RESOURCES • BLOG

THREAT DETECTION

Going off script: Thwarting OSA,

abuse

Experts from Red Canary, Jamf, and MITRE ATT&CK opine on ways to detect and prevent manipulation of macOS's scripting architecture.

SYDNEY GELB

Originally published November 1, 2022. Last modified April 30, 2024.

Living off the land has been commonplace on Windows systems for years, so it's no surprise that adversaries frequently leverage native tooling when they seek to compromise macOS systems. For the long-awaited return of our Detection Series **webinars**, Red Canary's **Tony Lambert** and **Brandon Dalton** joined **Cat Self** from MITRE and **Ferdous ("Sal") Saljooki** from Jamf to explain why adversaries exploit Apple's native scripting capabilities, and how to ward them off.

So, what are these native capabilities?

Scripting languages on macOS are beholden to a structure known as Apple's Open Scripting Architecture (OSA). According to **Apple**:

> "The Open Scripting Architecture (OSA) provides a standard and extensible mechanism for interapplication communication in OSX."

Here, Cat offers a clarifying explanation of OSA and its

SentinelOne offers an insightful deep-dive on OSA for further learning.

Cat continues on to explains the benefits of leveraging OSA:

The two primary scripting languages under the OSA structure include **AppleScript** and **JavaScript for Automation** (JXA). So, let's get into how to dodge, duck, dip, dive, and dodge the adversaries who abuse these scripting languages.

Who's taking advantage?

What should I be looking for?

XCCSET, a malware threat that targets developers, is distributed as poisoned XCode project files.

Distributed as read-only, compiled AppleScript, OSAMiner is a multi-stage threat that retrieves a Monero miner and installs it on a macOS system.

Often used by Red Team operators, the Apfell Agent is a JXA agent created to talk to Mythic C2.

Brandon illustrates the purpose and facilitation of Apple's Endpoint Security Framework (ESF) for monitoring system events.

Sal walks us through ways to advance detection coverage by leveraging available telemetry.

Can I emulate these behaviors to test detection coverage?

Absolutely! Thus far, the panelists have discussed how and why adversaries abuse AppleScript and JXA, where defenders can find telemetry to observe suspicious activity, and how you can leverage that telemetry to develop or improve detection coverage.

Using our newly released **POSIX AtomicTestHarness** suite you can quickly test for detection coverage gaps. **AtomicTestHarnesses** focus on the art of the possible. If an adversary were to leverage AppleScript / JXA to attack macOS, what different ways could they go about doing that? AtomicTestHarnesses help answer this question.

Brandon discusses how to test your visibility into suspect AppleScript and JXA activity in your environment.

Speaking of the POSIX AtomicTestHarness suite, Red Canary's Brandon Dalton and Dave Bogle wrote a blog delving into how the POSIX Atomic Test Harnesses suite leverages Python to emulate multiple variations of a given ATT&CK technique on **Linux** and **macOS systems.** Read it **here!**

KEEP WATCHING

RELATED ARTICLES

THREAT DETECTION

Artificial authentication: Understanding and observing Azure OpenAI abuse

THREAT DETECTION

Apple picking: Bobbing for Atomic Stealer & other macOS malware

THREAT DETECTION

Keep track of AWS user activity with SourceIdentity attribute

Trending cyberthreats and techniques from the first half of 2024

Subscribe to our blog

First Nam

Last Name

Email Address

SUBSCRIBE >

You'll receive a weekly email with our new blog posts.