# Get-AzPasswords: Encrypting Automation Password Data

Get-AzPasswords is a function within the MicroBurst toolkit that's used to get passwords from Azure subscriptions using the Az PowerShell modules. As part of this, the function supports gathering passwords and certificates that are attached to automation accounts.

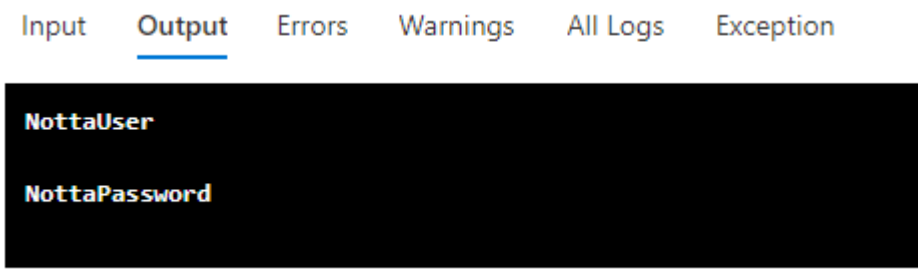
These credentials can be stored in a few different ways:

- Credentials – Username/Password combinations
- Connections – Service Principal accounts that you can assume the identity of
- Certificates – Certs that can be used in the runbooks

If you have the ability to write and run runbooks in an automation account, each of these credentials can be retrieved in a usable format (cleartext or files). All of the stored automation account credentials require runbooks to retrieve, and we really only have one easy option to return the credentials to Get-AzPasswords: print the credentials to the runbook output as part of the extraction script.

## The Problem

The primary issue with writing these credentials to the runbook job output is the availability of those credentials to anyone that has rights to read the job output.




By exporting credentials through the job output, an attacker could unintentionally expose automation account credentials to lesser privileged users, resulting in an opportunity for privilege escalation. As responsible pen testers, we don't want to leave an environment more vulnerable than it was when we started testing, so outputting the credentials to the output logs in cleartext is not acceptable.

## The Solution

To work around this issue, we've implemented a certificate-based encryption scheme in the Get-AzPasswords function to encrypt any credentials in the log output.

The automation account portion of the Get-AzPasswords function now uses the following process:

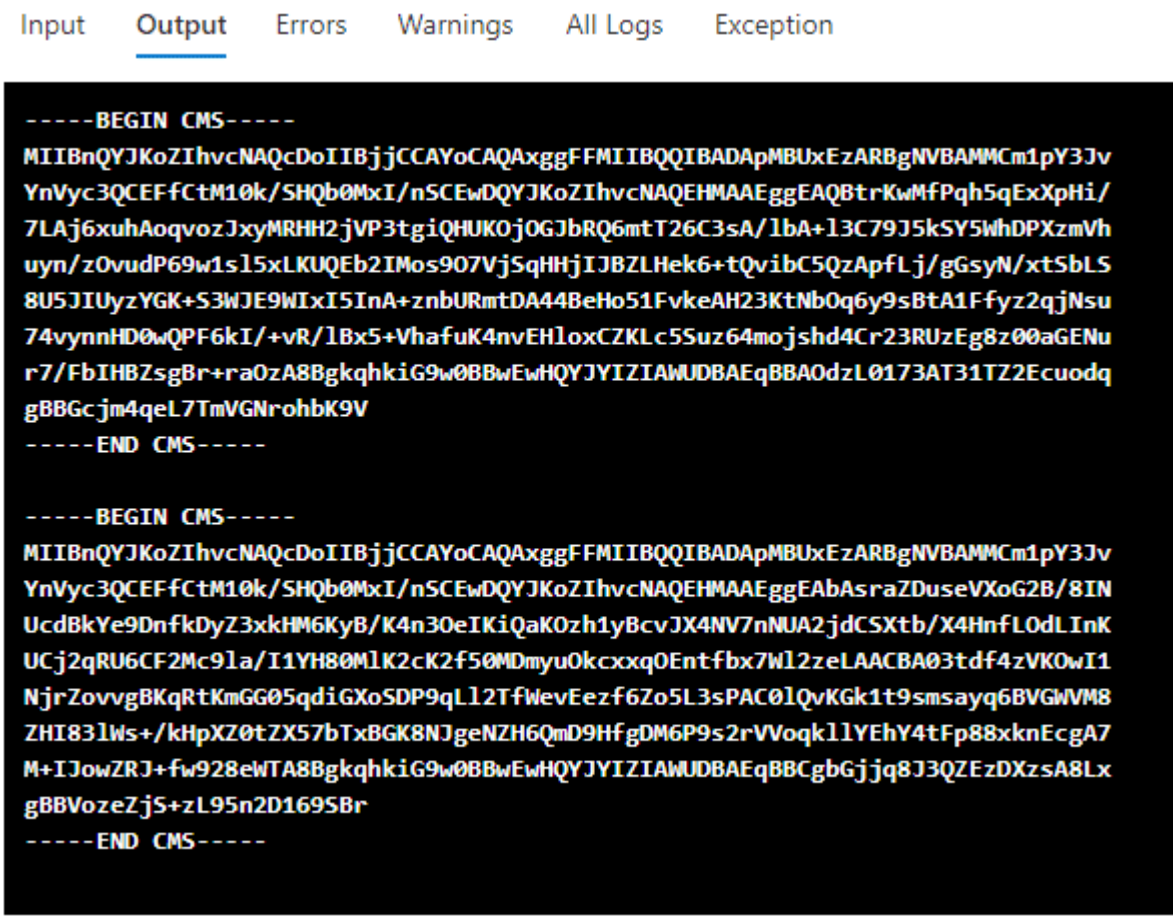
1. Create a new self-signed certificate (microburst) on the system that is running the function
2. Export the public certificate to a local file

3. Encode the certificate file into a base64 string to use in the automation runbook
4. Decode the base64 bytes to a cer file and import the certificate in the automation account
5. Use the certificate to encrypt (Protect-CmsMessage) the credential data before it's written to the output
6. Decrypt (Unprotect-CmsMessage) the output when it's retrieved on the testing system
7. Remove the self-signed cert and remove the local file from the testing system

This process protects the credentials in the logs and in transit. Since each certificate is generated at runtime, there's less concern of someone decrypting the credential data from the logs after the fact.

## The Results

Those same credentials from above will now look like this in the output logs:




On the Get-AzPasswords side of this, you won't see any difference from previous versions. Any credentials gathered from automation accounts will still be available in cleartext in the script output, but now the credentials will be protected in the output logs.

For anyone making use of Get-Azpasswords in MicroBurst, make sure that you grab the latest version from NetSPI's GitHub – https://github.com/NetSPI/MicroBurst