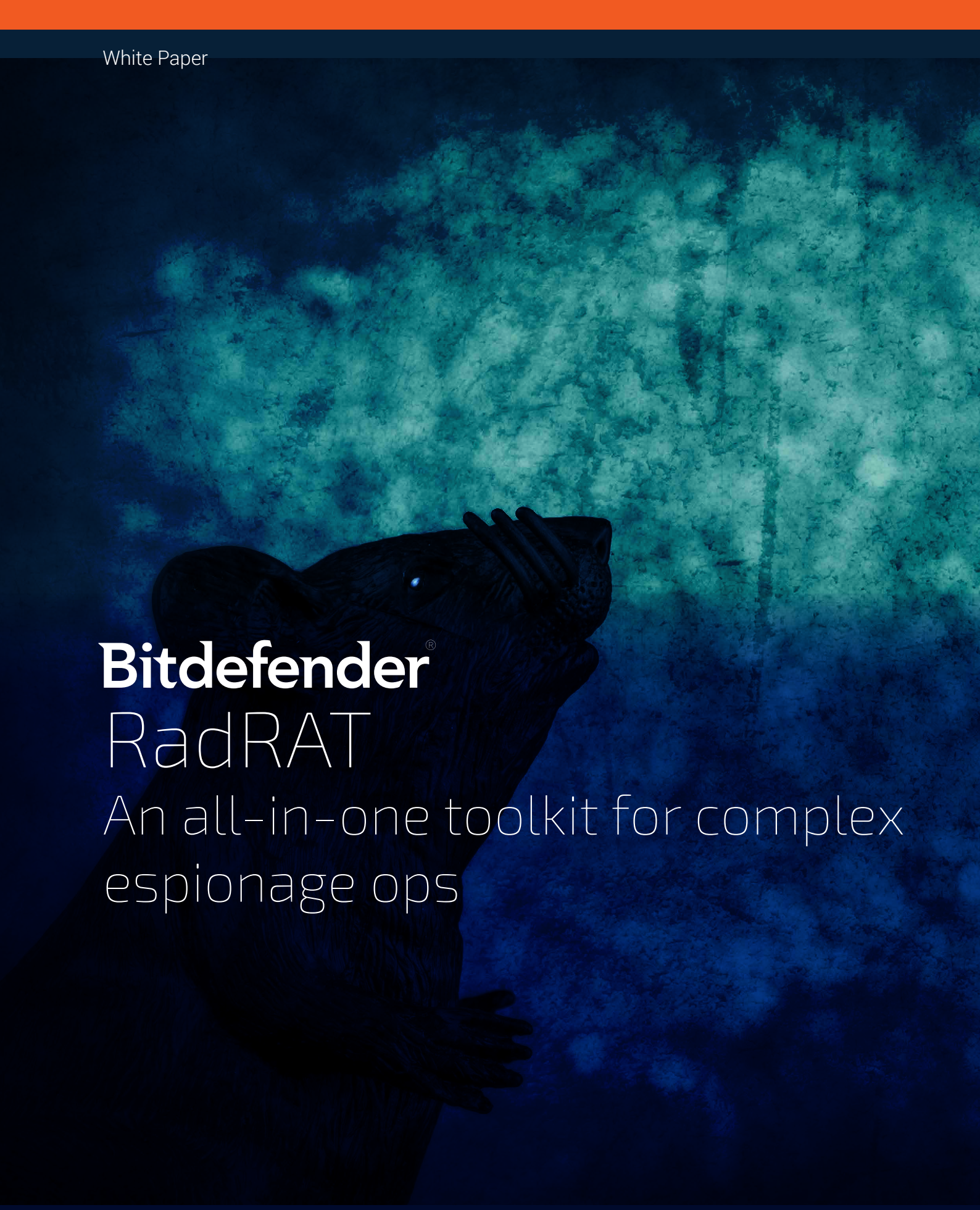
White Paper



Bitdefender®

RadRAT

An all-in-one toolkit for complex espionage ops

Author: Eduard BUDACA - Forensics Engineer, Cyber Threat Intelligence Lab

Executive summary

Around February this year, we came across a piece of malware that had previously gone unnoticed. Buried in the malware zoo, the threat seems to have been operational since at least 2015, undocumented by the research community.

Our interest was stirred by its remote access capabilities, which include unfettered control of the compromised computer, lateral movement across the organization and rootkit-like detection-evasion mechanisms. Powered by a vast array of features, this RAT was used in targeted attacks aimed at exfiltrating information or monitoring victims in large networked organizations.

In addition to its very powerful data exfiltration mechanisms, RadRAT features extremely interesting lateral movement mechanisms that include:

- Mimikatz-like credentials harvesting from `WDigest.dll` and `kerberos.dll`;
- NTLM hash harvesting from the Windows registry, inspired from the source code of the Mimikatz `lsadmp` tool;
- Using the infected machine to retrieve a Windows password from the LanMan (LM) hash, by cracking previously sniffed NTLM authentication challenges;
- An implementation of the Pass-the-Hash attack on SMB connections.

Deep dive inside the sample

The sample we examine throughout this analysis has a SHA256 of 4786fa468111632ea66f03dfd868ca95fb91d4472b2c332d46d8444c19c75624 and is closely related with several other payloads listed in the IOC appendix of this paper.

Unfortunately, while our information about the behavior and technical implementation of this remote access toolkit is complete, we can only guess at the original infection vector, which is most likely a spear phishing e-mail or an exploit.

Overview

This binary file is comprised of a set of tools (either embedded or downloaded along the way) that the attacking party uses for various malicious purposes. The toolkit is made of two main components and four secondary utilities, as follows:

Main components - these components perform most RAT/APT behavior:

- `wrpcs.dll`, usually running as the DcomLaunch service (forwarding ServiceMain and other methods to rpcss.dll, the DcomLaunch original service binary).
- `ntmgr2.dll`, usually running inside a new `Sysmgr` service (not a legitimate Windows service)

Secondary tools - these components are only used when needed:

- `defrag.exe`, stored as a resource in both `wrpcs.dll` and `ntmgr2.dll`, executes several RAT commands that require a separate process
- `sysmgr.exe`, stored as a resource in `ntmgr2.dll` and downloaded by `wrpcs.dll`, is a service executable file that invokes the `ntmgr2.dll` component
- `~rs19.tmp` (downloaded by the `wrpcs.dll` component on request) is a DLL called to restart the RAT in its update process
- `~rs.tmp` (stored as a resource in `ntmgr2.dll`) is an EXE file called in its update process

The main components also depend on the SSLeay and WinPcap libraries, which are downloaded on request.

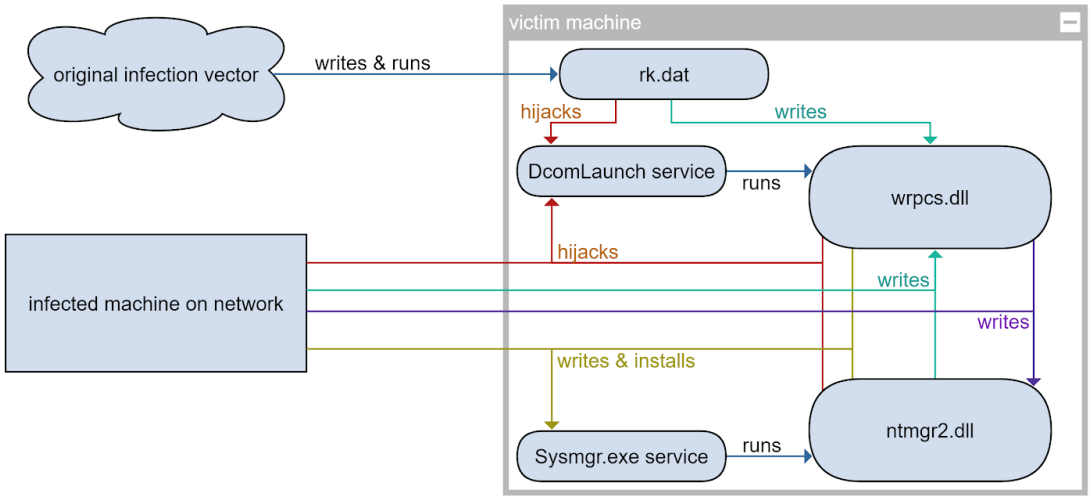
The sample identified with the above SHA256 is the `wrpcs.dll` component, but it is extremely similar to the `ntmgr2.dll` component in both code and behavior. In `wrcps.dll` mode, a `msrpcnm` file mapping is created as a marker. If the `ntmgr2.dll` component does not find this mapping when starting, it undertakes C&C communication duties as a fallback.

An interesting aspect in the operation of this threat is that it behaves differently based on its file name. This situation can be summarized as follows:

- Usually, the DLL's filename is `wrpcs.dll`, running as the DcomLaunch service. In this mode, this component performs the usual RAT tasks: updating itself, reading parameters from the registry, connecting to the C&C server, executing commands, and infecting other machines across the network.
- When the DLL's filename is `rk.dat`, this component runs in installation mode: the tasks are similar to the `wrpcs.dll` mode, but most validations and timeouts are ignored, so all tasks are executed.
- If the DLL is named `ir.ldc`, no RAT-specific actions are performed, and the service is only installed if the host process is `svchost.exe`.
- When the malware is named `mwcpi.dll`, the component is running in `AppInit_DLL` mode (it is injected in almost every process), and installs some hooks that hide the resource usage of the `ntmgr2.dll` component by reallocating CPU cycles from itself to the System Idle Process (rootkit-like behavior). Besides this unusual behavior, the component also gathers Windows and network credentials.

- When this DLL is named `msvcr71.dll`, the `wrpcs.dll` service DLL is updated (a newer version is downloaded and set as DcomLaunch DLL) and its host process is terminated.

A summary description of the infection process is illustrated below:



Persistence

The method of choice that currently discovered RadRAT variants employ for persistence is via services. These components replace the DcomLaunch service DLL from the legitimate `Windows\System32\rpcss.dll` binary to the malicious `wrpcs.dll`, saved as `Windows\System32\wrpcs.dll`. Also, a new service dubbed Sysmgr (displayed as `System Device Manager`) is created, running the `sysmgr.exe` component, which loads the `ntmgr2.dll` file. Both files are copied in `Windows\System32`.

The parameters used by the malware in its malicious actions are stored in the Windows registry under `HKEY_LOCAL_MACHINE\Software\Microsoft\Rad3Dev\Dev1`. Each value under this key serves a special purpose:

- `Det` controls whether a copy of the `wrpcs.dll` component will be added as `Windows/mwcpi.dll` to the `AppInit_DLLs` registry entry (in `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Windows`). The default value is `N`; if it is set to `Y`, this DLL gets injected in every process, adding malware-controlled hooks that harvest credentials and conceal resource usage.

- `Npd` enables Mimikatz-like credentials harvesting by injecting code into `lsass.exe`'s lsasrv DLL. The default value is `Y`, which enables this behavior.

- `Mt2` determines the automatic capabilities of infecting other machines across this network. Values range from 0 to 4 and determine the extent of the infection process. Since the default value is 0, this automatic infection is disabled unless explicitly requested.

- `Gap` controls the frequency of contact with the C&C servers. The default value is 20 minutes between queries, while the maximum value is 2 hours.

- `Dow` selects a day of the week when the component will contact the C&C servers. Valid days are Monday to Friday, but this behavior is disabled by default. The `rk.dat` mode ignores this check.

- Entries `Pw` and `Wce` (each of them `Y` or `N`) are unused, but their values are still sent to the attackers' servers in the connection process.

- `Src` stores a log of each update operation, with the process & DLL name that performed it, a timestamp, and whether it was performed by itself or another machine on the network.

Some persistent information, however, is stored in files on the machine. These files are enumerated below:

- `Windows\temp\~sdc978` stores NTLM authentication cracking requests, their progress, and their results. This file is encrypted with a XOR 0x63 operation.
- `Windows\temp\~adre379` stores captured SMB and RDP traffic waiting to be sent to the C&C servers.
- `Windows\temp\cfwe247` stores credentials stolen by components in AppInit_DLL mode (injected in other processes).

Main RAT behavior

The remote access tool component is undoubtedly the centerpiece of this attack toolkit. The following behavior is expected when the main component is running as `wrpcs.dll` or `rk.dat`.

AppInit_DLLs injection

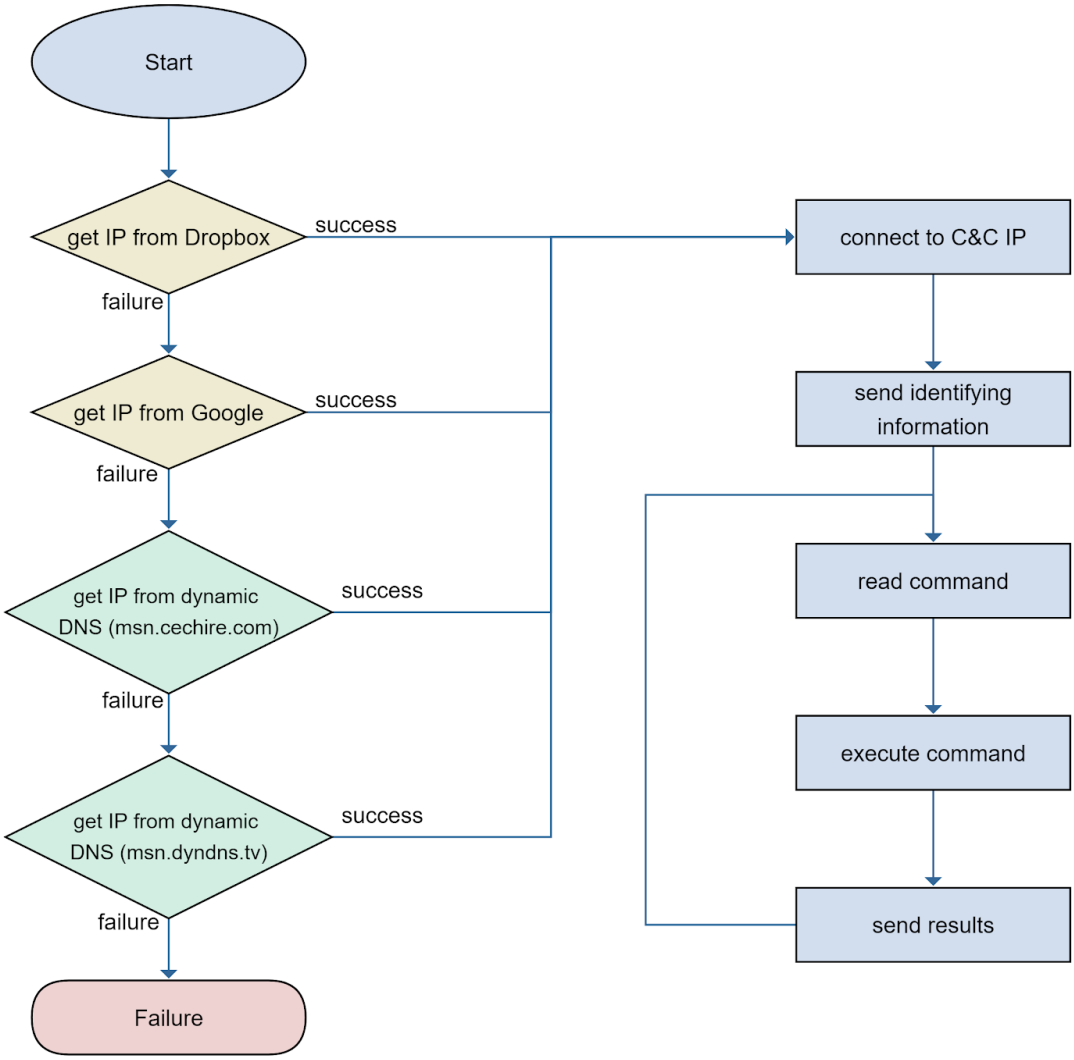
First, RadRAT reads the parameters in the Rad3Dev registry entry. Then, depending on the value of the Det flag, AppInit_DLLs injection is performed.

If this value is set, this component searches for an existing `mwcpi.dll` file inside the Windows directory. If this file does not exist or is an older version (the version is the 32-bit little-endian value which immediately follows the `CD AB CE BB` byte sequence marker), the `wrpcs.dll` component is copied as this file, then added to the AppInit_DLLs list under `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Windows`. Under the same registry key, two additional values are changed: `LoadAppInitDlls`, which enables the AppInit_DLLs mechanism, is set to 1, and `RequireSignedAppInitDlls`, which mandates that all such DLLs must be digitally signed, is set to 0.

CNC connection

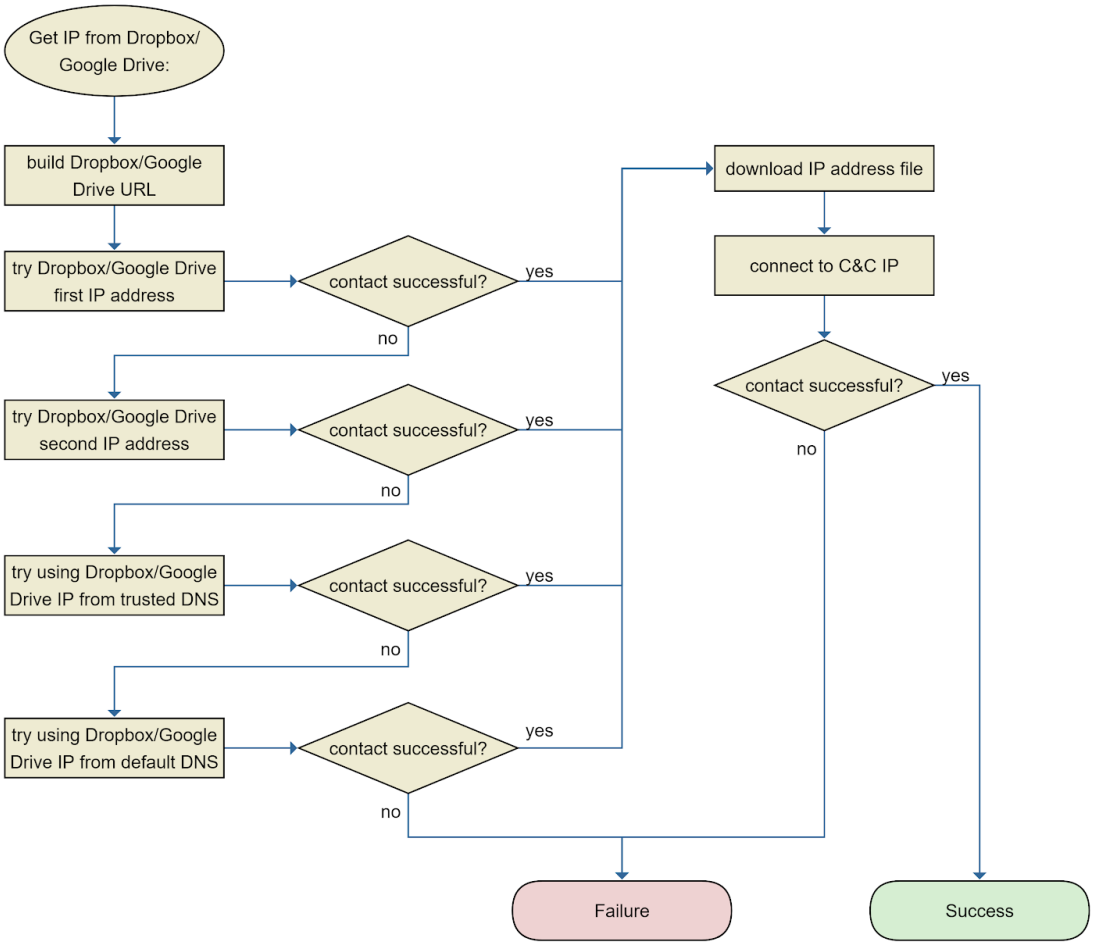
This threat avoids using default DNS requests or organization-controlled DNS, preferring the use of a hardcoded list of DNS nameservers while resolving the C&C server's IP address. This list corresponds to the nameservers of Dyn (a popular dynamic DNS provider, which seems to be favored by this group of threat actors), as well as the nameserver of MTNL, the Indian ISP that owns the attackers' IP addresses.

A visual representation of the steps the RAT takes to establish communication with the command and control infrastructure is pictured below.



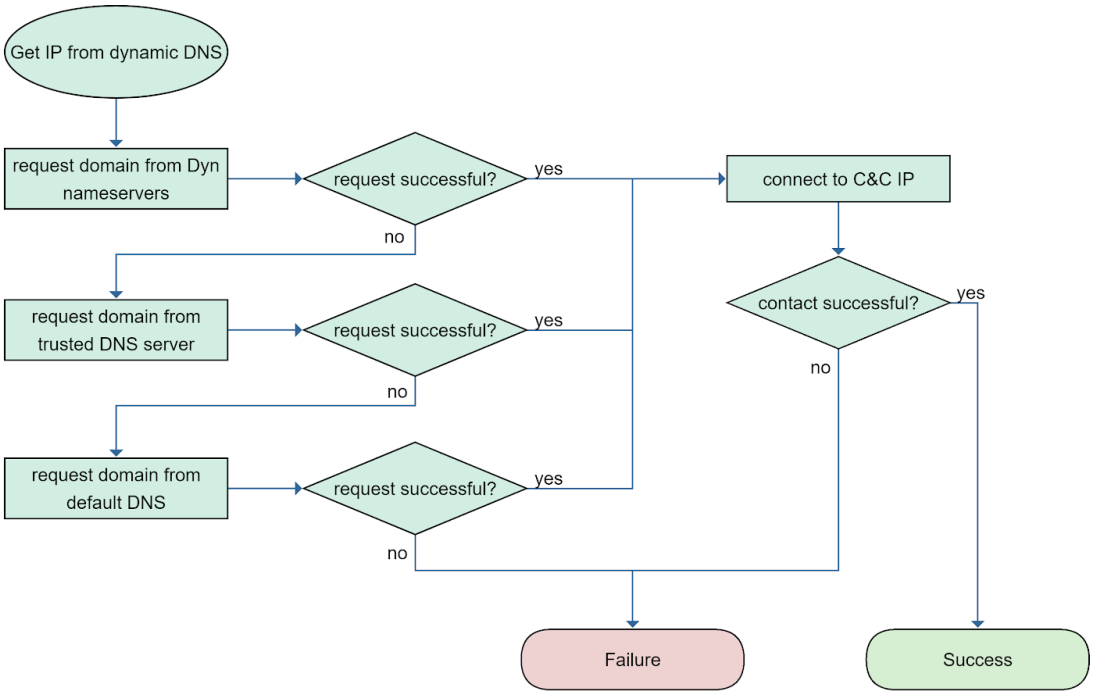
As an additional method to avoid traceable DNS resolutions, the attackers can provide an IP address via hardcoded Google Drive or Dropbox URLs (`https://dl.dropboxusercontent.com/s/xrkpsgq154qx8ko/pi0?dl=1` or `https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaelFTajB2Umw0b0k`). This component first attempts to use hard-coded lists of IPs of both Dropbox and Google Drive, then a trusted DNS server (controlled by BSNL, an Indian ISP), then the default DNS server for the system. The file available at these URLs contains an IP address, encoded XOR `0x12341234`, as a signed decimal number and a flag that enables or disables the use of DNS resolution if the connection to the given IP fails.

A flowchart of the process of receiving an IP address from Dropbox/Drive is depicted below:



When the IP cannot be obtained when the connection fails, this component uses the Dyn nameserver IPs to resolve its C&C dynamic DNS domains: `msn.cechire.com` and `msn.dyndns.tv`. At the time of writing, the IP address downloaded from Google Drive or Dropbox matched the resolutions of the dynamic DNS subdomains above. Also, the attackers seem to lack a static IP address, as the resolution changes between addresses in the same Autonomous System (AS).

This is a visual representation of the process of obtaining the C&C IP address from Dynamic DNS services:



Once this component has downloaded or resolved an IP address, it tries connecting on port 443, via raw TCP sockets, then on port 80, as HTTP POST messages. The identifying information sent to the C&C server consists of version, computer and user names, the DLL and host process filenames, registry parameters, and whether the system is 64-bit or not. This component expects the response code `0xA1B19988`. Once the connection is established, the server can start sending commands.

RAT commands

RadRAT's current command set supports 92 instructions, some of which are only available to one of the two main components, `wrpcs.dll` or `ntmgr2.dll`. These commands can be split into multiple categories:

File or registry operations

The attacker can use these commands to gain specific knowledge about the file layout and registry data of the victim machine or of network connected machines. The attacker has the ability to read any file, list the shares of machines on the network, obtain a list of files inside a directory, or get their sizes. Some advanced commands operate on chunks of larger files, being able to read them, compute and compare hashes of byte sections inside the file, and upload them in case of an unknown hash. Other commands can also allow the modification of the filesystem, enabling the creation, deletion, duplication or renaming of any file or directory, creation of directories, or checking whether a path exists or not. File creation dates and attributes can also be changed if the attacker wishes. In addition to file system modification capabilities, the attacker(s) also have full control of the Windows registry, which can be read and modified as well. These components can also take ownership of any file, directory or registry key.

Most commands can also operate on network machines, either by using the Windows API functions with network paths (even if this requires an already authenticated connection to the target resource, an unsecured share or a share that the logged in user already has access to). However, if a username/password combination is required, this component has reimplemented the relevant parts of the Server Message Block (SMB) protocol that supports the pass-the-hash attack on NTLM authentication. Because of this, it can craft packets to authenticate to the remote machine without knowing the password and perform file or registry operations without passing a set of valid credentials.

Data theft operations

The highest risk that arises with the RadRAT is its powerful data theft capabilities. The main focus of these commands is credential theft, even if additional information such as browsing history and network traffic are also targeted.

Another command that performs NTLM hash harvesting from the Windows registry uses a significant chunk of the source code of the Mimikatz `lsadmp` tool. These harvested hashes can later be used for pass-the-hash attacks on this machine, or on another machine that uses the same credentials, since NTLM hashes are not salted.

An interesting command injects code into the lsass.exe process to decrypt the credentials storage under the `AppData\Microsoft\Credentials` and `LocalAppData\Microsoft\Credentials` directories. Stored credentials (Internet Explorer saved passwords for popular social media websites and domain visible passwords) and Wi-Fi SSIDs and passwords are also read and decrypted. Some of these commands perform Mimikatz-like credentials harvesting from WDigest.dll and kerberos.dll.

Also, a command is available that decrypts any input using the already logged in user's credentials. The C&C servers provide an encrypted blob which gets decrypted by the RAT via CryptUnprotectData. This blob could contain Chrome stored passwords uploaded by an earlier command, for instance.

In AppInit_DLL mode, hooks that intercept usernames and passwords entered into credential prompts given when connecting to network shares are installed into almost every new process. These results are written to a `Windows\temp\cfwe247` file, which is also sent to the server via one of the commands.

Browsing history is another highly sought-after piece of information: this component has the ability to gather and decode both Mozilla Firefox and Internet Explorer history (for the latter, the `defrag.exe` component is used).

The commands in this group also send network traffic that is captured via commands in the Network operations group.

Network operations

This group of operations focuses on network discovery, communication with other machines on the host and network traffic sniffing through ARP poisoning.

A set of commands attempts to find network machines and other resources, domain controllers using the WIndows API or find machines in a given IP range, using ARP and NetBIOS discovery. A command simply uploads the system's ARP cache to find the network hosts that were recently contacted. Two commands open tunnels from the C&C server either to a host on the network, on any port, or to a Windows named pipe on this system (or on a remote machine). Some commands attempt to detect network security measures by performing a DNS request or a HTTP request, since some organizations block certain domains at the DNS level. The attackers can also use the HTTP request command to access the organization's intranet, where sensitive information may be stored. In addition, one command attempts a Remote Desktop or SMB connection without any credentials, reporting a success or failure.

But the most powerful set of commands specialize in network traffic sniffing via ARP poisoning. The attackers can choose multiple IPs between which the connection will be intercepted. These components will then resolve their MAC addresses and create spoofed ARP packets to redirect all packets intended for these hosts to themselves. Afterwards, these components forward packets, logging all Remote Desktop and SMB sessions - possibly to intercept credentials - until requested to stop, or until 10 capture reads have failed with an error. Finally, the ARP poisoning stops and the network communication is reverted to the initial state.

Operations on processes

In addition to network traffic interception, this remote access trojan also features a set of commands to interact with processes. One command runs a process with arguments, while keeping the window invisible and optionally writing its outputs to a file, while waiting for a timeout.

A second command is able to load a DLL file into the current process, then call its `_run@0` export. These commands may be paired with a write file command, which allows the attackers to download and run a malicious file on the victim machine. This allows the malware to execute a executable file with system privileges and gain unfettered access to the device's resources. A third command can terminate any process, selected by its PID; this can be obtained from commands in the System information command group.

Operations on system information

Most of the RAT operations rely on particular intelligence on the targeted system. Because of this, RadRAT features a large number of commands that gather information on the victim machine.

A set of commands gather network information related to:

- **hosts on the network** that are currently connected via SMB or Remote Desktop Protocol (RDP) to this machine. Harvested information includes remote names for inbound and outbound connections and usernames of inbound connected clients, what shares and files are opened by these
- **all open TCP connections**. This includes source and destination IP address and ports, as well as the process that owns the connection
- **detailed information on Remote Desktop sessions** from clients to the current infected machine, whether those connections are active or not: includes client machine, username, domain and IP addresses
- **network adapter information.** This includes name, description, type, IP addresses with subnet masks, gateway IP addresses with subnet masks and adapter MAC address
- **default proxy servers** for HTTPS connections

A separate set of commands gather process information:

- **processes currently running on this machine**: includes PID, executable path, the user who started the process and their corresponding domain, process bitness (32-bit or 64-bit), executable file version and product version, description and company name from executable manifest and process start time
- **installed security solutions** registered with the Windows Management Instrumentation (WMI): information contains security solution names, paths, states and timestamps.
- **scheduled tasks on the system**: information contains task folders, task names, last run times and return codes, status, authors, paths, users, repeat settings, and more
- **visible windows**: process names, PIDs and window titles (this information is gathered with the help of the `defrag.exe` component)
- **loaded module paths inside a given process**

Other information harvested:

- **one or more screenshots**, at specific intervals (these commands are executed with the help of the `defrag.exe` component)
- **a backup of the Security event log**
- **time since last input event** (executed by the `defrag.exe` component)
- **version and bitness of the system's lsasrv.dll**, used for lsass.exe injection by data theft commands
- **KnownDlls** on this current machine
- **a vast amount of miscellaneous information**: this data includes drive letters with free space, system date and time, install date, system uptime, process owners (usernames with domains, whether they have administrative rights), mapped network drives, and screen resolution (this information can be useful in determining the importance of this machine, and possibly whether the system is a virtual machine, although there is no code that explicitly checks this).

Propagation operations

AS the subcategory title says, the purpose of these operations is to install RadRAT (either the `wrpcs.dll` or `ntmgr2.dll` components) to another host on the network. One command, exclusive to the `wrpcs.dll` component, installs the Sysmgr service and its `ntmgr2.dll` component to another machine and one command found only in the `ntmgr2.dll` component copies the wrpcs DLL to a host in the network and sets it as the DcomLaunch service.

A separate command progressively infects remote machines with the current component (`wrpcs.dll` inside the `DcomLaunch` service

or `ntmgr2.dll` inside the Sysmgr service). This command receives a "level" argument, which is the step of the infection to be checked:

- level -5 tries connecting to the Windows registry on that machine
- level -4 enables the `RemoteRegistry` service on the target machine
- level -3 opens the remote service manager on the machine
- level -2 closes the connection to the machine
- level -1 connects to the host with a username and password
- level 0 connects to the `ADMIN$` share on the machine
- level 1 connects to the `ADMIN$` share on the machine, checks whether the component is installed as a service and, if not, writes the service DLL, dependencies and the service executable (in the case of `ntmgr2.dll`'s `Sysmgr.exe`) and installs the service; finally, it disconnects from the host
- level 2 performs the same actions as level 1, but also updates the service if it is already present as an inferior version.

This command can work both with the Windows API methods and the reimplemented SMB layer.

Update operations

These commands are used by the malware to install newer versions of RadRAT's components on an infected machine. The main source of updated versions is Google Drive or Dropbox URLs. However, for one of the commands in this group, the C&C server sends newer versions of the RAT files, bypassing the original update source.

Two commands, each exclusive to one of the two main components, modify the current service DLL (the current component) on disk without restarting it. However, another command uses an additional tool (`~rs19.tmp` for `wrcps.dll`, `~rs.tmp` for `ntmgr2.dll`) to restart the current component with the new version: `~rs19.tmp` waits for every thread in the component to stop, then reloads the library, and `~rs.tmp` stops and restarts the Sysmgr service.

Another command is responsible for updating the component's own service files and re-adding it as a service. A separate command, that must be run while in the `sysmgr.exe` process, updates only this executable file. Finally, a command sends Google Drive or Dropbox download statistics, informing the attackers of the number of successful and unsuccessful download attempts.

Bookkeeping operations

This group of commands performs tasks that change the behavior of the main components.

A set of commands reads and writes parameters stored in the registry. For example, a change in the Det entry can trigger the AppInit_DLL behavior, injecting the `wrpcs.dll` component in every new process. A command changes the Sysmgr service display name, another command disables send buffering on the socket (at the kernel level). A command starts a new RAT command execution thread, and another command restarts the system, but there is also a set of commands that return a success code without performing any actions.

An interesting set of commands uses the victim machine to retrieve a Windows password from the LanMan (LM) hash by cracking sniffed NTLM authentication challenges. The server sends the first part of a sniffed LM authentication challenge (a pair of 8-byte hex strings, the sniffed challenge input and its output) and a range of keys to search (minimum and maximum key lengths and characters to try for each byte in the key). Then a thread is created for each challenge that tries to guess Windows passwords (first 7 characters), compute their LM hash, then try this LM hash against the challenge input and target output. When a match is found, it is saved and the thread exits. These challenges, along with hashing progress, are written to disk at `Windows\temp\~sdc978` and are resumed when the `wrpcs.dll` component starts.

However, since this hash cracking is CPU intensive, the attackers have implemented some measures to avoid detection. Firstly, these threads run at THREAD_PRIORITY_IDLE, which ensures that the threads will not cause noticeable slowdowns on a system and only use CPU cycles which are normally unused. Secondly, when the first LM authentication challenge request is given to this component, the Det registry parameter is set, which enables AppInit_DLLs injection.

When AppInit_DLLs injection is enabled, the `wrpcs.dll` component gets injected in almost every new process. Once loaded, this

component checks for the existence of the ~sdc978 file where challenge cracking requests are stored. If this file exists, some rootkit-like hooks will be installed.

A NtQuerySystemInformation hook intercepts the following information classes:

- SystemProcessInformation: the hook removes the kernel time, user time and cycle time changes from the `Sysmgr.exe` entry (hosting the `ntmgr2.dll` component, where challenges may be cracked) and adds them to the System Idle Process. Below is the pseudocode of the updating procedure in these structures:

```
for ( processInfo = argDestBuffer; ; processInfo = (processInfo + processInfo->NextEntryOffset) )
{
  if ( processInfo->UniqueProcessId )   // PID 0 is reserved for the System Idle Process
  {
    if ( processInfo->ImageName.Buffer
      && processInfo->ImageName.Length >= length_sysmgrExe
      && !_memicmp(processInfo->ImageName.Buffer, strW_sysmgrExe, length_sysmgrExe) )
    {                                   // this is the Sysmgr service process
      processInfo->UserTime = sysmgr_initialUserTime;
      processInfo->KernelTime = sysmgr_initialKernelTime;
      processInfo->CycleTime = sysmgr_initialCycleTime;
    }
  }
  else
  {                                     // this is the System Idle Process
                                        // idleProcess_delta variables contain
                                        // both the System Idle Process time and the Sysmgr time
    processInfo->UserTime.QuadPart = idleProcess_deltaUserTime + idleProcess_lastUserTime;
    idleProcess_lastUserTime = processInfo->UserTime.QuadPart;
    processInfo->KernelTime.QuadPart = idleProcess_deltaKernelTime + idleProcess_lastKernelTime;
    idleProcess_lastKernelTime = processInfo->KernelTime.QuadPart;
    processInfo->CycleTime.QuadPart = idleProcess_deltaCycleTime + idleProcess_lastCycleTime;
    idleProcess_lastCycleTime = processInfo->CycleTime.QuadPart;
  }
  if ( !processInfo->NextEntryOffset )
    break;
}
```

- SystemProcessorPreformanceInformation: the hook computes the kernel and user time for the `Sysmgr.exe` process and removes them from the CPU kernel and user time, making the processor appear idle.

Multiple hooks targeting completely undocumented Windows API functions used by Advapi32.dll to provide the Windows API performance counters feature (PcwCreateQuery, PcwAddQueryItem and PcwCollectData) induce a similar behavior.

These hooks change the behavior of functions used by task managers and resource monitors to hide CPU resources used by the NTLM challenge cracking operation.

Unimplemented operations

Finally, some operations are not implemented and subsequently, have no effect on the system whatsoever. Some of these operations feature large amounts of unused code, but this code is unconditionally skipped and empty results are sent to the C&C servers.

Automatic operations

These operations are independent of RAT commands and are performed automatically by RadRAT, although some of them may be controlled by setting a registry entry.

Automatic update and lateral movement

This is performed by the main RAT components, in a separate thread. First, if these components were not already installed on the system or if the installed component is an older version of the running component, an upgrade to the newer version is performed (in `rk.dat` mode, the upgrade is executed unconditionally). If, however, an attempted update had failed, a hook that retries the update with the new context's token is installed to secur32.dll's AcceptSecurityContext.

Separately from their own update mechanism, until the RAT stops for an update, these components try infecting remote hosts depending on the value of the Mt2 registry entry, every three hours:

- If the value is 0, no infection is attempted. This is the default value.
- if the value is 1, the main components enumerate network hosts, then try infecting them with the current component (wrcps.dll or `ntmgr2.dll`) using empty credentials.
- if the value is 2, the main components enumerate network hosts, including hosts for which credentials are stored and used automatically by Windows and infect them with the current component.
- if the value is 3, the main components enumerate network hosts and, only on 64-bit systems, hosts with stored credentials (as above) are collected; then, every 72 hours, if the Npd registry value is set (default behavior), the main components will try to find credentials stored in the Application Data directories (under the `AppData\Microsoft\Credentials` and `LocalAppData\Microsoft\Credentials` directories) which are decrypted using lsasrv.dll injections. These credentials, along with empty credentials, will then be used every 3 hours as mentioned above, to infect network hosts with the current component.

AppInit_DLLs hooks

Once the registry Det parameter is set, the `wrpcs.dll` component adds itself as Windows\`mwcpi.dll` enabling the AppInit_DLL mode, which loads itself into almost every new process (with very few exceptions).

When this component is loaded into a process as the `mwcpi.dll` component, it installs hooks that hide the CPU usage of NTLM authentication challenge cracking and harvests credentials from password prompts. The CPU usage hiding hooks are only installed when the `Windows\temp\~sdc978` file exists, which means that there are pending cracking requests, and, subsequently, cracking threads consuming processor resources. These hooks have been explained in the **Bookkeeping operations** section.

Credentials harvesting is done by hooking:

- **Credui.dll's CredUIPromptForCredentialsW and CredUIPromptForWindowsCredentialsW**: log the process that requested the credentials, the prompted text (caption and message) and entered username and password. These functions are called when an application requests a native credentials prompt.
- **mpr.dll's WNetUseConnectionW**: log the resource name, username and password used to connect to that resource. This function is called either directly when an application requests to connect to a machine on the network (using Windows credentials) or indirectly by WNetAddConnection, WNetAddConnection2, WNetAddConnection3 (both ANSI and Unicode variants), or WNetUseConnectionA. Notably, under Windows 10 (but not under Windows XP), this function is not called by WNetAddConnection2W (only the Unicode version). Presuming that WNetAddConnection2W is often used, some applications are protected. Logs are written to the Windows\temp\cfwe247 file, which will be uploaded to the C&C servers with the data exfiltration RAT commands.

Conclusion

This deep dive into the RAT's components reveals an extremely complex attack toolkit that is particularly optimized for networked environments such as enterprises or large businesses running Windows. Its complexity allowed it to run undetected for years, away from prying eyes, potentially causing irreparable damage to affected companies. And, despite not having access to the victim's profile and the pool of information exfiltrated, the complexity and wide range of capabilities can only hint at an extremely successful cyber-espionage toolkit that has carried it mission to completion successfully.

Appendix 1: RadRAT samples

Main components:

SHA-256	type	version	notes
4786fa468111632ea66f03dfd868ca95fb91d4472b2c332d46d8444c19c75624	`wrpcs.dll` 32-bit	234	marker `CD AB CE BB`; this sample
229666558fb0f45d00ea2f91bf0de26afdf3b5c98a8f4613b70ca14c31af6772	`ntmgr2.dll` 32-bit	215	marker CD AB CE AB; `ntmgr2.dll` component analyzed along this sample; downloaded from update URLs
1c31f34913aae2399cc9bbd55466e90ec1ab490973e113cbfba9505807b19c8b	`ntmgr2.dll` 64-bit	215	marker CD AB CE AB; downloaded from update URLs
fa3d342a1d1500cea18bb8ff27bcd7cde947b72d7ae4fbcb044c7c6fc090b701	wrcps.dll 32-bit	228	marker CD AB CE AB
cf7aa86f6f83fdd77b70ca0d86c3937006f59a984faa1c375b89960341261efb	`wrpcs.dll` 32-bit	228	marker CD AB CE AB

Secondary tools:

These are secondary tools, downloaded from update URLs (Google Drive/Dropbox)

SHA-256	type
d390f6ea7513b9d7277c0c958e5c6276d1cccefd3e40aec894c0253faa199693	`~rs19.tmp` 32-bit
987c88f579317057ffb7ac446ece0d2fc2a8457de82aae0893c447c6bc34740e	`~rs19.tmp` 64-bit
ee15ac659659504f42e55307ac1a63db3e0705ed916ef69a646af1adf0cd0b23	`~rs.tmp` 32-bit
00ca4e3df5d9a8f7e1b6afb7c13c9ca68744cb41f4386d061c68ab3dfeacf8f0	`sysmgr.exe` 32-bit
cf4b21e9170d1c398109c9e149dc7222688b2f77334ca745689b117787f729ef	`sysmgr.exe` 64-bit
97814cca9593faedc3d226f532ad21028aeb6d3c205996f777f19e0a6ade8d7c	`defrag.exe` 32-bit

Test files:

These are, apparently, files used by threat developers in testing that have found their way on public sample collections like VirusTotal.

SHA-256
3960c8633478d3c12dd9ddcfe122a55c622969369ed65aea781f14c22d0121b0
d972bb4206c316f8984979cf4b0bfa81cab12160bf90460c45631f052473a0a5
365f93c61f08133cf3d6f5ca2924bab28b600797119bfa169f437d9031881af6
044c277771f47453641b96e60af6d195827920113a373ad602601e3d9eecd85a
fccf57019ca2e20c1ad02eedc7536b667e4d42668115fc79bd6ed3ce9cd05618
bc061acd1dcb7c34926ee789c08dcac206f3c4e67a1f2871fddd2c9b8158b1e3
d729d642f0aa67cd9e10ce1552dbda55ceef6a1d6040f0a76e57e515ec831ccf
d4706974f474259ccb5d21924382e2be59941013a55aa04d8cb2ed4240067af4
cb48516d776eae64bc6c53647567d3a92576626fa8897a9136260f1e0790d26b
580c32929187495df2aeb233ae2d6f782d51981a13bc6999dc329bca00e3d62c

Appendix 2: Google Drive/Dropbox update URLs

Dropbox:

Component	URL
`wrpcs.dll`	https://dl.dropboxusercontent.com/s/irci38y1l6pecy2/d4c2?dl=1
`wrpcs.dll (64-bit)`	https://dl.dropboxusercontent.com/s/jfejbkk78qf7agc/d4c264?dl=1
`ntmgr2.dll`	https://dl.dropboxusercontent.com/s/mh95qzfiqxaw993/pkc2?dl=1
`ntmgr2.dll (64-bit)`	https://dl.dropboxusercontent.com/s/tryqoe2jcq36hw4/pkc264?dl=1
`sysmgr.exe`	https://dl.dropboxusercontent.com/s/s707c5dju771neq/sysc2?dl=1
`sysmgr.exe (64-bit)`	https://dl.dropboxusercontent.com/s/dwx6hcg6lpcaqwo/sysc264?dl=1
`~rs19.tmp`	https://dl.dropboxusercontent.com/s/mroar2mb0car404/drsc2?dl=1
`~rs19.tmp (64-bit)`	https://dl.dropboxusercontent.com/s/ayf1mxtttf27ctg/drsc264?dl=1
ssleay.dll	https://dl.dropboxusercontent.com/s/bgkm8lw063z1akm/sslc2?dl=1
ssleay.dll (64-bit)	https://dl.dropboxusercontent.com/s/h1lw7hqpv9etfrd/sslc264?dl=1
libeay.dll	https://dl.dropboxusercontent.com/s/va194n8k5zhs470/libc2?dl=1
libeay.dll (64-bit)	https://dl.dropboxusercontent.com/s/jzexisn4ydofpmh/libc264?dl=1
npf.sys (WinPcap)	https://dl.dropboxusercontent.com/s/9zfo2pqddggsnmo/np.f?dl=1
NPPTools.DLL (WinPcap)	https://dl.dropboxusercontent.com/s/pjb0bkeky4b94q4/npptool.s?dl=1
wpcap.dll (WinPcap)	https://dl.dropboxusercontent.com/s/bf5vlzjom181305/wpc0?dl=1
packet.dll (WinPcap)	https://dl.dropboxusercontent.com/s/ct3t2ccnq6qf1ca/pack0?dl=1
C&C IP address	https://dl.dropboxusercontent.com/s/xrkpsgq154qx8ko/pi0?dl=1

Google Drive:

Component	URL
`wrpcs.dll`	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaSllkTDRmenRwbHM
`wrpcs.dll` (64-bit)	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaOHduY3ZnQlFLTE0
`ntmgr2.dll`	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvablNpaE5JZVJ0eWc
`ntmgr2.dll` (64-bit)	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaT0U1Tl8tVG1lMU0
`sysmgr.exe`	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvabnI0NzIxODdqQk0
`sysmgr.exe` (64-bit)	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaSEZXZEhsOG1TVDA
`~rs19.tmp`	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaemlsbTlDOWxsZ0k
`~rs19.tmp` (64-bit)	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaYXVrdkJDcjc0R28
ssleay.dll	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaSGJpWlM5YV81bE0
ssleay.dll (64-bit)	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaVTNkNEpWdUdIM00
libeay.dll	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaeVNKNFFqalNQM0E
libeay.dll (64-bit)	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaNEt3VTNTMHFya3M
npf.sys (WinPcap)	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaV2lVcDM2Y1hweE0
NPPTools.DLL (WinPcap)	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvacHgtbHNRbGhvNGM
wpcap.dll (WinPcap)	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaR3dsTnY0b2I4eWM
packet.dll (WinPcap)	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaVThraUV2NzJldjA
C&C IP address	https://drive.google.com/uc?export=download&id=0BxijCDz_mIvaelFTajB2Umw0b0k

Appendix 3: Other Indicators of Compromise

For more indicators of compromise, see Appendix 1.

Type	Indicator of Compromise
domain	msn.cechire[.]com
domain	msn.dyndns[.]tv
file mapping	msrpcnm
registry path	HKLM\Software\Microsoft\Rad3Dev\Dev1
filename	Windows\System32\`wrpcs.dll`
filename	Windows\System32\`ntmgr2.dll`
filename	Windows\System32\`sysmgr.exe`
filename	Windows\`mwcpi.dll`
filename	`defrag.exe` in Temp directory
filename	Windows\temp\~adre379
filename	Windows\temp\cfwe247
filename	Windows\temp\~sdc978

Appendix 4: RAT command codes and descriptions

File/registry operations:

command ID	description
1	list shares on machine or files in directory (can use pass-the-hash)
2	upload file (from machine to C&C)
3	download file (from C&C to machine)
4	delete file (can use pass-the-hash) or delete directories
5	rename file (can use pass-the-hash)
6	copy files or directories; can use a raw copy system and/or delete sources after
7	check if file exists and can read it (can use pass-the-hash)
27	read and hash chunks of files, compare them to given hashes, upload if no matches
28	read and hash chunks of files and send hashes
29	write to chunks of files (can also conserve filetimes)
30	create directory with name
39	enumerate and send registry path, local or remote, in a format similar to Regedit's .REG
71	set filetimes for file (can use pass-the-hash)
73	send directory size
75	send whether command 6's last copy finished successfully
80	take ownership of file or registry key
81	set file attributes (can use pass-the-hash)
91	set a registry key under HKEY_LOCAL_MACHINE

Data stealing:

command ID	description
9	send keylogger logs (keylogger behavior is currently disabled)
10	gather and send credentials (from many sources), Firefox history and captured network traffic
21	initiate credentials gathering from lsass.exe processes and modules (lsasrv.dll, wdigest.dll, kerberos.dll)
47	send subset of credentials (from the Windows credential storage)
48	send names for credentials in Windows credential storage
50	send subset of credentials (from AppData or LocalAppData directories)
51	(only on `ntmgr2.dll`) older version of lsasrv.dll/wdigest.dll/kerberos.dll decryption
55	send saved WLAN information, including decrypted credentials
56	send user information, along with decrypted NT & LM password hashes (useful for pass-the-hash)
57	used in lsasrv.dll/wdigest.dll/kerberos.dll decryption

63	restore lsasrv.dll/wdigest.dll/kerberos.dll after decryption
68	get IE history and IE saved passwords for some websites (popular social media websites)
69	decrypt blobs using user credentials (using CryptUnprotectData)
88	send lsasrv.dll/wdigest.dll/kerberos.dll decryption results

Network operations:

command ID	description
12	connect to another host and create tunnel from the C&C server to and from the another host
18	enumerate network resources and get domain controllers
19	enumerate machines in a domain and send domain, host and IP address
31	gather information about devices that WinPcap can capture on
32	use ARP discovery to identify machines on network, try to resolve IPs to NetBIOS hostnames
34	packet-level sniffing using ARP spoofing, targeting RDP and SMB connections
33	stop MITM thread and send parameters and results
35	stop MITM thread and send success
58	send ARP table
61	discover NetBIOS hosts inside an IP range
78	make an HTTP request
82	open tunnel between C&C and a named pipe on the local or a remote machine
86	resolve domain or parse IP
92	attempt SMB or RDP connection (without credentials)

Process operations:

command ID	description
8	run a process, optionally capture stdout/stderr and exit code
74	run a DLL's "_run@0" exported function
83	terminate process by its PID

Propagation operations:

command ID	description
20	progressively (with separate commands, at different levels), try infecting remote machines with own (`wrpcs.dll` or `ntmgr2.dll`) service or DLL
49	(only on `ntmgr2.dll`) infect a host with the `wrpcs.dll` component (inside DcomLaunch)
79	(only on `wrpcs.dll`) infect a host with Sysmgr service (running `ntmgr2.dll`)

Update operations:

command ID	description
13	(only on `wrpcs.dll`) update current DLL
14	(only on `ntmgr2.dll`) update current DLL
16	update own service DLLl (`wrpcs.dll` or `ntmgr2.dll`) and re-add as a service
23	on `wrpcs.dll`: reload own DLL (download a DLL (drsc2), start thread in DLL, stop RAT behavior, wait for threads to stop, run system32/`wrpcs.dll`); on `ntmgr2.dll`: restart Sysmgr service (using the `~rs.tmp` tool)
62	update `sysmgr.exe` (must be run from `sysmgr.exe`)
89	save binaries for some components manually (bypassing Drive/Dropbox, will be used in later updates)
90	send Drive/Dropbox download statistics

System information operations:

command ID	description
11	take and send screenshot
40	get network information (hosts in network, SMB/RDP connections to this machine, TCPView-type information)
42	take multiple screenshots at a certain time interval (using `defrag.exe` tool)
43	send the result of an earlier command 42 (take multiple screenshots)
46	send security solutions registered with Windows
52	send information on Remote Desktop sessions on this machine
53	send information on IP adapters on this machine
54	send detailed information on processes on this machine
59	send scheduled tasks
60	backup Security event log to file and send filename (to be uploaded later)
67	send time since last input event
70	send miscellaneous information
76	send visible window process names, PIDs and window titles
77	send default proxy settings
84	send KnownDLLs list
85	send modules of a process by PID
87	send version and bitness (32 or 64 bit) of Windows\System32\lsasrv.dll

Bookkeeping operations:

command ID	description
15	start new RAT thread
22	(only on `ntmgr2.dll`) disable send buffering on socket (at kernel level)
24	set registry parameter, re-read parameters, send final version (after applying min/max constraints)

25	send registry parameter
36	send logical processor count & NTLM cracking requests and progress (persisted to disk)
37	add a new entry to NTLM cracking requests and trigger AppInitDlls injection
38	wait for a NTLM cracking request entry to exit and remove it
65	change Sysmgr service display name
66	restart system
72	set field in C&C communication structure
26	return from RAT function (will reconnect)
44	return from RAT function (will reconnect)
64	return from RAT function (will reconnect)

Unimplemented operations:

command ID	**description**
17	read 1 byte, do nothing
41	not fully implemented, send empty results
45	not fully implemented, send empty results

Bitdefender is a global security technology company that delivers solutions in more than 100 countries through a network of value-added alliances, distributors and reseller partners. Since 2001, Bitdefender has consistently produced award-winning business and consumer security technology, and is a leading security provider in virtualization and cloud technologies. Through R&D, alliances and partnership teams, Bitdefender has elevated the highest standards of security excellence in both its number-one-ranked technology and its strategic alliances with the world's leading virtualization and cloud technology providers. More information is available at http://www.bitdefender.com/

FOR MORE INFORMATION VISIT: enterprise.bitdefender.com

BD-Business-Apr.12.2018-Tk#: crea2645
Bitdefender-Whitepaper-RADrait-crea2645-A4-en_EN