Trust, But Verify...

Exposing Design Risks in Your Application Catalog

adyen

engineered
for ambition

Our Fleet at 10,000 ft.

MacOS Presence among

97%

As of January 2025

CIS L1 and CIS L2 compliance regime for workstations and mobile devices.

30+

offices in EMEA, Americas, and APAC regions

5600

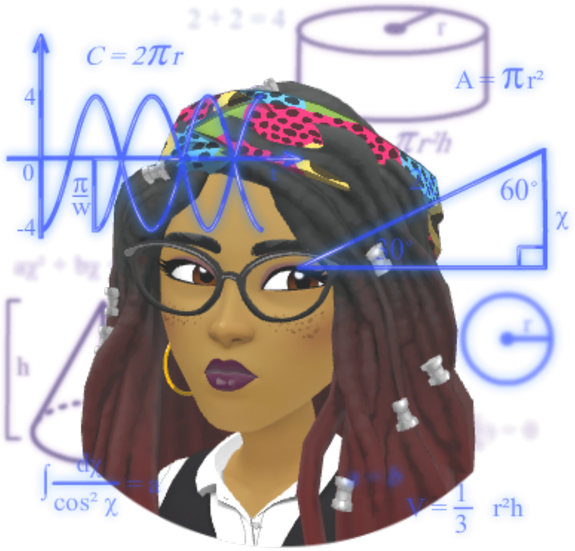
User endpoints

Hybrid infrastructure between SaaS-native and internally developed tools.

Local admin disabled for all internal users regardless of role.

The Big Question

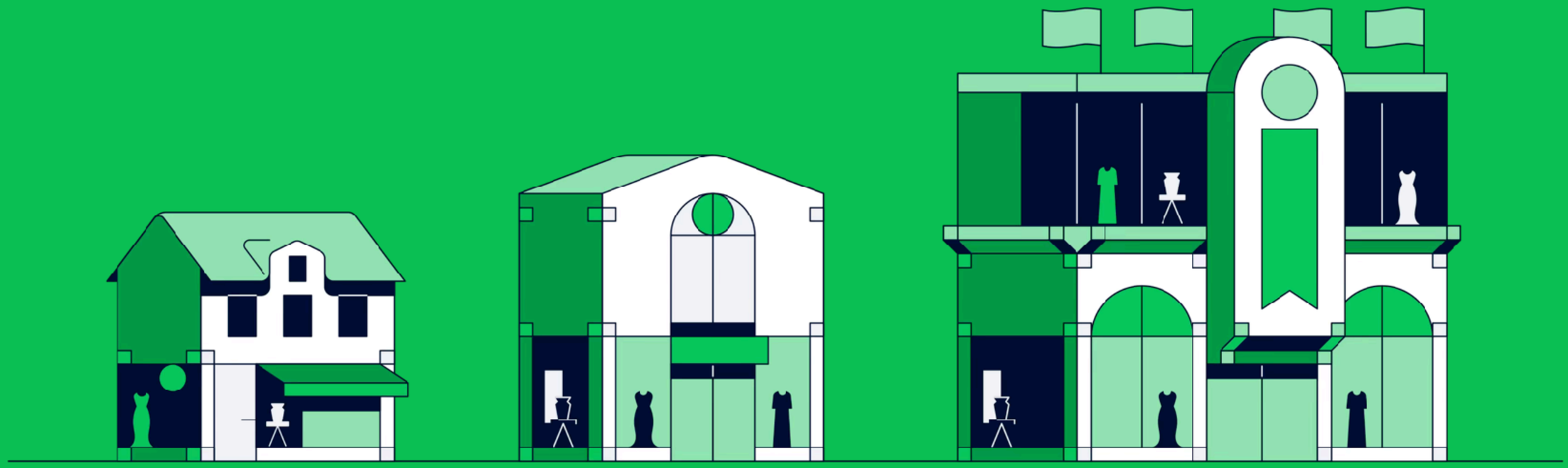
How much do we really know about what our tools can do?



For many organizations, the stark truth is tha we know shockingly little about how the tools our user-communities work with everyday, accomplish their stated goals.

Administrators and Security Engineers are hard pressed to strike the balance between giving the people what they need to succeed and ensuring proper evaluation of potential risks.

Our problem space expands with our catalog, as new apps enter and updates ship with ever larger, more sophisticated language models.




Gemini
OPENCLAW
GIVE IT TO ME NOW!!!

Denial By Analogy

Is the app or utility sufficiently similar to another that we're already using and approved in the past?

Cost Considerations

Does the new tool come with license//use costs we don't want to pay or advocate for?

Basic Checks

Is the app or tool signed in a way that Gatekeeper will accept at first run?

Security Escape Hatch

Sorry we just don't allow that via company or security policy.

We need an approach that allows us to move away from relying on strong intuitions and luck to mitigate risk . Instead we should look to structural choices developers have made to decide whether we're comfortable with production use.

adyen

Increased visibility leads to informed choices.

Legacy Methods:

Check for use of old and custom methods and features

Bloat & Packing:

Investigate for signs of bloating or packing.

Strong Linking:

Verify strong types and standard paths for libraries and run paths.

Safe Execution:

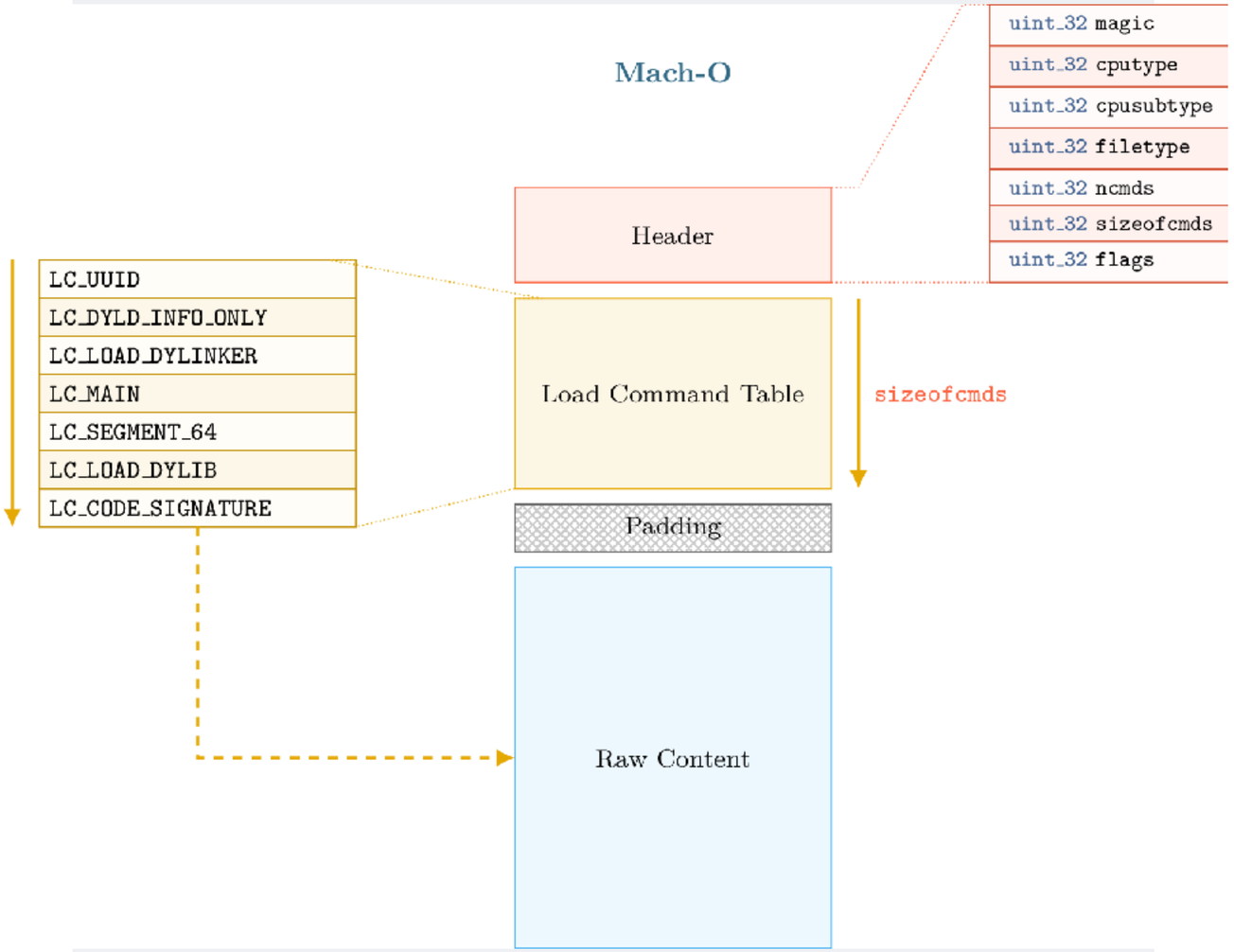
Ensure compliance with Apple-native runtime protections.

Getting Started

The Mach-O Format, Hardened Runtime, and App Entitlements

adyen

engineered
for ambition



One Format to Rule Them All

- Mach Binaries are typically composed of three parts: a header, a set of load commands, and a set of segments containing the executable code.
- Their code is read by the dynamic linker (normally dyld) in order from top to bottom.
- Feature flags in the header instruct dyld on how to handle the binary's code once loaded into memory.
- Load commands tell the linker which libraries to use and how to work with the symbols they contain.
- Then, finally, the binary's core content is loaded and run, contextualized by feature flags,

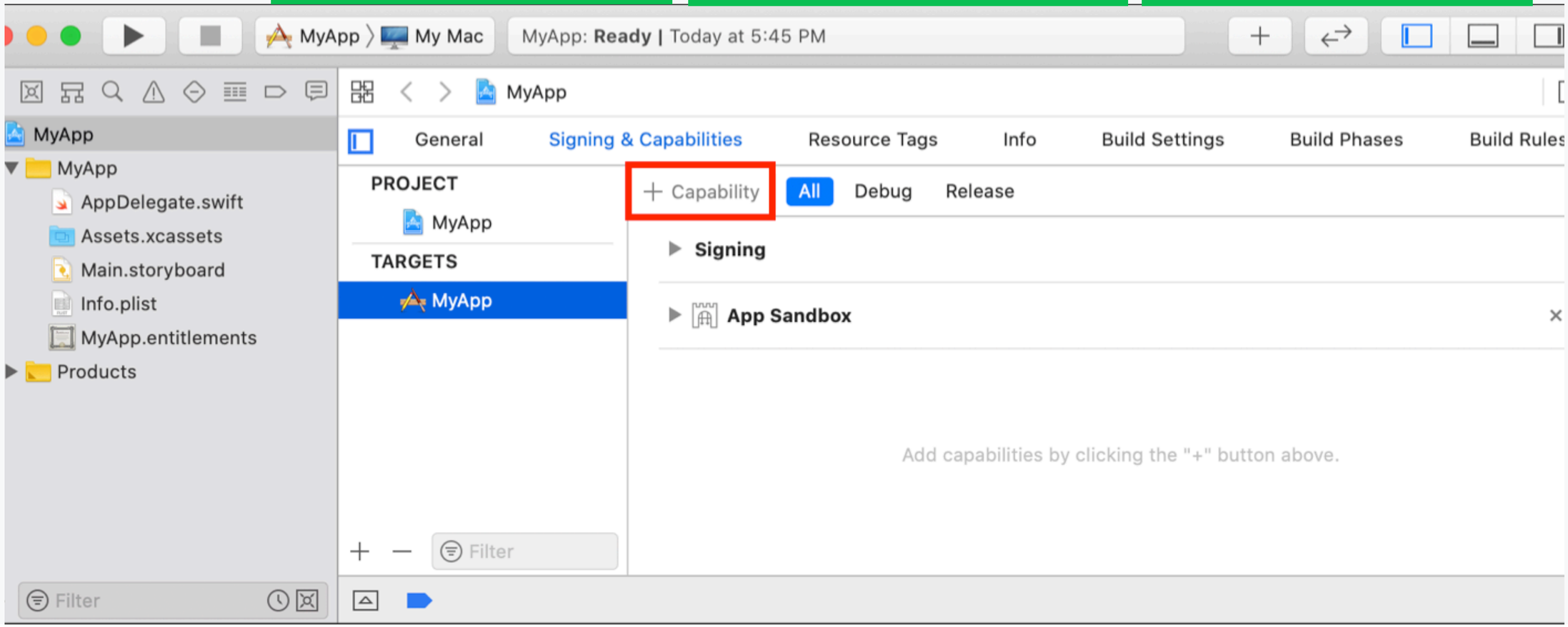
Runtime protections help keep our code safe

Code Injection

Dynamic Library Hijacking

Execution Space Tampering

Just-in-Time Compiling



Entitlements enable runtime exceptions as well as system permissions.

```
[Key] com.apple.security.assets.music.read-only
[Key] com.apple.security.assets.pictures.read-write
[Key] com.apple.security.automation.apple-events
[Key] com.apple.security.cs.allow-jit
[Key] com.apple.security.cs.disable-library-validation
[Key] com.apple.security.device.audio-input
[Key] com.apple.security.files.bookmarks.app-scope
[Key] com.apple.security.files.bookmarks.document-scope
[Key] com.apple.security.files.user-selected.read-write
[Key] com.apple.security.network.client
[Key] com.apple.security.personal-information.addressbook
[Key] com.apple.security.personal-information.photos-library
[Key] com.apple.security.print
[Key] com.apple.security.scripting-targets
[Key] com.apple.security.temporary-exception.apple-events
```

Diving Deeper

Examining development choices, to gauge risk.

Commands to check for Legacy & Custom Methods

- `/usr/bin/otool -hv "$target_binary_path" | /usr/bin/awk '/MH_MAGIC/ { for (i=9; i<=NF; i++) print $i }' | grep -E '(FORCEFLAT|ALLOW_STACK_EXECUTION|ROOT_SAFE|WEAK_DEFINES|BINDS_TO_WEAK)\b'`
- `/usr/bin/otool -fhlLDGvV "$target_binary_path" | /usr/bin/grep -i segname | /usr/bin/grep -v -E '\s+__(?:TEXT|DATA_CONST|DATA|LINKEDIT|PAGEZERO)\b'`
- `/usr/bin/otool -fhlLDGvV "$target_binary_path" | grep -E '\s+(?:LC_DYLD_ENVIRONMENT|LC_UNIXTHREAD|DYLD_INSERT_LIBRARIES)\b’`

Don't Worry, We Have You Covered.

MH_NOUNDEFS

A marker of completeness. No undefined references.

MH_TWOLEVEL

Function collision protection enabled.

MH_PIE

Indicates use of relative addresses over absolute ones. Enables other security features.

MH_APP_EXTENSION_SAFE

Indicates binary is safe to use as an app extension.

MH_INCRLINK

File was produced by an incremental link.

MH_BINDATLOAD

Overrides loading functions on demand. Forces loading all at start.

MH_NO_HEAP_EXECUTION

Sets heap memory to "non-executable".

MH_DYLDLINK

Indicates use of 'dyld' and .dylib files.

MH_WEAK_DEFINES

Indicates program contains symbols that can be overwritten.

MH_SPLIT_SEGS

Enables read-only data to be shared across multiple instances of the application running.

MH_SUBSECTIONS_VIA_SYMBOLS

"Dead Code Stripping" is enabled.

MH_BINDS_TO_WEAK

Indicates binary can load symbols with 'NULL' values.

MH_HAS_TLV_DESCRIPTORS

Indicates support for Thread Local Storage

MH_SIM_SUPPORT

Means the binary was intended for use with iOS or watchOS simulators.

Commands to validate Load Command Paths

- `/usr/bin/otool -l "$target_binary_path" | /usr/bin/awk '$1 == "cmd" && $2 == "LC_LOAD_WEAK_DYLIB" { found = 1 } $1 == "name" && found { print $2; found = 0 }'`
- `/usr/bin/otool -l "$target_binary_path" | /usr/bin/awk '$1 == "cmd" && $2 == "LC_LOAD_DYLIB" { found = 1 } $1 == "name" && found { print $2; found = 0 }'`
- `/usr/bin/otool -l "$target_binary_path" | /usr/bin/awk '$1 == "cmd" && $2 == "LC_RPATH" { found = 1 } $1 == "name" { print $2; found == 0 }'`
- `/usr/bin/otool -l "$target_binary_path" | /usr/bin/awk '$1 == "cmd" && $2 == "LC_LOAD_DYLINKER" { found = 1 } $1 == "name" && found { print $2; exit }'`

Live Demo: Inspecting Load Command Contents

adyen

engineered
for ambition

Using ‘dd’ to unpack Large Load Commands

- **Set Input File**

```
INFILE=“<path to binary>”
```

- **Set Output `.blob` File**

```
OUTFILE=“<chosen path to .blob file>”
```

- **Invoke `‘dd’` to Parse Bytes**

```
dd if="$INFILE” of=“$OUTFILE” bs=1 skip=[file offset value]
count=[file size value]
```

- **Redirect via `‘strings’` to Export Text**

```
strings <path to .blob file> > <chosen output file path>
```

Live Demo: Runtime Bitmasks & Entitlements

adyen

engineered
for ambition

Take a closer look at Code Signature Blocks

- Examine Code Signatures via ‘codesign’

```
/usr/bin/codesign -dvvv <path to binary or .app>
```

- Check for Hardened Runtime compliance via flags

```
CodeDirectory v=20500 size=38787 flags=0x10000(runtime)
hashes=1201+7 location=embedded Hash type=sha256 size=32
```

- Inspect Entitlements for Runtime Exceptions

```
/usr/bin/codesign -d —entitlements - <path to binary
or .app>
```

Before you ask, yes we have outlined the hardened runtime flags too.

CS_VALID (0x00000001)

Code signature is valid.

CS_ADHOC (0x00000002)

Binary does not have a trusted signer.

CS_GET_TASK_ALLOW(0x00000004)

Binary holds the get-task-allow entitlement (*legacy*).

CS_INSTALLER (0x00000008)

Binary holds the installer entitlement (*legacy*).

CS_INVALID_ALLOWED (0x00000020)

Allows overriding standard XNU runtime monitoring to force-terminate changes to memory.

CS_KILL (0x00000100)

Instruction to kill the process if invalid.

CS_HARD (0x00000200)

Sets strict enforcement against loading bad memory pages.

CS_CHECK_EXPIRATION (0x00000400)

Sets validation of certificate expiration at runtime.

CS_RESTRICT (0x00000800)

Ignore custom dyld environment variables.

CS_REQUIRE_LV (0x00002000)

Requires all libraries carry valid code signatures

CS_RUNTIME (0x000010000)

A marker of Hardened Runtime being enabled.

CS_LINKER_SIGNED (0x00020000)

A marker that the binary was signed by the linker daemon.

Conclusion

Wrapping Up

So...Any Questions?

From all of us at Adyen. Thank you!

adyen

engineered
for ambition