JNUC 2024 Nashville

Jamf API CLI Tool:
One Solution for Secure Jamf API Access in Policies and Scripts

Howard Griffith

University of Texas at Austin

Senior Systems Manager

Liberal Arts Instructional Technology Services

MacAdmins Slack: @HowardGMac

JNUC 2024

My Tech Origin Story
(highlight reel edition)…

United States Navy

Microsoft Technical Support

Eastern Washington University

University of Texas at Austin

Jamf API CLI Tool

The **apitokentool** is a command line utility that, when called, provides an API Bearer Token Key and its expiration information without exposing the sensitive credentials on the device it is called from.

Agenda

- How the apitokentool works and why use it
- Overview of the security model
- Setting up and building it
- How it can simplify our scripts
- Handling potential misuse

- Questions and Answers

How the apitokentool works…

When invoked the tool sends the signing information of the tool and serial number of the device to an initializing web hook.

If the sent values are verified, the initializing web hook returns an address to the authorization web hook.

The tool then sends a client token key from the device to the authorization web hook.

If the client token key is verified, the authorization web hook returns the appropriate API bearer token and its expiration information.

How the apitokentool works…

If either web hook call rejects any of the values presented, it returns a 401 Unauthorized error message.

The tool caller can then use the bearer token information to make appropriate API calls to the Jamf server based on the permission level of the token key.

The tool can be used in a script to refresh the bearer token information if the expiration period of the initial bearer token is reached.

…Why use It?

The apitokentool negates the need for embedding the obfuscated API credential information directly in a script or sent as a script parameter in the Jamf policy.

When setup properly, the apitokentool can provide different levels of API access based on the client token key used and without exposing any underlying API credential information on the calling device.

Overview of the security model…

The tool must be signed with an organizationally recognized signing certificate.

The tool signature must indicate that is has not been altered.

The tool must be run from a device that is enrolled in your Jamf instance.

Security Layer 1

The tool must present a valid client token key.

The authorization web hook will authenticate the client token key.

The authenticated client token key determines API access permissions.

Security Layer 2

Overview of the security model…

A simple UUID contained in a configuration profile.

```
#!/usr/local/bin/managed_python3
import uuid

client_token_key = uuid.uuid1()
```

Client Token Key

Overview of the security model…

The client token key + known modifier

```
#!/usr/local/bin/managed_python3
import uuid

client_token_key = uuid.uuid1()

authenticated_token_key = \
uuid.uuid5(uuid.NAMESPACE_DNS, 'site.mdm.server/' + \
str(client_token_key))
```

Authenticated Client Token Key

Overview of the security model…

2baf8294-63c3-11ef-a148-3a05fe0eb375

43a88e04-63c3-11ef-893d-3a05fe0eb375

60df8f5e-63c3-11ef-b887-3a05fe0eb375

Client Token Key

f723fc8f-43bb-5bc6-8ad6-52feee4266b2

865abb4e-acde-5209-85d1-1783c35c0649

56eced66-859d-589e-ad5e-34df4dce4811

Authenticated Client Token Key

Computer - Read Only

Computer - Read & Update

Computer/Device - Read, Update, and Delete

Jamf API Permissions

JNUC 2024

Setting up and building it…

- Configuration profile identifier for the client token key.
- How many API levels of access are needed.
- Determine which Apple Developer signing certificate is to be used.
- Hosting location for apitokentool web hook responders.

Preliminary Decisions

- Generate and document client token keys.
- Upload and configure apitokentool web hook responders.
- Document web hook responder URLs.

Preliminary apitokentool Steps

Setting up and building it…

- Create configuration profiles for each client token key.
- Setup appropriate API user accounts with proper permissions.
- Create necessary transition scripts and policies for existing scripts and policies that make Jamf API calls.

Preliminary Jamf server Steps

Setting up and building it…

- Set initial web hook responder URL and configuration profile identifier in apitokentool binary source.
- Build, sign and package the apitokentool binary.
- Set signing information, Jamf server URL, Jamf API read-only access credentials, and authorization web hook URL in the initial web hook responder.
- Set the authenticated token key values, Jamf server URL and corresponding Jamf API credentials in the authorization web hook responder.

Intermediate apitokentool Steps

Setting up and building it…

- Deploy apitokentool package to appropriate test devices.
- Assign client token key configuration profiles to appropriate test devices.
- Test all the new scripts and policies that use the apitokentool for Jamf API calls.

Intermediate Jamf server Steps

Setting up and building it…

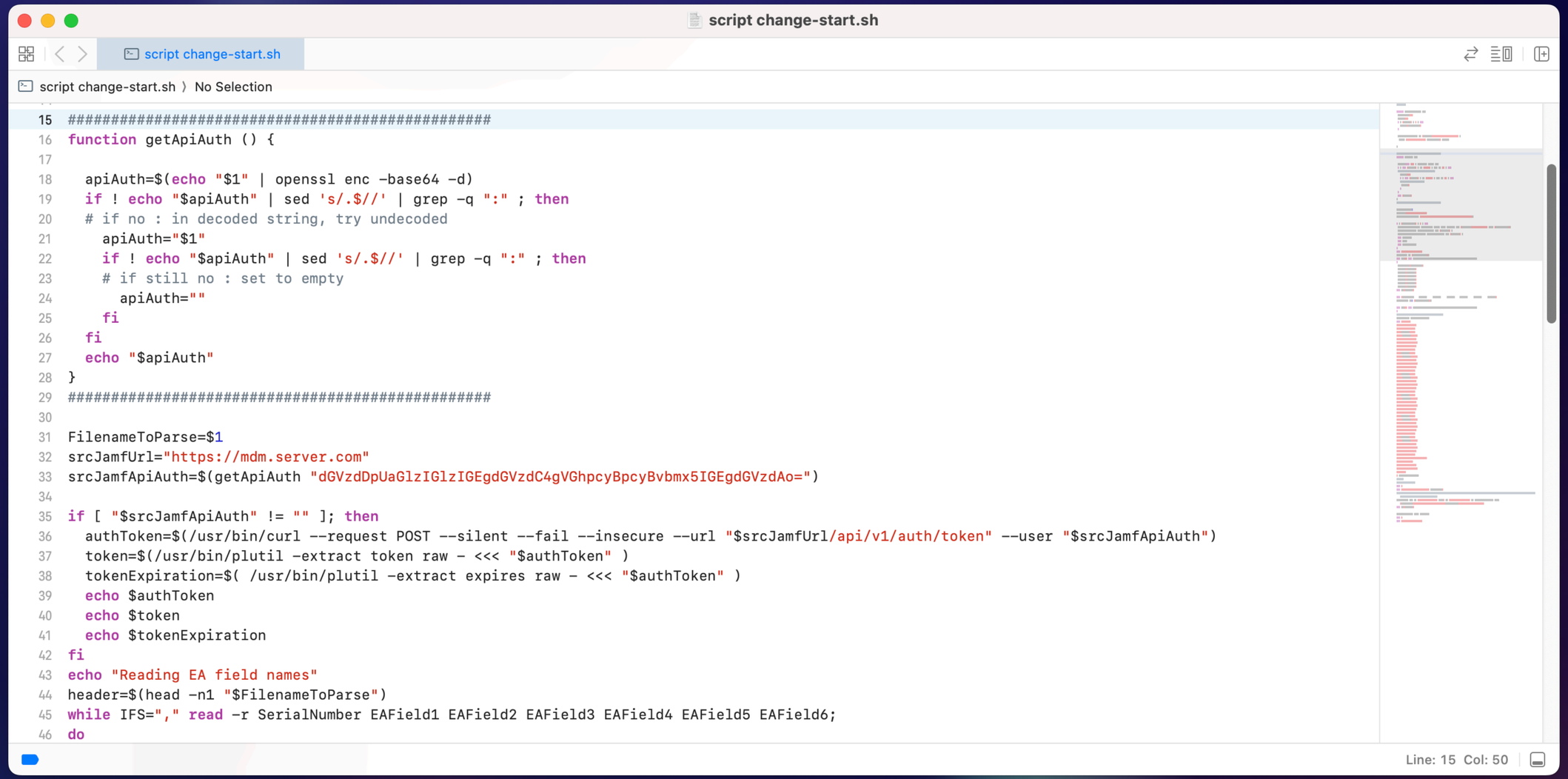
- Open Terminal on a test device and use the apitokentool binary.
- Verify functionality of presented API bearer token.

Final apitokentool Steps

- Roll out changes to scripts and policies that use the apitokentool for Jamf API calls.
- Deploy apitokentool package to all appropriate devices.
- Assign client token key configuration profiles to all appropriate devices.

Final Jamf server Steps

How it can simplify our scripts...



JNUC 2024

How it can simplify our scripts…

JNUC 2024

Handling potential misuse…

Single Device compromise

Unscope the device from the client key configuration profile.

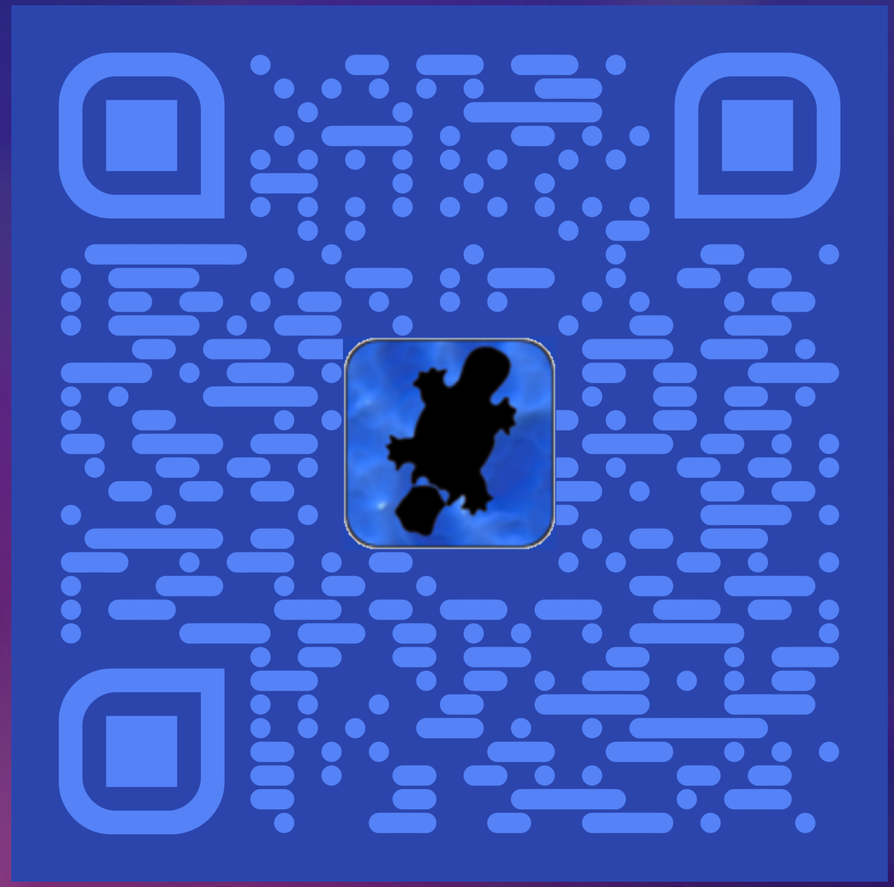
Single API Permission Level compromise

Change the client token key(s) on the authorization web hook responder.

Multiple API Permission Level compromise

Change the authorization web hook URL on the initial web hook responder.

Thank you everyone !



https://github.com/HowardGMac/apitokentool

JNUC 2024