution better. Despite this anecdotal evidence, we lack a comprehensive overview of how browsers have supported EME in practice and what privacy implications are caused by their implementations. In this paper, we fill this gap by investigating privacy leakage caused by EME relying on proprietary and closed-source DRM systems. We focus on Google Widevine because of its versatility and wide adoption. We conduct empirical experiments to show that browsers diverge when complying EME privacy guidelines, which might undermine users' privacy. For instance, we find that many browsers gladly give away the identifying Widevine Client ID with no or little explicit consent from users. Moreover, we characterize the privacy risks of users tracking when browsers miss applying EME guidelines regarding privacy. Because of being closed-source, our work involves reverse engineering to dissect the contents of EME messages as instantiated by Widevine. Finally, we implement EME Track, a tool that automatically exploits bad Widevine-based implementations to break privacy.

KEYWORDS

CDM) a DRM key system. Multiple actors are involved in providing DRM key systems, three of which are distinguished: Microsoft PlayReady [40], Google Widevine [24] and Apple FairPlay [3]. Naturally, different platforms support different DRM key systems. For instance, video streaming is protected by FairPlay on Apple devices (i.e., iOS devices, Apple TV, and Safari on macOS), while Edge on Windows can rely on PlayReady. The lack of cross-platform DRM compatibility was a major reason behind DRM being ultimately abandoned for iTunes Music in 2007.

DRM and the Web are no strangers since users often stream video on browsers. Historically, DRM on the Web was supported in plug-ins for a long time (e.g., Microsoft Silverlight and Adobe Flash). However, the advent of HTML5-based media playback systems encouraged the media industry to make DRM integration more seamless. In 2012, Google and Microsoft partnered with Netflix (a content provider) to propose a "built-in" DRM extension for the Web: the W3C Encrypted Media Extensions (EME) [11], with the overarching goal to define a standard DRM API (Application Programming Interface) that would work across multiple browsers or operating systems on a broad of range of devices. The EME specification does not create yet-another DRM key system. Instead, it allows browsers to discover, select and interact with any DRM module automatically. Thus, EME removes the burden of implementing content protection from browsers, whose role is now restricted to only redirecting DRM messages to the right native key system. In 2016, all major browser vendors demonstrated interoperable support of EME, one year before becoming a W3C Recommendation [60].

Standardizing EME received much controversy, as EME stands at the intersection between the desire of freedom of the Internet and the increasing need to protect premium contents and services

README GPL-3.0 license

Device Fingerprinting through EME Widevine

This project shows how the W3C EME API can be used to perform fingerprinting of devices using the Widevine DRM system. It collects Widevine Client ID when in clear, and open persistent sessions within the OS file system to provide both stateful and stateless user tracking to curious origins.

How does it work?

This PoC uses the EME API from compatible web browsers to communicate with the underlying Widevine DRM system. In its usual workflow, Widevine generates license key requests to the license server in order to get media content keys to play protected assets. Under the opaque protocol of Widevine, such messages can be filled with distinctive identifiers leading to potential user tracking issues raised by the EME recommendation. Such distinctive identifiers can range from build info, CPU architecture, Widevine version up to device unique certificate hash.

Our PoC uses a JavaScript file to request a license key response from the Widevine integration test server and redirects the actual request to a rogue server in charge of collecting fingerprints.

Full details can be found in our research paper.

Setup of the Proof-of-Concept

To set up a working environment, you need to provide the `index.html` and `script_eme_full.js` to a webpage through HTTPS. Replace the rogueUrl variable within the JS script to the url of the server in charge to collect the fingerprints.

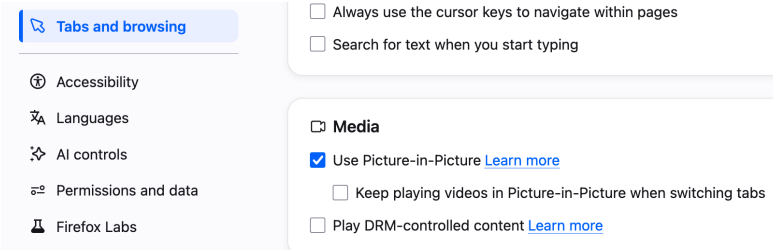
Docker Setup

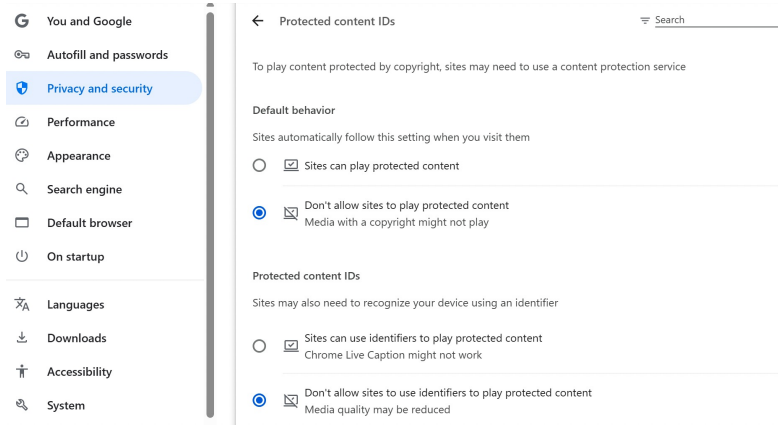
The `Docker` directory host a docker compose file to build a test environment on `localhost`. This docker setup an Apache server serving both `index.html` and `script_eme_full.js` files over HTTPs, allowing EME usage for both Firefox and Chrome-based browsers.

Patat, Gwendal; Sabt, Mohamed; Fouque, Pierre-Alain (2023). "Your DRM Can Watch You Too: Exploring the Privacy Implications of Browsers (mis)Implementations of Widevine EME". *Proceedings on Privacy Enhancing Technologies (PoPETs)*. 2023 (4): 306–321.

Browser-based mitigations

- Consider disabling DRM-protected content playback
- Software-based DRM: client ID stored in browser profile
  - Consider creating fresh profile on each streaming session
  - Note: container tabs not sufficient here
- Hardware-based DRM: client ID based on TEE PubKey fingerprint
  - Cannot rotate without swapping out GPU





Conclusion

- Hardware-backed DRM protects keys and decoding, but not the final peripheral handoff
- DReaMcatcher: class break attack that fully bypasses Google Widevine L1 and Microsoft PlayReady SL3000 audio DRM
- Vulnerability disclosure in September 2025 – vendor response pending

- DRM content protection and user security/privacy tradeoffs remain challenging

Thanks

Björn Ruytenberg

@0Xiphorus@infosec.exchange

https://bjornweb.nl

DReaMcatcher Paper

https://dl.acm.org/doi/10.1145/3767295.3803583

Additional slides

Video: E2E secure decryption pipeline