Microsoft

Microsoft Defender Advanced Threat Protection

Tutorial

Live response

May 2019

Copyright

Our detection philosophy

It's simple.

We make sure that known advanced persistent threat (APT) indicators or techniques are visible in our telemetry, that we recognize them and we are able to raise the relevant alerts.

When we raise an alert near real-time, we provide the relevant context, including actor attribution, their victimology, geo-affinity, and main tactics. This is realized through a rich, dynamic library of known attack indicators, including known threat components previously observed on real machines, script and web page snippets from compromised or malicious websites, as well as IPs, URLs, and domains representing the attacker's infrastructure. This library is constantly updated with new threat intelligence generated mainly by Microsoft's own APT hunting and research teams, but enriched by collaboration with partners and shared feeds.

Because threats are constantly being crafted and modified, we monitor a large set of anomalous and suspicious behaviors to find new and unknown actor activity. These anomalous and suspicious activities raise alerts for the Security Operations Center (SOC) analyst to validate and address. With the help of information about proximate events on the same machine and other relevant machines, SOC analysts can validate actual breach activity, determine risk, establish the scope of the breach, define containment activities, and then contain, mitigate and fully respond to the attack.

Introduction to the live response tutorial

Live response is a capability that gives you instantaneous access to a machine using a remote shell connection. This gives you the power to do in-depth investigative work and take immediate response actions to promptly contain identified threats – real-time.

Live response is designed to enhance investigations by enabling you to collect forensic data, run scripts, send suspicious entities for analysis, remediate threats, and proactively hunt for emerging threats.

Live response allows to run four types of commands:

1. Run basic and advanced commands to do investigative work.

2. Download files such as malware samples and outcomes of PowerShell scripts.

3. Run remediation / undo remediation commands.

4. Upload a PowerShell script to a library and run it on the machine from a tenant level.

In this tutorial, you'll be guided the following scenarios:

- Initiating a live response session an perform basic remediation
- Use a PowerShell script to get a mini memory dump
- Use forensic tools to gather forensic information

Live response allows you to run commands to help you look for common indicators of compromise. As you are guided through the tutorial, you'll be shown how to locate a suspicious file which you can then remediate using the available live response commands.

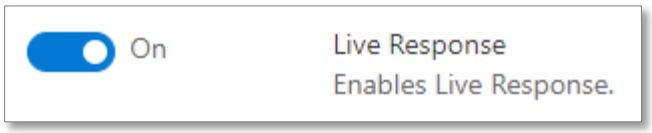
The tutorial will also guide you through the concept of using the library where you can upload scripts. You'll be shown how to upload the script the library and run the script to get a mini memory dump which you can then analyze.

Finally, the tutorial will demonstrate you can use forensic tools to gather forensic information.

Before you begin

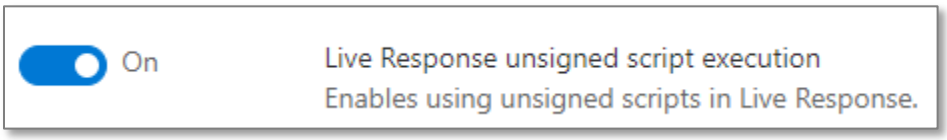
Before you can initiate a session on a machine, make sure you fulfill the following requirements:

- Machines must be Windows 10, version 18323 (also known as Windows 10 19H1) or later.
- You must complete the DIY scenario 1 from the Microsoft Defender ATP tutorials.
- **Enable live response from the settings page**
  You'll need to enable the live response capability in the Advanced features settings page.



  Note: Only users with manage security or global admin roles can edit these settings.

- **Enable live response unsigned script execution (optional)**
  If you plan to use an unsigned script in the session, you'll also need to enable the setting in the Advanced features settings page.



- **Ensure that you have the appropriate permissions**
  Only users who have been provisioned with the appropriate permissions can initiate a session. For more information on role assignments see Create roles.

Depending on the role that's been granted to you, you can run basic or advanced live response commands. Users permission are controlled by RBAC custom role.

Initiate a live response session and perform basic remediation

1. Log in to the Microsoft Defender Security Center.
2. Navigate to Machines list page.
3. Select a compromised machine to open the machine page.
4. Launch the live response session by clicking Initiate Live response session.

   ▷ Initiate Live Response Session

5. Wait while the session connects to the machine.

   Command console Command log

   ```
   C:\> connect
   Sense IR is running and registered
   ```

6. When the machine is connected, type *help* in the command console. It will show you all the available commands.

   ```
   C:\> help
   For more information on a specific command, type HELP command-name

   analyze         Analyzes the entity for threats and returns a verdict (malicious, clean, suspicious)
   cd              Changes the current folder
   cls             Clears the console screen
   connect         Establishes connection with the machine for the live response session
   connections     Shows all active connections
   dir             Shows the list of files and sub-folders in a folder
   drivers         Shows all drivers installed on the machine
   fileinfo        Shows information about a file
   findfile        Locates files with a given name on the machine
   getfile         Downloads a file from the machine
   help            Shows information about live response commands
   library         Lists or takes action on files in the live response library
   persistence     Shows all known persistence methods on the machine
   processes       Shows all processes running on the machine
   putfile         Uploads a file from the library to the machine
   registry        Shows information about specific keys or values in the registry
   remediate       Remediates an entity on the machine
   run             Runs a PowerShell script from the library on the machine
   scheduledtasks  Shows all scheduled tasks on the machine
   services        Shows all the services on the machinee
   trace           Sets logging on this console to debug mode
   undo            Restores an entity that was remediated
   ```

7. Look for other common indications of compromise by running the other commands such as:

- *connections* to see current connections

```
C:\users> connections
Name                    Pid     Process Name            Local Ip    Local Port  Remote Ip   Remote Port  Status
=====================   =====   =====================   =========   ==========  =========   ===========  ======
sshd.exe                2852    sshd.exe                0.0.0.0     22          0.0.0.0                  LISTEN
svchost.exe             1372    svchost.exe             0.0.0.0     135         0.0.0.0                  LISTEN
LMS.exe                 10356   LMS.exe                 0.0.0.0     623         0.0.0.0                  LISTEN
vmms.exe                5200    vmms.exe                0.0.0.0     2179        0.0.0.0                  LISTEN
sshd                    18880   sshd                    0.0.0.0     2222        0.0.0.0                  LISTEN
svchost.exe             1708    svchost.exe             0.0.0.0     3389        0.0.0.0                  LISTEN
svchost.exe             3792    svchost.exe             0.0.0.0     5040        0.0.0.0                  LISTEN
svchost.exe             7360    svchost.exe             0.0.0.0     7680        0.0.0.0                  LISTEN
MouseWithoutBorders.exe 7404    MouseWithoutBorders.exe 0.0.0.0     15100       0.0.0.0                  LISTEN
MouseWithoutBorders.exe 7404    MouseWithoutBorders.exe 0.0.0.0     15101       0.0.0.0                  LISTEN
LMS.exe                 10356   LMS.exe                 0.0.0.0     16992       0.0.0.0                  LISTEN
lsass.exe               688     lsass.exe               0.0.0.0     49664       0.0.0.0                  LISTEN
```

- *processes* to see current processes

```
C:\users> processes
Name                                                        PID   Status     User Name                                    Cpu Cycles (K)  Memory (K)
===========================================================  ====  =========  ===========================================  ==============  ==========
Secure System                                               72    Suspended  NT AUTHORITY\SYSTEM
Registry                                                    128   Running    NT AUTHORITY\SYSTEM
smss.exe                                                    788   Running    NT AUTHORITY\SYSTEM
csrss.exe                                                   896   Running    NT AUTHORITY\SYSTEM
wininit.exe                                                 980   Running    NT AUTHORITY\SYSTEM
csrss.exe                                                   988   Running    NT AUTHORITY\SYSTEM
services.exe                                                468   Running    NT AUTHORITY\SYSTEM
LsaIso.exe                                                  820   Running    NT AUTHORITY\SYSTEM                                697982          1272
lsass.exe                                                   688   Running    NT AUTHORITY\SYSTEM                                147679660       16752
winlogon.exe                                                1048  Running    NT AUTHORITY\SYSTEM                                900128          2780
```

- *drivers* to see current drivers

```
C:\users> drivers
Path                                                        Driver Loaded  Service Name                      Service State   Service Type
==========================================================  =============  ================================  =============   ============================
c:\windows\system32\drivers\1394ohci.sys                                   1394ohci                          SERVICE_STOPPED SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\3ware.sys                                      3ware                             SERVICE_STOPPED SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\acpi.sys                        true           ACPI                              SERVICE_RUNNING SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\acpidev.sys                                    AcpiDev                           SERVICE_STOPPED SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\acpiex.sys                      true           acpiex                            SERVICE_RUNNING SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\acpipagr.sys                                   acpipagr                          SERVICE_STOPPED SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\acpipmi.sys                                    AcpiPmi                           SERVICE_STOPPED SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\acpitime.sys                                   acpitime                          SERVICE_STOPPED SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\acx01000.sys                                   Acx01000                          SERVICE_STOPPED SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\acxhdaudio.sys                                 AcxHdAudio                        SERVICE_STOPPED SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\adp80xx.sys                                    ADP80XX                           SERVICE_STOPPED SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\afd.sys                         true           AFD                               SERVICE_RUNNING SERVICE_KERNEL_DRIVER
c:\windows\system32\drivers\afunix.sys                      true           afunix                            SERVICE_RUNNING SERVICE_KERNEL_DRIVER
```

- *services -output json* to see current services in json format

```
C:\users> services -output json
{
  "service_start_name": "NT Authority\LocalService",
  "service_flags": "SERVICE_RUNS_IN_NON_SYSTEM_PROCESS",
  "display_name": "Agent Activation Runtime",
  "start_type": "SERVICE_DEMAND_START",
  "service_name": "AarSvc",
  "args": "-k AarSvcGroup -p",
  "current_state": "SERVICE_STOPPED",
  "tag_id": 0,
  "get_service_status_hresult": 0,
  "process_id": 0,
  "load_order_group": "",
  "dependencies": "",
  "get_service_config_hresult": 0,
  "service_type": "SERVICE_WIN32_SHARE_PROCESS",
  "path": "c:\windows\system32\svchost.exe",
  "binary_path": "c:\windows\system32\svchost.exe -k aarsvcgroup -p"
}

{
  "service_start_name": "NT AUTHORITY\LocalService",
  "service_flags": "SERVICE_RUNS_IN_NON_SYSTEM_PROCESS",
```

- *scheduledtasks* to see current scheduled tasks

```
C:\users> scheduledtasks
Path                                                                                                  Display Name                                                                  Category  Is Enabled
====================================================================================================  ============================================================================  ========  ==========
C:\Program Files (x86)\Azure Information Protection Updater\MSIP.Tools.DogfoodClient.exe              AzIPClientDogfoodTask update                                                  Exec      true
C:\WINDOWS\SYSTEM32\WindowsPowerShell\v1.0\PowerShell.exe                                             CSEO-Backup-BitLockerKey                                                      Exec      true
C:\Program Files (x86)\Google\Update\GoogleUpdate.exe                                                 GoogleUpdateTaskMachineCore                                                   Exec      true
C:\Program Files (x86)\Google\Update\GoogleUpdate.exe                                                 GoogleUpdateTaskMachineUA                                                     Exec      true
"C:\Program Files\Intel\Intel(R) Management Engine Components\iCLS\Intel PTT EK Recertification.exe"  Intel PTT EK Recertification                                                     Exec      true
C:\Program Files (x86)\Microsoft\EdgeUpdate\MicrosoftEdgeUpdate.exe                                   MicrosoftEdgeUpdateTaskMachineCore                                            Exec      true
C:\Program Files (x86)\Microsoft\EdgeUpdate\MicrosoftEdgeUpdate.exe                                   MicrosoftEdgeUpdateTaskMachineUA                                              Exec      true
%localappdata%\Microsoft\OneDrive\OneDriveStandaloneUpdater.exe                                       OneDrive Standalone Update Task w2                                            Exec      true
%localappdata%\Microsoft\OneDrive\OneDriveStandaloneUpdater.exe                                       OneDrive Standalone Update Task-S-1-5-21-72051607-1745760036-109187956-268037 Exec      true
%localappdata%\Microsoft\OneDrive\OneDriveStandaloneUpdater.exe                                       OneDrive Standalone Update Task-S-1-5-21-729554484-2162643427-3054924103-500  Exec      true
```

- *dir* to see current files in directory

```
C:\users> dir
Path                  Created              Modified             Size   Is Directory   Read Only      Hidden
====================  ===================  ===================  ====   ============   =========      ======
C:\users\..           2019-03-06 02:57:44  2019-03-06 21:13:52          true           true
C:\users\.            2019-03-06 02:57:44  2019-03-06 21:13:52          true           true
C:\users\[redacted]   2019-03-06 21:13:52  2019-04-04 13:47:41          true
C:\users\desktop.ini  2019-03-06 03:22:29  2019-03-06 03:22:29   174                                  true
C:\users\Default      2019-03-06 02:57:44  2019-03-07 16:32:44          true           true           true
C:\users\Public       2019-03-06 03:28:23  2019-04-03 12:44:57          true           true
C:\users\Administrator 2019-03-06 21:13:44 2019-03-07 16:23:47          true
```

8. Locate suspicious file using *findfile* command.

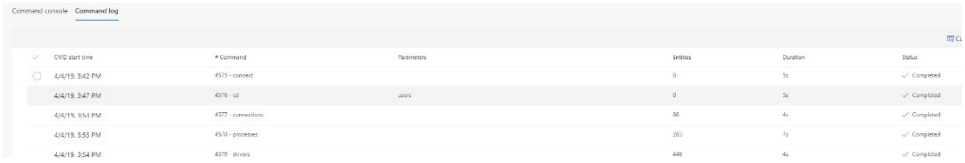
```
C:\> findfile trojan.zip
{
  "paths": [
    "c:\Users\Public\trojan.zip"
  ]
}
```

Note: The results from this command might take a while to show.

9. Run the remediate command on the file: *remediate file <filepath> -auto*.

```
C:\> remediate file "c:\Users\Public\trojan.zip" -auto
{
  "quarantine_guid": "800392AD-0000-0000-5447-53084BA9D4D6",
  "resources_info": [
    {
      "scheme": "file",
      "hresult": null,
      "key": "c:\users\public\trojan.zip"
    }
  ],
  "error_hresult": 0,
  "error_message": "",
  "resources_failed_remediation": [],
  "error_description": "",
  "remediation_status": "Quarantined",
  "remediation_state": "Finished"
}
```

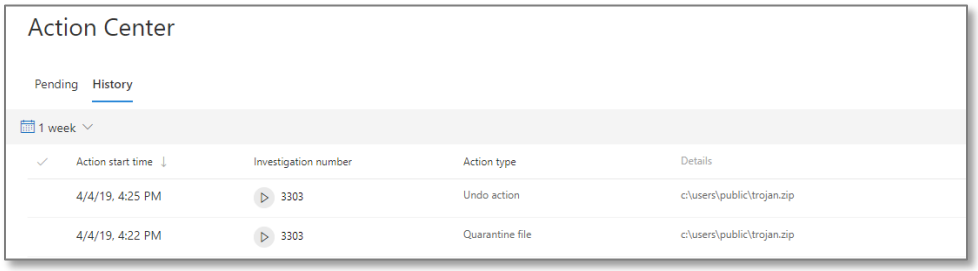
10. Confirm that all actions are logged in the Command log.



11. Undo the file remediation by typing the following command: undo file <filename>

```
C:\> undo file "c:\Users\Public\trojan.zip"
{
  "RemediationResult": {
    "quarantine_guid": null,
    "resources_info": [
      {
        "scheme": "file",
        "hresult": null,
        "key": "c:\users\public\trojan.zip"
      }
    ],
    "error_hresult": 0,
```

12. Alternatively, we can undo remediation from the **Action Center** > **History tab** (it is takes ~5 minutes to appear).



Congrats – you're initiated a live response session and performed basic remediation!

Next, let's learn how you can use a PowerShell script to get a mini memory dump.

Get a mini memory dump using a PowerShell script

Before you can run a PowerShell script, you must first upload it to the library. If you plan to use an unsigned script in the session, you'll need to enable the setting in the Advanced features settings.

1. Create PowerShell script with the following content:

```
$process_arg=$args[0]

function MiniDumpWriteDump
{
    [CmdletBinding()]
    Param (
        [Parameter(Position = 0, Mandatory = $True, ValueFromPipeline = $True)]
        [System.Diagnostics.Process]
        $Process
    )
    BEGIN
    {
        $WER =
[PSObject].Assembly.GetType('System.Management.Automation.WindowsErrorReporting')
        $WERNativeMethods = $WER.GetNestedType('NativeMethods', 'NonPublic')
        $Flags = [Reflection.BindingFlags] 'NonPublic, Static'
        $MiniDumpWriteDump = $WERNativeMethods.GetMethod('MiniDumpWriteDump', $Flags)
        $MiniDumpWithFullMemory = [UInt32] 2
    }
    PROCESS
    {
            # get the process dump
            $ProcessId = $Process.Id
            $ProcessName = $Process.Name
            $ProcessHandle = $Process.Handle
            $ProcessFileName = "$($ProcessName)_$($ProcessId).dmp"
            $ProcessDumpPath = $env:TEMP + "\$([IO.Path]::GetRandomFileName())_" +
$ProcessFileName
            $FileStream = New-Object IO.FileStream($ProcessDumpPath,
[IO.FileMode]::Create)
            $Result = $MiniDumpWriteDump.Invoke($null, @($ProcessHandle, $ProcessId,
$FileStream.SafeFileHandle, $MiniDumpWithFullMemory, [IntPtr]::Zero, [IntPtr]::Zero,
[IntPtr]::Zero))
            $FileStream.Close()

            if (-not $Result)
            {
                    # Remove any partially written dump files. For example, a partial
dump will be written
                    # in the case when 32-bit PowerShell tries to dump a 64-bit
process.
                    $Exception = New-Object ComponentModel.Win32Exception
                    $ExceptionMessage = "$($Exception.Message)
($($ProcessName):$($ProcessId))"
                    Remove-Item $ProcessDumpPath -ErrorAction SilentlyContinue
                    throw $ExceptionMessage
            }
            # Compress to ZIP
            $OutputFilePathZip = "$($ProcessDumpPath).zip"
            Compress-Archive -Path $ProcessDumpPath -DestinationPath
$OutputFilePathZip
            Remove-Item -Path $ProcessDumpPath -ErrorAction SilentlyContinue
            # write path to file and size
            Write $OutputFilePathZip
            Write "$([int]((Get-Item $OutputFilePathZip).length / 1024 / 1024)) MB"
    }

    END {}

}


try {
       # by pid
    $process_id = [convert]::ToInt32($process_arg, 10)
    Get-Process -Id $process_id | MiniDumpWriteDump
}
catch {
       # by name
    Get-Process $process_arg | MiniDumpWriteDump
}
```

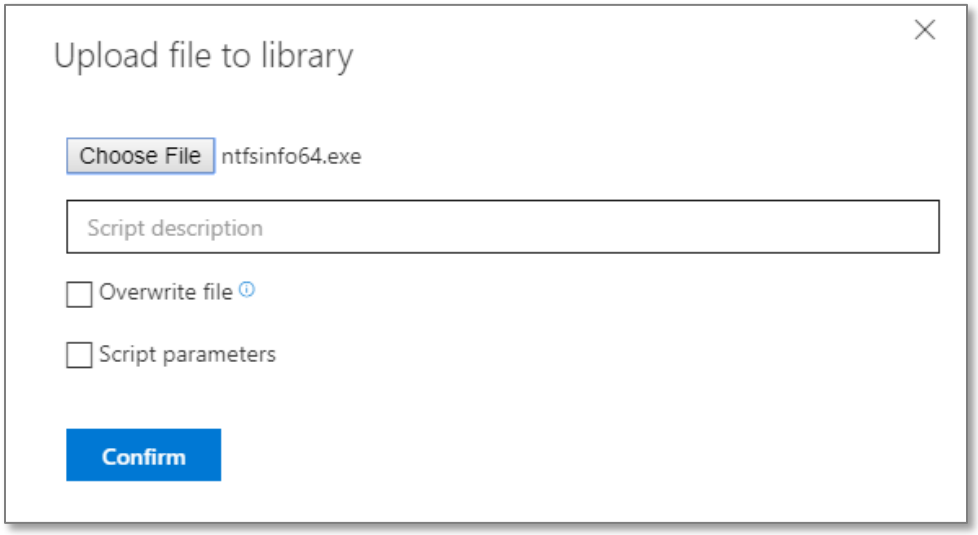
2. From the Live Response dashboard, upload the PowerShell script to the library by clicking **Upload file to library**.

3. Type ***library*** in the command console to see the list of available scripts.
4. Execute the script using the following command: *run <script name> <process name>* or <PID>
5. File is saved in C:\WINDOWS\TEMP\filname.dmp.zip
6. ***Download*** the recently created "mini memory dump" by typing: download <filepath> -autorun
7. After a few seconds you should see the file being downloaded in your browser. Depending on your browser settings, it allows you to save the file automatically or asks where you'd like to save the file for further analysis.

Use forensic tools to gather forensic information

This section will demonstrate how you can use any forensic tools such as Sysinternals to gather forensic information. In our example we will demonstrate how to dump a Master File Table (MFT) using ntfsinfo64.exe.

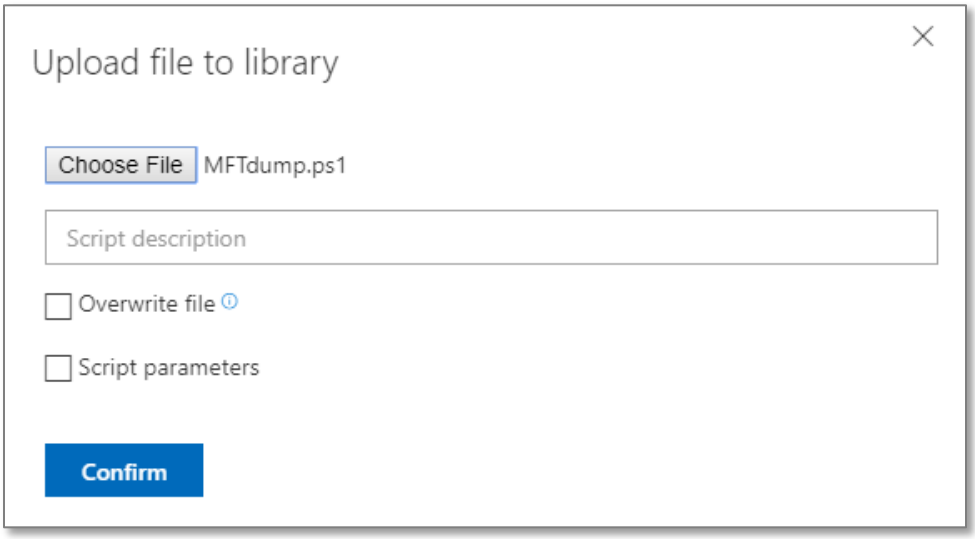
1. Download *ntfsinfo64.exe* from the NTFSInfo page.
2. Extract the downloaded zip file.
3. Upload *ntfsinfo64.exe* to your live response library.



4. Open Notepad. Copy and paste the following script:

   .\ntfsinfo64.exe /accepteula c:

5. Save the script file as *MFTdump.ps1*.
6. Upload this file to your library



7. Run *library* command to see the newly added files

```
ntfsinfo64.exe                                    No        Thu Apr 04 2019 19:52:11 GMT+0300 (Israel Daylight Time)
MFTdump.ps1                                       No        Thu Apr 04 2019 19:54:01 GMT+0300 (Israel Daylight Time)
```

8. Run *putfile ntfsinfo64.exe*
   This command will download the file from live response library to the *"C:\ProgramData\Microsoft\Windows Defender Advanced Threat Protection\Downloads\"* local machine directory on the machine you are connected to

```
C:\> putfile ntfsinfo64.exe
The file was uploaded to the machine.
Path: C:\ProgramData\Microsoft\Windows Defender Advanced Threat Protection\Downloads\ntfsinfo64.exe
```

9. Type *run MFTdump.ps1* in the live response CLI. This command will run the PowerShell script on the machine.

```
C:\> run MFTdump.ps1
Transcript started, output file is C:\ProgramData\Microso
575AE8BDE}.txt

Volume Size
-----------
Volume size              : 952741 MB
Total sectors            : 1951215615
Total clusters           : 243901951
Free clusters            : 171459886
Free space               : 669765 MB (70% of drive)

Allocation Size
----------------
Bytes per sector         : 512
Bytes per cluster        : 4096
Bytes per MFT record     : 0
Clusters per MFT record: 0

MFT Information
---------------
MFT size                 : 1643 MB (0% of drive)
MFT start cluster        : 786432
MFT zone clusters        : 52861536 - 52912768
MFT zone size            : 200 MB (0% of drive)
MFT mirror start         : 2

Meta-Data files
---------------
```

10. The MFT dump will be created in the same directory
11. Type download "MFTdump.tbd" in Live response CLI window and in a few sec you will be asked to download the file

Congrats - you just acquired the MFT table snapshot!

Conclusion

We've demonstrated how you can initiate a live response session and perform basic remediation, get a mini memory dump using a PowerShell script, and use forensic tools such as Sysinternals to acquire forensic information on a machine.

This tutorial emphasizes the typical scenarios and commands that would be useful when an in-depth investigation and remediation is needed on a compromised device.

We hope you enjoyed this tutorial and are now encouraged to explore live response as well as other features and capabilities. For more information, read the product guide at docs.microsoft.com.

Click the feedback icon in Microsoft Defender Security Center to let us know how you feel about this tutorial or any other aspects of the product. We would love to hear your ideas about additional simulations and tutorials. Thank you!