

CoPilot Exchange on timestamping Access code signing

CR: Is the vba project signature in Access timestamped?

Yes, VBA project signatures in Access can be timestamped, but it depends on how the digital signature is applied. If you sign the VBA project manually using **Tools > Digital Signature** in the VBA editor, the signature can include a timestamp, ensuring its validity even after the certificate expires.

For automated signing, developers often use **Microsoft SignTool** with the **Microsoft Office Subject Interface Packages** to digitally sign VBA projects, including a timestamp. Additionally, some users configure timestamping settings in the registry under `\HKEY_CURRENT_USER\SOFTWARE\Microsoft\VBA\Security\`, specifying a timestamp server URL.

Would you like help setting up a timestamping service for your VBA projects?

CR: Yes - explain how to add the timestamp

To add a timestamp to your VBA project signature, follow these steps:

1. Configure the Registry for Timestamping

Microsoft Office uses registry settings to determine the timestamping service. You need to add or modify the following registry keys:

- Open **Registry Editor** (`\ regedit`).
- Navigate to:
`\HKEY_CURRENT_USER\SOFTWARE\Microsoft\VBA\Security\`
- Add or modify these values:
 - **TimeStampRetryCount** (DWORD) = `3`
 - **TimeStampRetryDelay** (DWORD) = `3`
 - **TimeStampURL** (String) = `http://timestamp.verisign.com/scripts/timestamp.dll` (or another trusted timestamp server)

2. Sign the VBA Project with a Digital Certificate

- Open your VBA project in Access.
- Go to **Tools > Digital Signature** in the VBA editor.
- Select your certificate and click **OK**.
- Save and close the VBA editor.

3. Verify the Timestamp

- When the VBA project is signed, Office will attempt to timestamp it using the registry settings.
- If successful, the signature will remain valid even after the certificate expires.

Alternative: Using Microsoft SignTool

If you need more control over timestamping, you can use **Microsoft SignTool**:

```
cmd
signtool.exe sign /f "CertificateFile.pfx" /p "YourPassword" /fd "SHA256" /tr
"http://timestamp.verisign.com/scripts/timestamp.dll" /td "SHA256" "YourVBAProject.accdb"
```

This ensures the VBA project is signed with a timestamp.

Would you like help troubleshooting any issues with timestamping?

CR: How does this work with a certificate stored on a secure token?

When using a certificate stored on a **secure token**, the signing process differs slightly because the private key is **not exportable** and remains on the token. Here's how it works:

1. Ensure the Token is Recognized

- The secure token must be plugged in and recognized by Windows.
- Check that the certificate appears in **Certificate Manager** (` certmgr.msc `).
- Some tokens require proprietary drivers or middleware (e.g., SafeNet, YubiKey).

2. Use the Token for Signing

- Open **Access VBA Editor** (` ALT + F11 `).
- Go to **Tools > Digital Signature**.
- Select the certificate stored on the token.
- Click **OK** and save the project.

3. Timestamping with a Secure Token

Since the private key **never leaves the token**, timestamping must be done using a **CSP** (Cryptographic Service Provider) or **KSP** (Key Storage Provider) that supports hardware tokens. If using **Microsoft SignTool**, the command would look like:

```
`` `cmd
signtool sign /tr "http://timestamp.verisign.com/scripts/timestamp.dll" /td SHA256 /csp
>YourTokenCSP" /kc "YourKeyContainer" "YourVBAProject.accdb"
`` `
```

4. Considerations

- Some tokens require **PIN entry** before signing.
- If the certificate is **non-exportable**, it cannot be used on another machine without the token.
- Ensure the token supports **SHA256** signatures, as older SHA1 certificates may be rejected.

Would you like help troubleshooting a specific token setup?