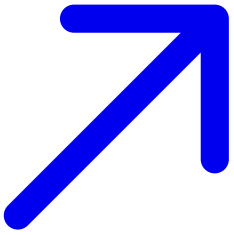
Office Templates and GlobalDotName - A Stealthy Office Persistence Technique | 0xShukruN

Published: 2022-10-12 · Archived: 2026-04-02 10:45:00 UTC

1. Blogs

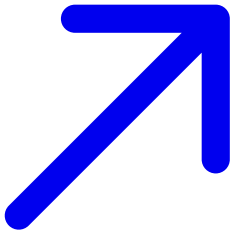
🔍Office Templates and GlobalDotName - A Stealthy Office Persistence Technique

A few weeks back, I was researching various adversarial techniques, when a couple of minutes into the research of

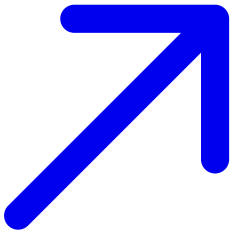


T1137 (Office Application Startup), there appeared to be a yet-to-be-documented capability that can be leveraged by adversaries using this technique.

I also noticed there isn't a lot of in-depth information about some of the techniques presented in T1137

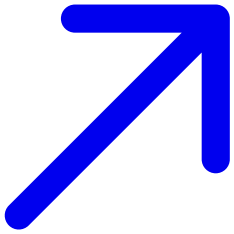


, such a the "Normal Template" technique, even though

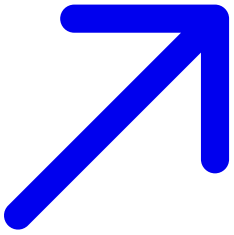


plenty of well known threat actors like "MuddyWater " leverage this technique, so I decided to shed some light about it and some of Word's inner workingarming a template file ws.

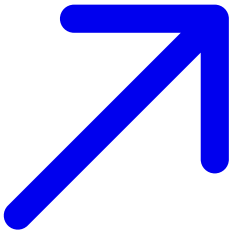
When researching adversarial techniques, I start by examining the MITRE ATT&CK Post-Exploitation matrix



to check the technique's existence, and gather additional information if exists on the web.



So at one point, I decided to research T1137 a.k.a "Office Application Startup".

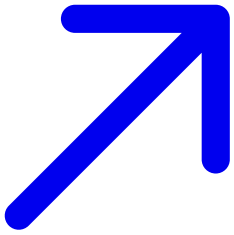


T1137 documented multiple implementations for achieving persistence in Microsoft Office application startup such as "Office Test" key, Add-ins, Templates, Rules & Home pages(Outlook), etc.

But today we're here to talk about Office Templates; Office Templates will eventually allow us to execute code on each application startup, **even on macro-free documents!** (docx, pptx, etc).

How Does this Technique Work?

It's relatively simple, every office application uses some form of a template, so if we will take Microsoft's Office Word for example, When executed, it loads its "Normal Template



" that contains default styles and customizations that determine the basic look of a document, and even active content such as macro.

By replacing the "Normal Template" of Word with an armed version of our own that contains VBA for example, we can achieve persistent code execution on the machine since the template will load each time a document is opened.

Later in this article, I will demonstrate how to arm your template, and how to make this technique even stealthier by using a custom template. We can define a custom template by using Word's UI or by editing the registry value "GlobalDotName" which we will learn about shortly.

One extremely cool thing about this technique is that the armed template **VBA code will be executed even when macro-free documents are executed!** (.docx,pptx, etc)

All Microsoft Office applications use templates, but for the sake of this article, we'll focus on Microsoft's Office Word. Word's default template file is named Normal.dot / Normal.dotm; it's also been referred to as "Normal Template".

The "Normal Template" is a file that contains default styles and customizations that determine the basic look of a document, and even active content such as macros.

In older versions of Word, Microsoft used a binary format (Word 2003 and below), and in Word 2007, Microsoft changed its format to an XML based format, hence the different extensions .dot/.dotx/.dotm.

The .dotm extension signifies that the "Normal Template" is a macro-enabled template which will come in handy later in this article ;).

The "Normal Template" is similar to other template files, aside from it being the default built-in template of Word, and some features which are not available in ordinary non-default templates.

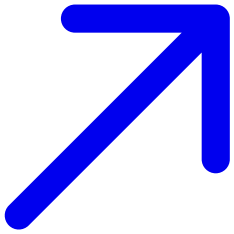
Word cannot open documents without a template, nor can it create new ones without it. Which is why Word has a built-in "Normal template" that would be re-created if Word can't find a template to use, or if the current template is corrupted/malformatted.

The default "Normal Template" is most commonly found in the **User Templates folde**r which is located at:

- *%appdata%\Microsoft\Templates*

However, this is not the only place Word searches for the "Normal Template". I examined Word's activity when executed and I found two locations that Word searches for the templates, one is Word's program folder, the other is the default location for the "Normal Template".

I did some googling and found an amazingly [in-depth article by Tony Jollans](#)



, Microsoft Word MVP, which mentioned another location that Word is searching for which is actually a registry value that point to a network-shared template.

Below is the a list of the templates possible location and their corresponding search order.

1.
2.
3. Workgroup Templates location
4. GlobalDotName Registry key*

*Will be detailed further down this article.

Word searches for the "Normal Template" in Word's Program folder which differs on different operating systems versions as well as Microsoft Office versions, you may find it in the following locations:

- *C:\Program Files\Microsoft Office\Office<version number>*
- *C:\Program Files\Microsoft Office\root\Office<version number>*

You can find out Word's Program path by its GUI or via the registry value ("vv.0" is placeholder for the version number):

- *HKCU\Software\Microsoft\Office\vv.0\Word\Options\PROGRAMDIR*

The "Normal Template" is rarely located at Word's program location,still, you should be aware of this location.

If Word can't find the "Normal Template" at Word's Program location it will attempt to find it at the User Template Location, and its default location is:

- *%appdata%\Microsoft\Templates*

If by any chance that location has been changed, you can discover the updated User Templates location by querying the following registry value:

- *HKCU\Software\Microsoft\Office\vv.0\Common\General\UserTemplates*

Workgroup Templates location

Finally, there is the Workgroup template, this option is available for sharing templates over the network and doesn't have any default value. If it configured, you can find the template's location in the following path:

- *HKCU\Software\Microsoft\Office\vv.0\Common\General\SharedTemplates*

When all else fails, and Word can not find its template, it generate a new one which is built-in to Word.

So far we've learned that:

- Word uses a default template called Normal.dot / Normal.dotm which can be found in a variety of locations and is loaded with each document's execution.
- The template can contain VBA code which will be loaded when a document execute, **even if it is a macro-free document.**
- What if I don't want to call my malicious template Normal.dotm?
- What if i want to give it an arbitrary name?
- or an arbitrary extension?
- And how about placing it in a location of my choosing?

Gain stealthiness with GlobalDotName

GlobalDotName is a registry value that when used, tells word the location of a custom "Normal template" of our choosing and provide us with a huge amount of flexibility.

This value can be found at:

- HKCU\software\microsoft\office\vv.v\word\options\GlobalDotName

How to set it up?

1. Create the Value of "GlobalDotName" in the relevant key
2. In the value's data input your path+filename+extension(optional)

GlobalDotName Highlights:

- **Takes priority over every other Normal template**
- Use any location you want
- Use almost any name/extension you want, they don't have to mean anything, yes, even the extension can be total nonsense. DO NOT however choose any meaningful extension to Word aside from the macro-enabled template(.dotm) which is not to be confused with the macro-enabled document(.docm)
- Well, you actually don't even have to choose an extension, it'll work anyway without it

- you can use a relative paths
- you can also use environment variables by using a REG_EXPAND_SZ Value

Arming a template file with VBA

1. Open a new document in Word.

2. Add the "Developer" tab, File -> Options:

3. Make sure to create your macro in the current document:

This macro will pop a message box

4. Once you write your macro, save the document as Normal.dotm (since we are not using GlobalDotName here):

5. Now before we implement the GlobalDotName Key and additional important steps(such as setting Trust for our template), lets drop it in the User Template Folder and execute **test.docx,** which is a normal macro-free document.

Modified Normal Template

6. And there we have it, a .docx document is opened and executes the macro from the template we just planted.

Beginning with Word 2007 and above Microsoft introduced a concept of "Trust" to improve its security and protect its users from unauthorized code running without their knowledge.

To be able to execute macro without the security warning popping out , you need to either disable the "VBAWarning" value in the registry, or you can add "Trust" to the relevant document by registering it in as "Trusted Document" or place it in a "Trusted location".

Luckily, we can control all of these options by editing some registry keys & values. Here are the relevant keys & values locations:

- HKEY_CURRENT_USER\SOFTWARE\Microsoft\Office\16.0\Word\Security
- HKEY_CURRENT_USER\SOFTWARE\Microsoft\Office\16.0\Word\Security\VBAWarning
- HKEY_CURRENT_USER\SOFTWARE\Microsoft\Office\16.0\Word\Security\Trusted Documents
- HKEY_CURRENT_USER\SOFTWARE\Microsoft\Office\16.0\Word\Security\Trusted Locations
- HKEY_CURRENT_USER\SOFTWARE\Microsoft\Office\16.0\Word\Security\Trusted Locations\ <key_name_of_location>\
- HKEY_CURRENT_USER\SOFTWARE\Microsoft\Office\16.0\Word\Security\Trusted Locations\ <key_name_of_location>\Path
- HKEY_CURRENT_USER\SOFTWARE\Microsoft\Office\16.0\Word\Security\Trusted Locations\ <key_name_of_location>\AllowSubFolders

In this case I will utilize the "Trusted Locations" key to set Trust for a certain location, which means the "VBAWarning" value and the "Trusted Documents" key are not needed for this proof of concept.

Setting up the persistence

- Open Word, create a new document, insert your macro to it and save it as a macro-enabled template file (.dotm) in the location you would point GlobalDotName to.
- Change the name & extension according to the rules we've established before and add the relevant path to the GlobalDotName value.
- Create a key nested within the "Trusted Locations" key, inside the new key, create a value named "Path" and place the path to the template file.
- (Optional) - A cool value that can make the directory you specified and all of its sub-directories trusted is the "AllowSubFolders"; this value is binary so activating it requires creating a DWORD value with the data equals to 1:

It is possible to to create the "C:\" drive as a location and allow sub-folders therefor making the entire disk trusted, which means no macro alerts whatsoever.

Let's execute a normal macro-free document from the desktop and see what happens

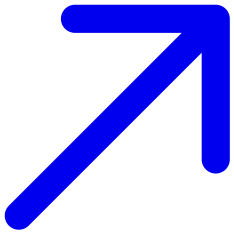
Success :)

You must create the "Trusted Location" key and place the relevant path value to the template's location before any document is executed. Otherwise the document execution will result in an error and the deletion of the crafted template. after the deletion, Word will re-create the "Normal template" from its defaults and place it in the location which GlobalDotName points to.

The usage of templates as means of persistence is leveraged by multiple threat actors and has been spotted in the wild before.

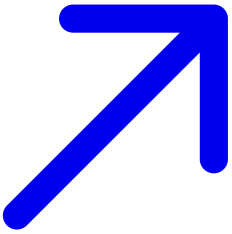
Having said that, I have yet to encounter the usage of the GlobalDotName which provides a somewhat stealthier implementation of this technique.

I highly recommend visiting the technique page on [MITRE ATT&CK](#)



which documents some of the threat actors that leverages this technique in-addition to links for various reports that presents its usage.

I want to extended my appreciation to Tony Jollans, Microsoft Word MVP, and an amazing researcher, which publishes tons of "Word Internals" material on his website,



wordarticles.com . His articles helped me immensely during my research.

If you discovered any mistakes or inaccuracies, please contact me :)
