hasherezade's 1001 nights

projects and tasks that I do in my free time

← Hook the planet! Solving FlareOn4 Challenge6 with libPeConv

Solving a PyInstaller-compiled crackme →

Process Doppelgänging – a new way to impersonate a process

Posted on December 18, 2017

Recently at Black Hat Europe conference, Tal Liberman and Eugene Kogan form enSilo lab presented a new technique called Process Doppelgänging. The video from the talk is available here. (Also, it is worth mentioning that Tal Liberman is an author of the AtomBombing injection).

This technique is a possible substituent of the well-known Process Hollowing (RunPE), that is commonly used in malware. Both, Process Doppelgänging and Process

Search

Recent Posts

- PE-bear – version 0.3.8 available
- White Rabbit crackme!
- Unpacking a malware with libPeConv (Pykspa case study)
- Solving a PyInstaller-compiled crackme
- Process Doppelgänging – a new way to impersonate a process

Archives

- April 2018 (1)
- February 2018 (1)
- January 2018 (2)

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Hollowing, gives an ability to run a malicious executable under the cover of a legitimate one. Although they both serve the same goal of process impersonation, they differ in implementation and make use of different API functions. This is why, most of the current antivirus solutions struggled in detecting Process Doppelgänging. In this post we will take a closer look on how the Process Doppelgänging works and compare it with the popular RunPE.

WARNING: Running this PoC on Windows 10 may cause a BSOD – the reason is a bug in Windows 10. Details here.

Process Doppelgänging vs Process Hollowing (aka RunPE)

The popular RunPE technique substitutes the PE content after the process is created (suspended), but before it is fully initialized. In order to implement this technique, we need to do by our own the step that WindowsLoader took so far: converting the PE file from it's raw form into a virtual form, relocating it to the base where it is going to be loaded, and pasting into the process' memory. Then, we can awake the process from the suspended state, and the WindowsLoader will continue loading our (potentially malicious) payload. You can find a commented implementation here.

The Process Doppleganging, in contrary, substitutes the PE content before even the process is created. We overwrite the file image before the loading starts – so, WindowsLoader automatically takes care of the fore-mentioned steps. My sample implementation of this technique can be found here.

NTFS transactions

- December 2017 (2)
- November 2017 (1)
- June 2017 (4)
- May 2017 (2)
- December 2016 (1)
- November 2016 (1)
- July 2016 (3)
- June 2016 (1)
- April 2016 (1)
- March 2016 (2)
- February 2016 (1)
- October 2014 (1)
- March 2014 (1)
- February 2014 (2)
- January 2014 (1)
- November 2013 (1)
- October 2013 (1)
- September 2013 (1)
- August 2013 (1)
- July 2013 (3)
- July 2012 (1)
- May 2012 (1)
- April 2012 (1)

Categories

- CONfidence (3)
- CrackMe (11)
- KernelMode (3)
- Malware (13)
- Malware Decryptor (5)
- PE-bear (11)
- Programming (4)
- Techniques (4)
- Tools (7)
- Tutorial (10)
- WKE (3)

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

On the way to it's goal, Process Doppelgänging uses a very little known API for NTFS transactions.

Transactions is a mechanism commonly used while operating on databases – however, in a similar way it exists in the NTFS file system. It allows to encapsulate a series of operations into a single unit. Thanks to this, multiple operations can be treated as a whole: they can either succeed as a whole – and be committed, or fail as a whole – and be rolled back. Outside of our transaction, the result of the operations is not visible. It starts to be noticeable after the transaction is closed.

Windows API makes several functions available for the purpose of transactions:

- CreateTransaction
- CommitTransaction
- RollbackTransaction
- CreateFileTransacted
- MoveFileTransacted
- DeleteFileTransacted
- CreateDirectoryTransacted
- RemoveDirectoryTransacted
- etc…

Briefly speaking, we can create a file inside a transaction, and for no other process this file is visible, as long as our transaction is not committed. It can be used to drop and run malicious payloads in an unnoticed way. If we roll back the transaction in an appropriate moment, the operating system behaves like our file was never created.

The steps taken

Blog Stats

- 190,957 hits

All my works included here are licensed under:

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Usage of NTFS transactions

Firstly, we need to create a new transaction, using the API CreateTransaction.

https://github.com/hasherezade/process_doppelganging/blob/master/main.cpp#L144

Then, inside of this transaction we will create a dummy file to store our payload (using CreateFileTransacted).

https://github.com/hasherezade/process_doppelganging/blob/master/main.cpp#L149

This dummy file will be then used to create a section (a buffer in a special format), which makes a base for our new process.

https://github.com/hasherezade/process_doppelganging/blob/master/main.cpp#L173

After we created the section, we no longer need the dummy file – we can close it and roll back the transaction (using RollbackTransaction).

https://github.com/hasherezade/process_doppelganging/blob/master/main.cpp#L188

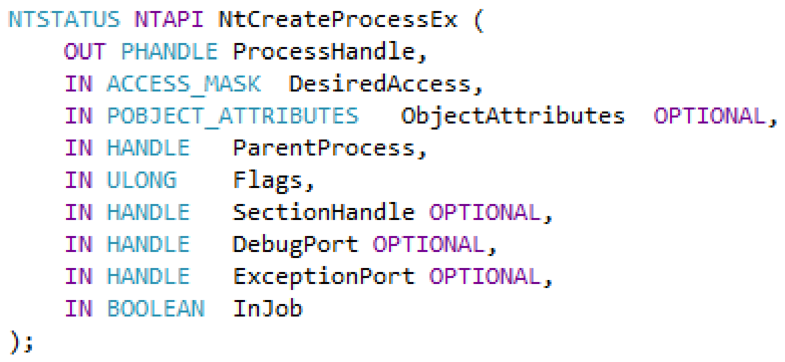
Usage of undocumented process creation API

So far we created a section containing our payload, loaded from the dummy file. You may ask – how are we going to create a process out of this? The well known API functions for creating processes on Windows require file path to be given. However, if we look deeper inside those functions, we will find that they are just wrappers for other, undocumented functions. There is a function `Zw/NtCreateProcessEx` which, rather than

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

the path to the raw PE file, requires a section with a PE content to be given. If we use this function, we can create a new process in a “fileless” way.

Definition of the NtCreateProcessEx:



```
NTSTATUS NTAPI NtCreateProcessEx (
    OUT PHANDLE ProcessHandle,
    IN ACCESS_MASK  DesiredAccess,
    IN POBJECT_ATTRIBUTES   ObjectAttributes  OPTIONAL,
    IN HANDLE   ParentProcess,
    IN ULONG    Flags,
    IN HANDLE   SectionHandle OPTIONAL,
    IN HANDLE   DebugPort OPTIONAL,
    IN HANDLE   ExceptionPort OPTIONAL,
    IN BOOLEAN  InJob
);
```

https://github.com/hasherezade/process_doppelganging/blob/master/main.cpp#L196

Creation of process by this way requires more steps to be taken – there are some structures that we have to fill and setup manually – such as process parameters (`RTL_USER_PROCESS_PARAMETERS`). After filling them and wring into the space of the remote process, we need to link them to the PEB. Mistake in doing it will cause the process to not run.

https://github.com/hasherezade/process_doppelganging/blob/master/main.cpp#L76

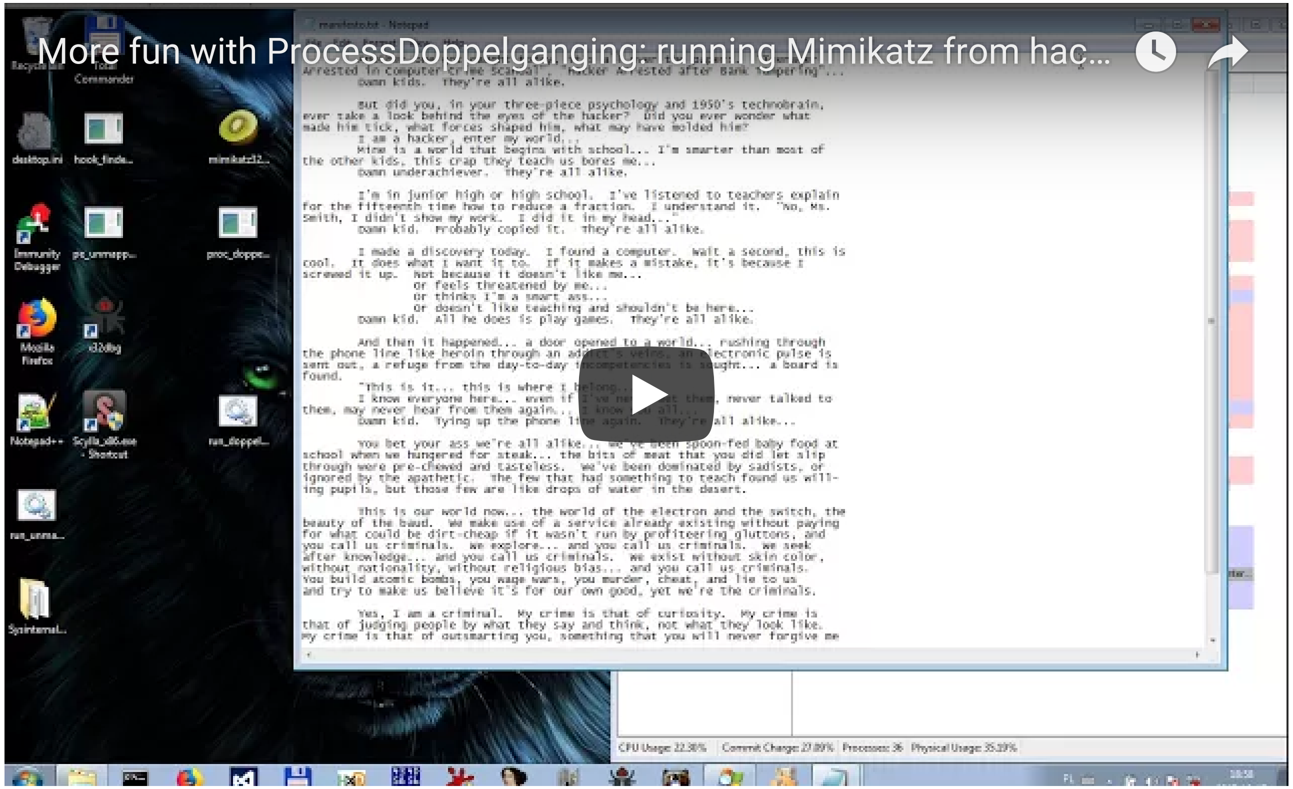
After setting everything up, we can run the process by creating a new thread starting from it’s Entry Point.

https://github.com/hasherezade/process_doppelganging/blob/master/main.cpp#L246

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Despite some inconveniences, creating the process by a low-level API gives also interesting advantages. For example, we can set manually the file path – making an illusion, that this is the file that has been loaded, even if it was not. By this way, we can impersonate any windows executable, but also we can make an illusion, that the PE file runs from a non-existing file, or a file of a non-executable format.

Below you can see an example where the illusion was created, that the PE file runs from a TXT file:

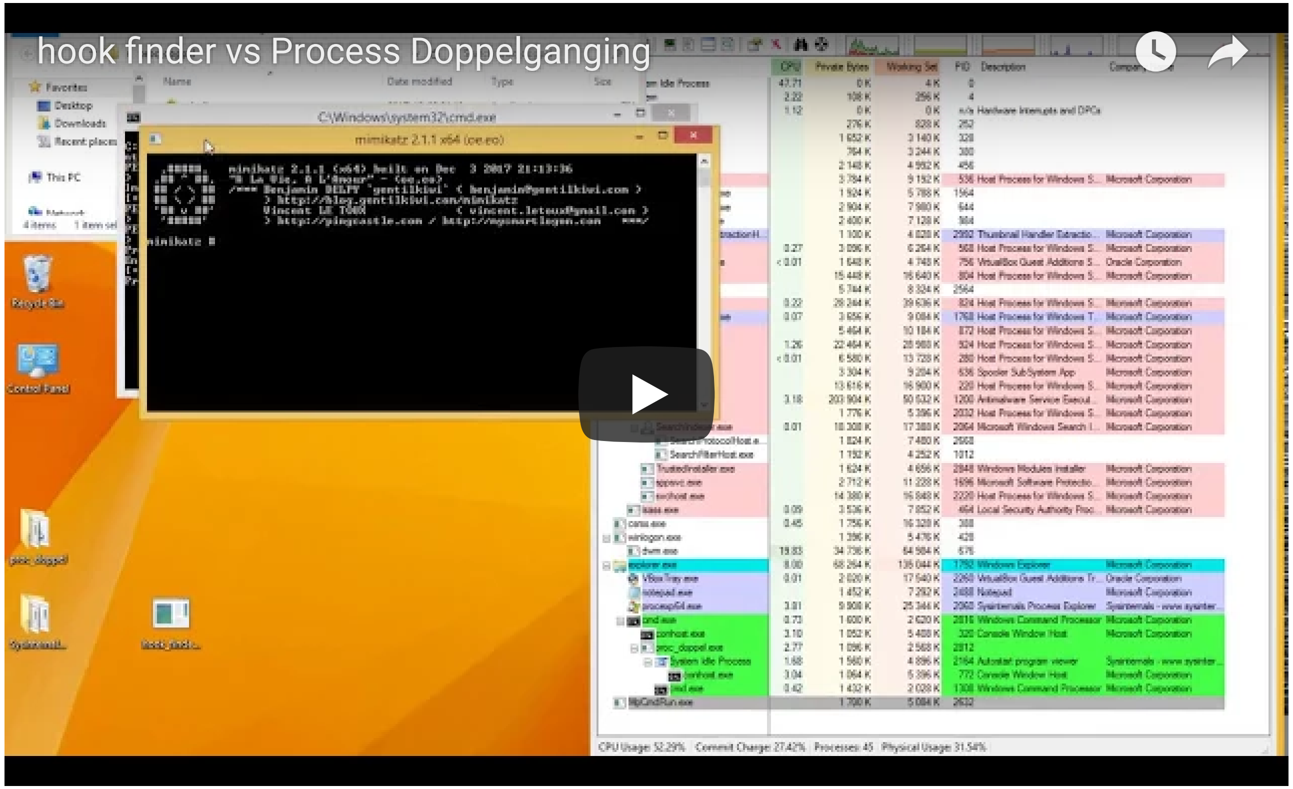


```
Arrested in Computer Crime Scandal", "Hacker Arrested after Bank Tampering"...
        Damn kids.  They're all alike.

        But did you, in your three-piece psychology and 1950's technobrain,
ever take a look behind the eyes of the hacker?  Did you ever wonder what
made him tick, what forces shaped him, what may have molded him?
        I am a hacker, enter my world...
        Mine is a world that begins with school... I'm smarter than most of
the other kids, this crap they teach us bores me...
        Damn underachiever.  They're all alike.

        I'm in junior high or high school.  I've listened to teachers explain
for the fifteenth time how to reduce a fraction.  I understand it.  "No, Ms.
Smith, I didn't show my work.  I did it in my head..."
        Damn kid.  Probably copied it.  They're all alike.

        I made a discovery today.  I found a computer.  Wait a second, this is
cool.  It does what I want it to.  If it makes a mistake, it's because I
screwed it up.  Not because it doesn't like me...
                Or feels threatened by me...
                Or thinks I'm a smart ass...
                Or doesn't like teaching and shouldn't be here...
        Damn kid.  All he does is play games.  They're all alike.

        And then it happened... a door opened to a world... rushing through
the phone line like heroin through an addict's veins, an electronic pulse is
sent out, a refuge from the day-to-day incompetencies is sought... a board is
found.
        "This is it... this is where I belong..."
        I know everyone here... even if I've never met them, never talked to
them, may never hear from them again... I know you all...
        Damn kid.  Tying up the phone line again.  They're all alike...

        You bet your ass we're all alike... we've been spoon-fed baby food at
school when we hungered for steak... the bits of meat that you did let slip
through were pre-chewed and tasteless.  We've been dominated by sadists, or
ignored by the apathetic.  The few that had something to teach found us will-
ing pupils, but those few are like drops of water in the desert.

        This is our world now... the world of the electron and the switch, the
beauty of the baud.  We make use of a service already existing without paying
for what could be dirt-cheap if it wasn't run by profiteering gluttons, and
you call us criminals.  We explore... and you call us criminals.  We seek
after knowledge... and you call us criminals.  We exist without skin color,
without nationality, without religious bias... and you call us criminals.
You build atomic bombs, you wage wars, you murder, cheat, and lie to us
and try to make us believe it's for our own good, yet we're the criminals.

        Yes, I am a criminal.  My crime is that of curiosity.  My crime is
that of judging people by what they say and think, not what they look like.
My crime is that of outsmarting you, something that you will never forgive me
```

How to detect?

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Although this technique may look dangerous, it can be easily detected with the help of any tool that compares if the image loaded in the memory matches the corresponding file on the disk. Example: detection with PE-sieve (former hook_finder):


hook finder vs Process Doppelganging
C:\Windows\system32\cmd.exe
mimikatz 2.1.1 x64 (oe.eo)

```
  .#####.   mimikatz 2.1.1 (x64) built on Dec  3 2017 21:13:36
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz #
```

The process of injection is also not fully stealthy. It still requires writing into the memory (including PEB) of the newly created process, as well as creating a remote thread. Such operations may trigger alerts.

In addition, the mechanism of NTFS transactions is very rarely used – so, if any executable call the related APIs, it should become an object of a closer examination.

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

So far this technique is new, that’s why it is not broadly recognized by AV products – but once we are aware of it’s existence, implementing detection should not be difficult.

Share this:

Twitter Facebook 33

★ Like

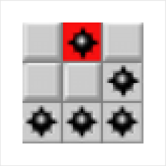
Be the first to like this.

Related

How to turn a DLL into a standalone EXE
In "Malware"

Starting with Windows Kernel Exploitation – part 3 – stealing the Access Token
In "KernelMode"

Hijacking extensions handlers as a malware persistence method
In "Malware"



About hasherezade

Programmer and researcher, interested in InfoSec.

View all posts by hasherezade →

This entry was posted in Malware, Programming, Techniques and tagged PE, Process Doppelgänging. Bookmark the permalink.

← Hook the planet! Solving FlareOn4 Challenge6 with libPeConv

Solving a PyInstaller-compiled crackme →

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

3 Responses to *Process Doppelgänging – a new way to impersonate a process*

lallouslab *says:*

December 31, 2017 at 3:17 am

Informative and straight to the point, thanks!

Reply

Pingback: *PROCESS DOPPELGANGING | Security and its occultism*

Pingback: *Examining and Detecting the Doppelganging Malware Evasion Technique - CorrelationX Blog*

Leave a Reply

Enter your comment here...

hasherezade's 1001 nights

Blog at WordPress.com.

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD