RESOURCES • BLOG

THREAT DETECTION

Remote access tool or trojan?

GET A DEMO ›

detect misbehaving RATs

The only difference between a remote administration tool and a remote access trojan (RAT) is who's controlling it.

JUSTIN SCHOENFELD • JASON KILLAM

Originally published August 19, 2021. Last modified October 1, 2024.

Red Canary's Cyber Incident Response Team frequently observes adversaries abusing legitimate remote access utilities for lateral movement and execution of payloads. These tools perform reliably, as you may expect with most enterprise software, and allow operators to pivot and transfer data to and from victim machines.

Adversarial abuse of remote monitoring & management (RMM) software is **not new**, but—given the rash of costly and destructive ransomware attacks in recent months and years—it's particularly important that security teams develop robust security controls for detecting malicious use of RMM tooling. In fact, just last week **AdvIntel** reported on adversaries who—after gaining initial access—had installed an RMM tool called Atera and used it as a functional backdoor in the lead up to a Conti ransomware outbreak.

Similar to how we detailed the **various exfiltration tools** used by adversaries during ransomware extortion, in this post we'll discuss why it's important to monitor RMM software in your enterprise, and we'll offer detailed guidance on how to observe and detect it.

RAT v. RAT

GET A DEMO ›

that can be tricky. With each of these tools, you'll need to "know normal," **as SANS says**, and ask behavioral questions of each tool. Start with simple questions like:

- Which directory does it normally execute from?
- Where does it normally establish connections to?
- What do these tools typically write to the filesystem?
- Should I expect child processes?
- Is this software approved in my environment?

The goal of this admittedly non-comprehensive list of questions is to detect abnormalities in your telemetry. Basically, you will take a baseline of the application's behavior and apply best efforts to writing detection logic on commonly observed techniques that adversaries employ. Detection engineers want to cut through the noise as much as possible and find anomalies in vast data sets. On top of this, each tool's execution behaviors vary, which makes answering the questions above a time-consuming process for analysts.

NetSupport

In the case of one remote administration tool, known as "NetSupport," malicious usage can usually be distinguished by a few factors:

- connections to malicious domains
- renaming the primary NetSupport binary, `client32`, to something else
- executing from directories outside of the usual "program files" folder

In the below instance, we would qualify the copy of NetSupport as "suspect" based on the directory and process name.

Note: `ctfmon.exe` *is a renamed version of NetSupport binary.*

GET A DEMO ›

Adversaries frequently rename binaries, as shown above, in an effort to evade overly specific detections. However, these binaries often depend on dynamic link libraries (DLL) that cannot be renamed, which make for useful indicators of the tool being abused. Analysts can search for the utility in question on a site like VirusTotal to get a list of associated DLLs the utility loads at run time, and these module loads may be reliable indicators that the utility is being used, even if it has been renamed and is running from an unusual path.

In this example, `ctfmon.exe` is the renamed `client32` NetSupport binary, which you can see by the hashes of the two binaries.

GET A DEMO ›

While from a defender's perspective this all might seem obvious, you'll have to remember that this is all happening in the background from a user's perspective. And if a user runs a tool like Task Manager, all they'll see is `ctfmon` without the path.

Finally, this instance of NetSupport makes network connections to a couple domains: first to `geo.netsupportsoftware.com`, which is used by the client to report back the client's location, and second to a randomly named domain being used for its command and control (C2).

Detection opportunity 1

We suggest writing, implementing, and testing the following rules within your EDR platform or SIEM:

NetSupport executing from unexpected directories:

description = `'netsupport client application'` & process_path != `"program files"`

GET A DEMO ›

internal_name: `client32.exe` & process_name != `client32.exe` (the internal name used by the primary NetSupport executable)

NetSupport making external network connections:

Any external network connections to `geo.netsupportsoftware.com` (if you do not use NetSupport normally)

Remote Utilities

Remote Utilities is a remote desktop suite known to the security community as "RURAT" when used in a malicious context. Execution from folders outside of "program files"—such as `appdata` or `programdata`—often indicates malicious use of Remote Utilities. If you do not use Remote Utilities within your environment, alert on the execution of `rutserv.exe` or `rfusclient.exe` on all hosts within your environment. In the wild, it has been abused by various ransomware groups such as **Epsilon Red**, **TA505**, and even some suspected **state-sponsored adversaries**.

From a network alerting perspective, look for connections on **ports 5655 and 5650**, since they're not usually used for anything other than remote utilities. For alerting on host execution behavior, refer to the detections below.

In this case, we saw the `rutserv` binary running from a folder in `AppData\Roaming`.

GET A DEMO ›

In another case, we saw the remote utilities binary being renamed and using a maliciously created service named `USBPrintManagerGrp` for persistence.

GET A DEMO ›

Detection opportunity 2

We suggest writing, implementing, and testing the following rules within your EDR platform or SIEM:

Rutserv suspicious directory:

Process appears to be `rutserv.exe` or `rfusclient.exe` (or a renamed version of it) & path != `program files\remote utilities`

Rutserv renamed:

description = `Remote Utilities - Host` & process_name != `rutserv.exe` or `rfusclient.exe`

External network connection:

GET A DEMO ›

ScreenConnect

The ScreenConnect software (aka **ConnectWise Control**) has been leveraged in various cyber attacks since **at least 2016**. The application is feature-rich, allowing for remote management of hosts typically used for help desk support. Some notable features include drag-and-drop file transfers, screen recording, and access to the command line to execute custom commands.

Typically, adversaries use ScreenConnect to deploy malicious payloads like **Zeppelin** and **Sodinokibi** ransomware to numerous hosts across victim environments. It has been linked to a sophisticated, suspected state-backed **cyber espionage operation** against foreign entities wherein the software was used to launch automated actions after installation. Additionally, ScreenConnect was pivotal in the **attacks against Wipro back in 2019**, when attackers used it to conduct gift card fraud.

While we're not fully aware of the technical aspects of these attacks, we can start to generate detection logic surrounding abnormal behavior. Start with a broad question: what can I expect from normal execution of ScreenConnect? This may be a tough question to answer in your environment, depending on whether or not the tool is approved for use. If it's not approved, your pseudo-detection logic may be as simple as: *Detect whenever ScreenConnect executes*.

Based on our own telemetry and intelligence gained from past **incident response engagements**, we've found that it is highly unusual for ScreenConnect or its child processes to write executable files to disk.

We detected ScreenConnect deploying ransomware to endpoints by writing and executing them directly to the temp directory commonly used by the application:

GET A DEMO ›

ScreenConnect allows users to enter custom commands to be executed within the context of a remote session. Whenever a user executes a command within the ScreenConnect command shell, it creates and executes a script file ending with `run.cmd` under the temp directory. An example of a commonly used file name and directory may look like the following:

```
C:\WINDOWS\TEMP\ScreenConnect\6.9.22228.6977\1eb49efb-9375-4da2-a96c-5a935a1e2d42run.cmd
```

More information about this execution chain can be found in **this blog** by the Morphisec team. The script file is eventually executed via `cmd.exe`.

A practical detection rule could be based on identifying the following:

- process command lines containing `ScreenConnect`
- a command line including `run.cmd`
- a child process writing a binary file to disk

We have observed initial payloads designed to drop ransomware families such as **REvil/Sodinokibi** via the following process chain:

GET A DEMO ›

Detection opportunity 3

We suggest writing, implementing, and testing the following rules within your EDR platform or SIEM:

ScreenConnect suspect child lineage:

Parent_process = `cmd.exe`

parent command line includes = `run.cmd` & `screenconnect`

Process = `cmd.exe`

Childprocess = `powershell.exe`

GET A DEMO ›

ScreenConnect writing binaries to `C:\Windows\Temp\screenconnect\<Version Number>\*`

ScreenConnect spawning binaries:

ScreenConnect spawning binaries from the following file path with many file modifications: `C:\Windows\Temp\screenconnect\<Version Number>\*` (This may indicate successful ransomware execution).

Anydesk

Anydesk markets itself as a cross-platform, fast, and secure remote desktop application capable of performing reliably under bandwidth constrained network connections. Due to its simplicity, Anydesk has recently become a de facto RMM tool used by attackers due to its lightweight footprint and ease of use for moving laterally.

Anydesk is a popular tool for controlling victim machines and deploying ransomware payloads, more commonly seen with the following ransomware families: **Blackheart**, **Sodinokibi/REvil**, **Netwalker**, and **Darkside**. Attackers will typically drop instances of Anydesk to conspicuous paths not normally observed. Detecting abnormal Anydesk behavior can be fairly simple in most instances, since it has fairly predictable behavior when executed.

Under normal circumstances, Anydesk typically executes from these paths:

- `C:\Program Files (x86)\anydesk\anydesk.exe`
- `C:\Program Files\anydesk\anydesk.exe`
- `C:\Users\<User>\Downloads\anydesk.exe`

Conversely, we've identified Anydesk being dropped in the following file paths during active ransomware incidents:

GET A DEMO ›

- `C:\ProgramData\anydesk.exe`
- `C:\ProgramData\Adobe\Setup\AnyDesk.exe`

Our first suggestion for defenders is to monitor your endpoints for relocated instances of Anydesk. Given that Anydesk may legitimately run from custom file paths, this may be an imperfect detection solution in some environments. However, you should still investigate and identify the origin from which Anydesk was written. If the activity turns out to be benign, standard tuning practice may apply.

While investigating activity in compromised environments, we identified some anomalies in what files Anydesk writes to the filesystem. We created additional detection logic surrounding these anomalies, which revealed additional instances in other victim environments. Note that it's highly abnormal for AnyDesk to write executable files to disk besides `gcapi.dll`, which is a legitimate DLL that's part of the Google Chrome web browser used to interact with the Google Cloud API.

We also noticed that in legitimate use, setting custom session passwords via the command line is highly abnormal. We observed the following command during a suspected pre-ransomware engagement:

Detection opportunity 4

We suggest writing, implementing, and testing the following rules within your EDR platform or SIEM:

GET A DEMO ›

Relocated AnyDesk execution:

AnyDesk executing from a process_path != `C:\Program Files (x86)\anydesk\anydesk.exe`
or `C:\Program Files\anydesk\anydesk.exe` or `C:\Users\<User>\Downloads\anydesk.exe`

AnyDesk writing binaries:

AnyDesk writing any files not named `gcapi.dll`

`cmd` piping a password into an Anydesk session via `cli`:

```
C:\WINDOWS\system32\cmd.exe /C cmd.exe /c echo J9kzQ2Y0qO |C:\ProgramData\anydesk.exe --set-password
```

As an added bonus, you may have noticed that the Windows RDP client (`mstsc.exe`) spawned this instance of `cmd.exe`, which led to Anydesk execution. This may be detectable on its own with the following bonus analytic, although this may also require tuning or generate high levels of false positives in certain environments!

Bonus detection opportunity

Suspicious process lineage:

Parent_process = `Mstsc.exe` & process = `cmd.exe` & child_process = `cmd.exe`

Parent_process = `Mstsc.exe` & process = `powershell.exe` &

GET A DEMO ›

Parent_process = `Mstsc.exe` & process = `powershell.exe` & child_process = `reg.exe`

Parent_process = `Mstsc.exe` & process = `powershell.exe` & child_proces = `schtasks.exe`

Happy hunting!

We hope this blog has been helpful in demystifying some aspects of detecting abnormal usage of commonly abused RMM tools. Along with NetSupport, Remote Utilities, ScreenConnect, and Anydesk, we have observed ransomware operators using the following additional RMM tools over the years:

- Splashtop
- Atera
- LogMeIn
- TeamViewer
- Pulseway
- RemotePC
- Webroot SecureAnywhere
- PCAnywhere
- Kaseya
- GoToMyPC

The detection logic above may be applicable to these tools as well.

As always, keep in mind that each detection strategy presented in this post is not perfect and will likely require **extensive tuning** on your part. This detection guidance wouldn't be possible without the great analysis and reporting being performed by researchers across the industry each and every day. So keep on fighting the good

GET A DEMO ›

RELATED ARTICLES

THREAT DETECTION

Artificial authentication: Understanding and observing Azure OpenAI abuse

THREAT DETECTION

Apple picking: Bobbing for Atomic Stealer & other macOS malware

THREAT DETECTION

Keep track of AWS user activity with SourceIdentity attribute

GET A DEMO ›

THREAT DETECTION

Trending cyberthreats and techniques from the first half of 2024

GET A DEMO ›

Subscribe to our blog

You'll receive a weekly email with our new blog posts.

First Name

Last Name

Email Address

SUBSCRIBE ›

GET A DEMO ›

Get a Demo

See Red Canary in action

Schedule your demo now

Search

PRODUCTS

Managed Detection and Response (MDR)
Readiness Exercises
Linux EDR

SOLUTIONS

Deliver Enterprise Security Across Your IT Environment
Get a 24×7 SOC Instantly

GET A DEMO ›

What's New?
Plans

Protect Your Users' Email, Identities, and SaaS Apps
Protect Your Cloud
Protect Critical Production Linux and Kubernetes
Stop Business Email Compromise
Replace Your MSSP or MDR
Run More Effective Tabletops
Train Continuously for Real-World Scenarios
Operationalize Your Microsoft Security Stack
Minimize Downtime with After-Hours Support

RESOURCES

View all Resources
Blog
Integrations
Guides & Overviews
Cybersecurity 101
Case Studies
Videos
Webinars
Events
Customer Help Center
Newsletter

PARTNERS

Overview
Incident Response
Insurance & Risk
Managed Service Providers
Solution Providers
Technology Partners
Apply to Become a Partner

COMPANY

About Us
The Red Canary Difference
News & Press
Careers – We're Hiring!
Contact Us
Trust Center and Security

GET A DEMO ›

GET A DEMO ›