EFT Secrets: A Primer

THE INFORMATION IN THIS ARTICLE APPLIES TO:

- EFT v8.0.1 and later

DISCUSSION

EFT Secrets Storage & Security

This article describes the various mechanisms used by EFT to protects its secrets. The word "secrets" in this context includes passwords, passphrases, keys, etc. used by EFT or those used for authenticating users.

Types of Secrets

There are two distinct categories of secrets used by EFT:

1. **Non-Reversible Secrets.** These are secrets associated with a user or administrator accounts and used for authentication during interactive sessions. These secrets are not stored or known to EFT once they are created. Only a one-way salted hash of the secret is stored in accordance with high-security guidelines for protecting passwords. When a user or administrator authenticates, EFT hashes the user-supplied password, compares the computed hash to the stored hash, and authenticates the account if there is a match. The password supplied by the user at authentication is destroyed in memory after having been hashed, except when authentication pass-through is enabled in settings. (Non-reversible secrets are not the primary subject of this article and are covered elsewhere in great detail.)

1. **Reversible Secrets.** These secrets are created by an administrator or, in some cases, by EFT itself for various unattended operations, and thus must be stored somewhere so they can be retrieved by EFT when needed. An example of this type of secret is the passphrase used by EFT to decrypt the SSL certificate private key. This article focuses on how those secrets are protected, since it not possible to convert those secrets into a one-way hash, and thus pose an increased security risk.

Taxonomy

EFT's reversible secrets are split between Server and Site secrets.

EFT Secrets: A Primer

Server secrets include:

1. Remote administration SSL cert passphrase
2. ARM database connection string
3. SMTP authentication password

Site secrets include:

1. The Site's SSH private key passphrase
2. The Site's SSL cert passphrase
3. The Site's PGP secret key passphrase
4. Any Connection Profile passphrases
5. Any AS2 profiles password
6. Any passwords used in Event Rules for outbound connections
7. Any service credential override passphrases
8. Tokens or access keys used by EFT to authenticate against remote REST services such as AWS or Azure
9. The Site's authentication provider secrets (LDAP account passphrase, ODBC connection string, etc.)
10. Any Socks/Proxy passwords
11. The Encrypted folder passphrase (post v8)
12. The Encrypted personal data passphrase (post v8)

Local or Remote Vault

Prior to version 8.0, all of EFT's reversible secrets were encrypted using EFT's master secret (see Master Secrets) and stored in EFT's FTP.cfg file, a binary config file that was not human readable. In version 8.0, EFT switched from using a flat file to local database file for storing configuration and secrets.

Version 8.0.1 introduced the ability to optionally specify a remote location for storing EFT's reversible secrets (only for Site secrets, not Sever secrets). Azure Key Vault (AKV) was the first remote location supported, but others are forthcoming. When a remote key vault is chosen for secrets storage, EFT stores only a reference GUID locally in its configuration database, and then creates a corresponding object in the remote vault with a name equal to the GUID and a value consisting of the non-encrypted secret. In Azure's nomenclature, the entire name:value pair is referred to as a "Secret."

EFT Secrets: A Primer

Azure's AKV service provides two distinct methods for authenticating an application (EFT in this case) or service against the vault. The first is via an authentication key or token. The downside of this approach is that the authentication secret must be stored locally in EFT (encrypted with EFT's master secret), which introduces a narrow but high-risk attack vector. Risk can be mitigated somewhat via frequent key rotation, and by limiting ACLs for physical access to the system hosting EFT. A superior approach is to use AKV's "managed identities" feature, which is only possible if the system hosting EFT is running on an Azure VM. When configured in this manner, the VM itself is pre-authorized against the key vault, which is an impersonation technique similar to how IWA works for password-less authentication of users in Windows.

Below is a benefits/drawbacks matrix comparing local to remote secrets storage:

Local

Benefits:

- No dependency on network layer
- Faster performance, more resilient
- No need to configure & maintain key vault
- Less components that can break

Drawbacks:

- Secrets remain decentralized
- Broader attack vector due to decentralization
- A compromised host could result in secret retrieval, if secrets can be decrypted*
- No versioning of secrets
- No other vault like features

Remote

Benefits:

- Centralization of secrets used by EFT or other apps or services
- Narrow attack vector means lower risk (but high impact)
- A compromised EFT won't yield its set of encrypted secrets*
- Provides additional benefits such as versioning

Drawbacks:

- Depends on network for communicating with vault
- Slower, error prone if connection broken (can result in EFT service outage, fatal errors)
- Need to setup and maintain key vault service
- Introduces a single point of failure
- Centralization of secrets is high risk impact if compromised

.telerik-reTable-4 { border-collapse: collapse; border: solid 0px; font-family: Tahoma; }
.telerik-reTable-4 tr.telerik-reTableHeaderRow-4 { border-width: 1.0pt 1.0pt 3.0pt 1.0pt;

margin-top: 0in; margin-right: 0in; margin-bottom: 10.0pt; margin-left: 0in; line-height: 115%; font-size: 11.0pt; font-family: "Calibri" , "sans-serif"; width: 119.7pt; background: #4F81BD; padding: 0in 5.4pt 0in 5.4pt; color: #FFFFFF; } .telerik-reTable-4 td.telerik-reTableHeaderFirstCol-4 { padding: 0in 5.4pt 0in 5.4pt; } .telerik-reTable-4 td.telerik-reTableHeaderLastCol-4 { padding: 0in 5.4pt 0in 5.4pt; } .telerik-reTable-4 td.telerik-reTableHeaderOddCol-4 { padding: 0in 5.4pt 0in 5.4pt; } .telerik-reTable-4 td.telerik-reTableHeaderEvenCol-4 { padding: 0in 5.4pt 0in 5.4pt; } .telerik-reTable-4 tr.telerik-reTableOddRow-4 { border-width: 1pt; color: #666666; vertical-align: top; border-bottom-style: solid; border-bottom-color: #4F81BD; } .telerik-reTable-4 tr.telerik-reTableEvenRow-4 { color: #666666; vertical-align: top; } .telerik-reTable-4 td.telerik-reTableFirstCol-4 { border-width: 1pt; border-color: #4F81BD; padding: 0in 5.4pt 0in 5.4pt; border-bottom-style: solid; border-left-style: solid; } .telerik-reTable-4 td.telerik-reTableLastCol-4 { border-width: 1pt; border-color: #4F81BD; border-bottom-style: solid; border-right-style: solid; padding: 0in 5.4pt 0in 5.4pt; } .telerik-reTable-4 td.telerik-reTableOddCol-4 { border-width: 1pt; border-color: #4F81BD; padding: 0in 5.4pt 0in 5.4pt; border-bottom-style: solid; } .telerik-reTable-4 td.telerik-reTableEvenCol-4 { border-width: 1pt; border-color: #4F81BD; padding: 0in 5.4pt 0in 5.4pt; border-bottom-style: solid; } .telerik-reTable-4 tr.telerik-reTableFooterRow-4 { color: #355C8C; background-color: #FFFFFF; vertical-align: top; padding: 0in 5.4pt 0in 5.4pt; } .telerik-reTable-4 td.telerik-reTableFooterFirstCol-4 { border-width: 1pt; border-color: #4F81BD; border-bottom-style: solid; border-left-style: solid; padding: 0in 5.4pt 0in 5.4pt; } .telerik-reTable-4 td.telerik-reTableFooterLastCol-4 { border-width: 1pt; border-color: #4F81BD; border-bottom-style: solid; border-right-style: solid; padding: 0in 5.4pt 0in 5.4pt; } .telerik-reTable-4 td.telerik-reTableFooterOddCol-4 { border-width: 1pt; border-color: #4F81BD; border-bottom-style: solid; padding: 0in 5.4pt 0in 5.4pt; } .telerik-reTable-4 td.telerik-reTableFooterEvenCol-4 { border-width: 1pt; border-color: #4F81BD; border-bottom-style: solid; padding: 0in 5.4pt 0in 5.4pt; } *Read below for more on how EFT protects local secrets. If managed identities are not used then further risk is introduced as the AKV authentication token must be stored locally in EFT and introduces a narrow but high impact risk attack vector.

Remote Vault FAQ

Usage of a remote vault, while beneficial in some regards, especially when centralized key management is a company policy, does introduce a few challenges and behaviors that might not be present or noticeable when using local storage. Below is a short set of questions and answers that help describe these behavioral differences.

Q. How and when does EFT interact with the vault under normal operations?

EFT Secrets: A Primer

A. At service startup, for Sites that point to a remote vault, when EFT is loading Site configuration it will connect to the vault, and retrieve all* secrets from the vault (consisting of GUIDs as the name, and the secret as the value). EFT will then encrypt the retrieved secrets in memory, and cache them using the cache-aside pattern. When a secret is need by EFT, a copy of the encrypted secret is decrypted and used, then immediately destroyed after use. If an administrator makes a change to EFT's configuration settings, EFT will evaluate whether one or more secrets have changed and only if a change has occurred, will EFT create a new GUID and corresponding secret in the vault. This design minimizes interaction with the vault to service startup and secret updates, thus mitigating any side-effects of intermittent connectivity failures or temporary vault outages.

Q. What happens if EFT cannot connect to the vault upon service startup?

A. If it cannot connect on service startup, the results will be largely catastrophic as EFT won't be able to load the configuration it needs. As such, for "Tier 1" or mission critical environments we recommend that EFT only be deployed within the Azure environment where it will have a higher chance of maintaining connectivity to the vault (as the VMs and vault are all in the same network environment).

Q. What happens if EFT cannot connect to the vault after the initial, successful retrieval of secrets?

A. If EFT has successfully connected at service startup, but cannot subsequently connect to the vault (to apply a change for example), EFT will go into retry mode, attempting to update the secret in the vault every few seconds for an indefinite period or until interrupted, at which point an error will be logged. If connectivity is restored between retries, then EFT will upon next attempt update the secret in the vault.

Q. When are secrets that are stored in the key vault updated?

A. Secrets are never updated. When an administrator changes EFT's configuration, EFT will create a new GUID and corresponding secret in the key vault. This is done to support EFT's backup/restore configuration. If you restore a from backup, then you would want EFT to be able to retrieve the secret(s) associated with a prior set of GUIDs.

Q. What if a secret is updated directly in the key vault?

A. EFT will be unaware as it will rely on the cached value retrieved at service startup. To retrieve values updated in the key vault directly, you would need to restart EFT's service, which forces EFT to flush its cache and re-retrieve the list of secrets.

Q. What if I update a secret directly in the key vault, and before restarting EFT, I also update that same secret in EFT's admin GUI?

A. The new value specified in EFT's GUI will result in a new GUID and secret being created in the vault. As such, the GUID you modified in the vault will not be used.

Q. What if I delete one of the secrets in the key vault?

A. This could cause a fatal to occur in EFT depending on the secret that was deleted. Some secrets are operationally critical and can even result in the Site not starting. You should never delete values in the vault, or at least make a backup before you do.

Q. The frequent change to EFT's secrets has resulted in a large number of entries in my vault. How do I clean up legacy and orphaned entries?

A. Navigate to the EFT's secrets storage configuration page and change it to EFT managed secrets rather than vault managed. This will effectively copy all current secrets from the vault to EFT's configuration file (encrypted). Now you can delete all of entries in the remote vault. Next, swap back using the remote vault and only the most recent entries will be posted there. Do a backup and restore at each step, but be aware that prior backups will be obsoleted as any referenced GUIDs will no longer be found in the vault, except for the latest ones.

Q. What happens when I delete a Connection Profile or other object in EFT that had a secret attached. Will EFT delete that object's secret in the vault, assuming that secret isn't referenced by other objects?

A. No. EFT will not delete the old secret in the vault. This will result in an orphaned secret and admins should create a procedure to routinely clean up orphaned secrets in the vault. The process for identifying those orphans is possible but non-trivial (for now). Contact us for guidance.

Master Secrets

EFT Secrets: A Primer

Reversible-secrets stored locally in EFT's configuration database, or in-memory upon retrieval from a remote vault, are kept in encrypted form until they are needed. To encrypt these secrets, EFT uses a master secret, which is hard-coded into EFT's binaries. Prior to version 8.0.1, a single master secret was used. In version 8.0.1 and later, there is a separate master secret for Sites in addition to a Server master secret. These secrets cannot be encrypted (in theory you could, but then you would need to encrypt the secret used to encrypt those, ad infinitum). However, there is a way for an administrator to change the default, hard-coded master secret used by Sites to encrypt reversible secrets, but it requires working with our tier-3 support team due to the high risks involved (of essentially breaking one's existing config).

In real-world deployments, if the system EFT is running on has been compromised, then EFT, its file data, and its configuration data and secrets should be considered compromised, as a malicious actor could attach a debugger to capture moments when decryption occurs, or retrieve the master secret override (if present) stored in EFT's config, or perform several other exploits only possible when complete physical access is available. As a security best practice, limit physical access to only those who really need it, and use EFT's administrative roles to limit access to EFT's console based on the trust and needs of each particular administrator.

GlobalSCAPE Knowledge Base
https://kb.globalscape.com/Knowledgebase/11534/EFT-Secrets-A-Primer