WHITEPAPER

Windows Malicious Events Detection with Security Monitoring

26.10.2017
Index

1. Introduction .................................................................................................................. 3
   1.1. Scope ......................................................................................................................... 4
   1.2. Objective ................................................................................................................... 4
   1.3. Methodology ............................................................................................................. 4
2. Basic security concepts in Windows .............................................................................. 4
3. Proof of concept process ............................................................................................... 5
   3.1. Identification of events ........................................................................................... 5
   3.2. Constructing a decision tree .................................................................................... 6
   3.3. Configure individual events and detect them with Security Monitoring ................. 7
      3.3.1. Discovering events by eventID ............................................................................. 13
      3.3.2. Discovering events by eventID and message settings ..................................... 17
      3.3.3. Discovering other events without the eventID .................................................. 23
      3.3.4. ElevenPaths Threat Intelligence alert library .................................................. 27
   3.4. Implementing the decision tree ............................................................................... 28
Bibliography ...................................................................................................................... 30
About the authors ............................................................................................................. 31
About ElevenPaths ........................................................................................................... 32
More information ............................................................................................................. 32
Executive summary

Computers that use the Microsoft Windows operating system naturally generate a large quantity of security and status information in the form of events. While the majority of such events are not necessarily risks, they may be indications that something else is happening. When a series of events happens within given periods of time, those indications may come to represent sure signs.

All operating systems, be they large servers or work stations, supervise their activity through registers or logs. In the Microsoft Windows operating system, those logs are produced in the form of events that can be viewed in the Windows Event Viewer.

Such events describe activities such as unsuccessful logon attempts, changes in auditing, time changes, etc. They are events that, in and of themselves, do not represent a risk to the system, but could be indications of malicious activity.

Other security characteristics embedded in the operating system, such as the Windows Firewall, also have the ability to generate activity logs that can give us indications of malicious activities.

When a series of certain events occurs in a given period, we can certify that a computer is actually at risk and, accordingly, generate an alert that malicious activity is occurring within the computer.

1. Introduction

The Microsoft Windows operating system generates a large quantity of events that are described in the technical documentation published by Microsoft in its knowledge base, Technet [1].

Windows logs both events occurring within the operating system and in applications and services installed on the computer. Depending on the auditing level that we enable in the computer, we will be able to see the activity of all such events in the event viewer.

Exploring each of the event logs in multiple computers is an arduous task that requires a lot of time and resources. Automatic tools can be used to identify and correlate events, allowing us to identify patterns that correspond with malicious behaviours and, accordingly, to generate automatic alerts when they occur.

Security Monitoring has this potential and in this whitepaper we will describe a few small proof of concept tests that were performed on the computer of the Chief Data Officer of Telefónica, using this tool to identify such patterns.

1.1. Scope

The scope of this document is to collate the proof of concept tests that were performed using Security Monitoring to identify malicious events and series of events that indicate patterns of malicious behaviours in computers using the Windows operating system.
1.2. Objective

The objective of this document is to describe the potential of Security Monitoring as a tool to detect malicious events and behaviours.

1.3. Methodology

The following steps were followed in carrying out the tests:

- **Analysis of frequent attacks on Windows systems**: Based on the knowledge of attempts to penetrate into Windows systems, the detection procedure is described and the point at which the system is put at risk is identified.

- **Identifying indicators of malicious activity**: Among the list of events that the Windows operating system generates, a number of events were selected that are indicators of malicious activity, but that, considered individually, do not necessarily represent a risk.

- **Generating decision trees**: Immediately after, decision trees are generated through which a series of events in a concrete period of time indicate that the system has been compromised and, thus, that an alert must be produced.

- **Implementing in Security Monitoring**: Once the scope has been identified, this intelligence is transferred to Security Monitoring so that the tool can use this knowledge and be capable of automating this process.

The work performed in the present proof of concept tests do not come close to covering all possible cases. A mere selection of identification examples is presented to demonstrate the platform's flexibility in carrying out this type of detection.

2. Basic security concepts in Windows

Several types of indicators were considered in identifying security problems in Windows machines. Firstly, there are Windows events. Secondly, there are those in the infrastructure in which the machine is located (communications). Last, but not least, there are events related to vulnerabilities that are identified in pen testing processes.

In the pen testing process, the first step is discovery. In this phase, the goal is to identify, for example, a machine's open ports and the services running through those ports. This is done to, as a second step, exploit them and, finally, gain access and succeed in maintaining continuous access through a backdoor service or something similar.

Windows events allow one to identify actions linked either to the operating system or the infrastructure, which can indicate that such a process is occurring. If the Windows firewall reports connections to several ports via the network, followed by repeated login attempts to the ssh service and, finally, a legitimate ssh access, it is more than certain that the computer has been compromised.

Windows naturally identifies such events and allows one to view them in the **event viewer**, but one must search through all of those events in order to identify this behaviour. A tool to retain and correlate events, like Security Monitoring, automatically performs this task, generates alerts and maintains the computer under control at all times.
3. Proof of concept process

The first challenge that we are up against is to extract the specific processes of the generic indicators that will help us in identifying several malicious processes. For example, malware tends to install itself as a service in the system, but legitimate programs do so as well. What happens before that? A program with privileges is executed. For now, both of these processes are also performed by legitimate programs. However, if repeated attempts to login to the system are identified beforehand, that is not normal.

Firstly, we will address how to identify events that could be indicators, combine them to identify behaviours and, accordingly, obtain certainty that the computer has been compromised.

3.1. Identification of events

As mentioned above, the Microsoft knowledge base provides us access to events in the system that are hierarchically organized according to the type of event.

The first division is if the events are system, security or application-related. The system’s logs register events from the Windows system’s components. Security logs register events related to the use of the system’s resources. Application events correspond with applications and services.

Each event is attributed a unique eventID number that will help us to filter it within the series of events that we want to identify. For example, the event with eventID 4688 corresponds with the execution of a new process [2].

In this way, there are events that, through their eventID, can help us in identifying them. For example, the event with eventID 600 corresponds with the execution of a powershell.

However, there are events that reflect system processes capable of doing several things. Thus, identifying the eventID is not enough; values in the event’s message must also be identified.

For example, the event with eventID 4950 corresponds with “MPSSVC rule-level policy changes”. Since we are looking for deactivations in the firewall, we will look for a message containing “Enable” and “No”.

A new problem emerges here: the language of the operating system. The internationalization of Windows has made it such that, for the same event, different messages are produced if the system has been installed in different languages. Thus, the message in English is: “A Windows firewall setting was changed. Changed Profile: Public New Setting: Type: Enable Windows Firewall Value: No” and in Spanish: “Se cambió una configuración de firewall de Windows. Perfil cambiado: Público Nueva configuración: Tipo: Habilitar Firewall de Windows Valor: No”.

Due to this internationalization, identifying events is further complicated. In demonstrating the present works, we reduce the cases to those in Spanish and English.

Besides Windows events, we will also work with Windows Firewall messages. Within the operating system, we are able to enable the log of firewall events for each connection profile (public, private and domain if within one) and register incoming and/or outgoing connections, if desired.
Since this option is not enabled by default, it must be enabled in order to obtain the corresponding information. It is also possible to specify the file in which the events log will be kept and its maximum size.

That file keeps activity logs of the computer’s network interface communications up to the level that has been enabled in the connection.

Now that we have clarified the sources of events as well as how to identify them, it is possible to construct the behaviour that we are attempting to detect.

### 3.2. Constructing a decision tree

A series of behaviours has been identified that could lead to the computer being compromised:

![Decision Tree Diagram]

The first difficulty in these processes is knowing when to consider that a computer has actually been compromised. Three states in which the computer may be in the process of doing so have been identified and will serve as a stoplight indicator system to set off alerts:

- The computer should not be at risk (green).
- There are signs that the computer is at risk (orange).
The computer is at risk (red).

With this classification, we reach the following image:

![Diagram](image)

Figure 3. Decision tree of events with computer states.

Now we identify when an alert should be sent. If the state is in red, the computer is compromised and an alert should be generated.

Having determined what should happen, the following step is to identify when it must happen. The following image represents this step:

![Diagram](image)

Figure 4. Decision tree with computer states and process times.

Now we have obtained a clear view of the events that should happen and when they should happen. This is the moment to configure individual events and detect them with Security Monitoring.

### 3.3. Configure individual events and detect them with Security Monitoring

We individually configure the events that must be identified for the decision tree. Moreover, two more naturally malicious events will be configured: port scan, which is an indicator in the penetration reconnaissance phase, and the execution of a Meterpreter process, which responds to the exploitation phase in a computer.
<table>
<thead>
<tr>
<th>Event</th>
<th>EventID</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Powershell event</td>
<td>600</td>
<td>Event occurs one or more times in 1 minute</td>
</tr>
<tr>
<td>Auditing settings were changed</td>
<td>4907</td>
<td></td>
</tr>
<tr>
<td>A scheduled task was created</td>
<td>4698</td>
<td></td>
</tr>
<tr>
<td>A service was installed in the system</td>
<td>4697</td>
<td></td>
</tr>
<tr>
<td>The system time was changed</td>
<td>4616</td>
<td></td>
</tr>
<tr>
<td>A Windows Firewall setting has changed</td>
<td>4950</td>
<td>Message contains “Habilitar” or “Enable” and “No”</td>
</tr>
<tr>
<td>Port scan (FW log)</td>
<td></td>
<td>&gt;=5 different TCP ports from the same private source IP different from XXX.XXX.XXX.255 to the same destination IP in 5 minutes</td>
</tr>
<tr>
<td>Meterpreter</td>
<td>7045</td>
<td>• Message contains “powershell” and %COMSPEC%” OR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Service number “M+15 characters more” and message contains “%SYSTEMROOT%?????????.exe”</td>
</tr>
</tbody>
</table>

Once the events to be detected have been singled out, the next step is to make Security Monitoring identify these events.

The Security Monitoring agent for Windows contains the following service stack to perform real-time monitoring:
- **MagicLog**: Collects the machine’s log files to be monitored.
- **MagicEvent**: Collects Windows events.
- **MonitorService**: Monitors the machine’s settings.
- **Snare**: Monitor’s the machine’s health.
- **ProxyServerContainer**: Sends the logs of all previous services to the platform collector.

By default, **MagicEvent** collects Windows’ basic system and security events. But, since events must be collected from **Powershell** and other applications as well, it must be configured to include these events by adding their paths.
Once MagicEvent has been configured, MagicLog must be configured to collect the logs generated by Windows Firewall. First, Windows firewall audit logging must be enabled in Control Panel → Windows Firewall → Advanced Configuration → Properties.
In the tab of each profile to be monitored, “Customize” must be clicked on under “Login” to enable events to be logged in the following window:

Figure 7. Windows Firewall properties.

Once this has been done, **MagicLog** can be configured to collect these logs:
The folder path and file pattern of the Windows firewall log file must be specified, as well as the tags that will be added by the agent to ensure correct file parsing. Security Monitoring automatically identifies, parses and classifies the technology and type of file that is arriving through these tags. Thus, it is important to be careful when adding them. For Windows firewall, it is “firewall.windows.stdout”.

Now the computer is sending all required information to the Security Monitoring platform. Events may now be generated and data may be filtered to be subsequently singled out.

Within Security Monitoring, these operations are performed and alerts are created under “Log Management”:
We will now describe three cases to avoid saturating the document. The first will be an individual event identified by its eventID, the second using additional settings and the third using parsed fields from a log.

### 3.3.1. Discovering events by eventID

The example that will now be developed is the detection of Powershell events. As described in the previous table, to detect the execution of a Powershell script, we will look for events with eventID 600.

Entering into “Log Management”, Security Monitoring’s “Finder” will allow events to be searched for by category:
Figure 11. Complete view of the Security Monitoring Finder.

Windows machines’ events can be found under the category “box.win”. By selecting this category, all logs from Windows machines will be shown:

Figure 12. Event logs for box.win.

As one may observe, there are many events and it is difficult to look for an event manually. The “Log Management” section has a series of tools, accessed through its toolbar in the top right-hand part of the window, that will facilitate this task. Using the filter button, it is possible to show only events with eventID 600 by filtering the table’s values where the eventID column equals 600:
As one may observe, once the filter has been applied, only events with `eventID 600` are shown:

Now the alert will be created. To do so, the alert button must be clicked on in the toolbar and configured to occur when an event happens more than once in 1 minute:
NEW Alert Definition

You can set alerts that will start running in real time as they are created. Choose the alert type you want to create. You might use the column names in the description typing $columnName (case sensitive).

Message:
PowerShell

Description:
A PowerShell Process was launched on $Machine

Subcategory:
11Paths

Context:
PowerShell

EACH    SEVERAL    LOW

One alert when several events occurs in a period of time

Period:
1m

Threshold:
1

Keep counters for each value in columns

Drag and drop columns

CANCEL    CREATE

Figure 15. New PowerShell alert definition
When the alert is triggered, the following message is received:

```
Se ha creado un proceso powershell

el usuario VICTIM\Administrator ha creado un proceso powershell en la maquina victim:

Se creó un nuevo proceso. Sujeto: Id. de seguridad: S-1-5-21-568167839-2834584188-1483821995-500 Nombre de cuenta: Administrator Dominio de cuenta: VICTIM
Id. de inicio de sesión: 0x1206F Información de proceso: Id. de nuevo proceso: 0xe49F Nombre de nuevo proceso: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe Tipo de elevación de token: TokenElevationTypeDefault (1) Id. de proceso creador: 0x3F4 Línea de comando de proceso: Tipo de elevación de token indica el tipo de token que se asignó al nuevo proceso de acuerdo con la directiva Control de cuentas de usuario. El tipo 1 es un token completo sin privilegios quitados ni grupos deshabilitados. Solo se usa un token completo si Control de cuentas de usuario está deshabilitado o si el usuario es la cuenta predeterminada Administrator o una cuenta de servicio. El tipo 2 es un token elevado sin privilegios quitados ni grupos deshabilitados. Se usa un token elevado cuando Control de cuentas de usuario está habilitado y el usuario elige Iniciar el programa mediante Ejecutar como administrador. También se usa un token elevado cuando se configura una aplicación para que siempre requiera un privilegio administrativo para que siempre requiera el máximo privilegio y el usuario pertenece al grupo Administradores. El tipo 3 es un token limitado con los privilegios administrativos quitados y los grupos administrativos deshabilitados. El token limitado se usa cuando Control de cuentas de usuario está habilitado, la aplicación no requiere un privilegio administrativo y el usuario no elige Iniciar el programa mediante Ejecutar como administrador.
```

![Figure 16. Powershell alert message](image_url)

### 3.3.2. Discovering events by eventID and message settings

In this case, we will demonstrate the example of detecting a deactivation of the Windows Firewall. This will be done through an automatic graph by filtering eventID information and the content of the event message.

Looking at the table of events, the description of the detection corresponds with when the eventID is 4950 and when the field extMessage contains the words “Enable” and “No”.

First, access “Log Management” and filter the logs by the tags box and win, as performed previously.
Figure 17. Logs of events for box.win.

Now we will separate events so that only those with eventID 4950 are shown. Using the “Filter” button in the toolbar, we fill out the form that is displayed specifying that we want to filter events with an eventID value equal to 4950:

Figure 18. Filter events when eventID equals 4950.

The outcome is the events of all computers with an eventID equal to 4950:
But this shows all events corresponding with firewall policy changes. By observing the message that is produced by Windows Firewall upon its deactivation ("A Windows Firewall setting was changed. Changed Profile: Public New Setting: Type: Enable Windows Firewall Value: No") we can unequivocally detect this event if the field extMessage contains the words Enable and No.

As mentioned previously, the intention exists to internationalize detection efforts and make messages appear in Spanish and English. The message in Spanish is: "Se cambió una configuración de firewall de Windows. Perfil cambiado: Público Nueva configuración: Tipo: Habilitar Firewall de Windows Valor: No".

Thus, unquestionable detection will be reached here when the message (extmessage) produced contains "enable" and "no".

This means that 2 message filters must be applied. The function "contains" in Security Monitoring’s filter is inclusive. This means that if in a single filter it is specified that messages must be separated which contain "enable" and "no", the filter will show events that contain both terms and those that contain only one of the terms. Accordingly, the detection will not be precise since it will detect both activation and deactivation events.

This helps us, however, given the international context. First, we apply a filter for "contains", “Enable” and “Habilitar”:
As one may observe, this separates messages in both languages:

Figure 21. Events containing Habilitar or Enable in the extMessage field.

Now we add the second filter for when the message contains "No":

Figure 20. Filtering events containing Habilitar or Enable in the extMessage field.
<table>
<thead>
<tr>
<th>Operation</th>
<th>Filter Data</th>
<th>Group By</th>
</tr>
</thead>
</table>

**Arguments**

- **Value**: extMessage
- **contains**: No

Thus, detection is complete and unequivocal for both languages:

Figure 22. Events containing “Habilitar” or “Enable” in the extMessage field.

Figure 23. Detecting the Firewall deactivation event in English and Spanish.

Now an alert can be created using the “Alerts” button from the toolbar and specifying (with “each”) that it be triggered each time that the event occurs:
You can set alerts that will start running in real time as they are created. Choose the alert type you want to create. You might use the column names in the description typing $columnName (case sensitive).

Message:
Se ha detenido el Firewall de windows

Description:
Se ha detenido el firewall de Windows en la máquina $machine.

Detalles:
 SextaMessage

Subcategory: 11Paths
Context: Firewall Detenido

EACH
SEVERAL
LOW

One alert for each event

Figure 24. Configuring the alert for firewall deactivation in English and Spanish.

When it detects the event, the alert that will be triggered is:

Se ha detenido el Firewall de windows
Se ha detenido el firewall de Windows en la máquina víctima.

Detalles:
Se cambió una configuración de Firewall de Windows. $oldProfile; $newProfile; Perfil cambiado: $oldProfile; $newProfile; Nueva configuración: $newProfile; $newProfile; Tipo: Habilitar Firewall de Windows $oldProfile; $newProfile; Valor: No

Figure 25. Alert for firewall deactivation in English and Spanish.
3.3.3. Discovering other events without the eventID

In this case, Windows Firewall events will be used to detect a port scan in a Windows machine. These events are categorized as "Firewall. windows. stdout", as we saw in the previous section on configuring event collection. These events do not have an "eventID", but the capability of allows us to separate values very simply.

The first step is to determine the algorithm that will be used. Looking at the description in the previous event table, “> = 5 different TCP ports from the same private source IP different from XXX.XXX.XXX.255 to the same destination IP in 5 minutes" and the fields available in the table "Firewall. windows. stdout":

| eventdata | host | serverdate | action | protoc... | srcIP | dstIP | srcPort | dstPort | size | tcpflags | tcpseq | tcpack | tcpwin... | icmpType | icmpCode | in... | path |

Figure 26. Fields in the firewall. windows. stdout table.

This is the description of the following algorithm and the new, more complex steps:

- Filter by same TCP protocol.
- Filter by received traffic (path equal receive). We have already seen how to perform the first two steps in the previous cases.
- The destination IP must be private: the "logtrust" engine has a filter function named "isPrivateIPv4" that filters private IPs.

![Filter private IPv4 destination addresses.](image)

- Convert the source IP into a "string" and filter those that end in 255 to eliminate the network broadcast messages. To do that, a new column named "srcIPSTR" will be created with the function "ToString" starting from the column "srcIP".
The new column is filtered by those that end in 255 and are negated since those which are needed are all those that do not end in 255.

Group the events every 5 minutes and count the different ports. To do this, the "group by" button located in the toolbar is used. It is applied by host and source IP (srcIP).
If there are more than 5, the alert will be triggered. To do this, it must be specified manually using the "query editor" from the toolbar and adding the following lines:

```
select h1lppcount(dstPort) as dstPort,
floor(dstPort) as puertosDistintos
where puertosDistintos >= 5
```

These lines separate different ports into a new column called "puertosDistintos" and only show cases in which the value is greater than or equal to five.
The outcome is then shown:

Figure 31. Query Editor with the complete search.

Now, a new alert is created by clicking on the corresponding button and filling out the fields identically to the way that was shown previously. The options for the times that alert occurs have disappeared since the grouping tool did this task.
When performing a port scan on the Windows machine “squarefighter”, the message received is:

Figure 34. Port scan alert.

3.3.4. ElevenPaths Threat Intelligence alert library

These and other alerts are displayed in an alert library within Security Monitoring called ElevenPaths Threat Intelligence, where all alerts will be added as new ones are created.
3.4. Implementing the decision tree

Once individual alerts have been configured, this intelligence must be inserted in the Security Monitoring correlator so that it will detect the series of events over time. This task cannot be performed by the user by now. Although make it available, it is part of the roadmap of the Logtrust platform’s functionalities.

The task for ElevenPaths and Telefonica CDO has been to configure behaviours in accordance with certain periods of time in the decision tree. The developers of the “Logtrust” platform included two alerts that reflect these behaviours.

As described previously, once the first red is obtained, the alert will trigger. Accordingly, the alerts are described as follows:

- **Server Risk Audit:**
  - Case 1: A Powershell event is executed (eventId 600) and in less than one minute the computer’s audit settings are changed (eventId 4907).
Case 2: A Powershell event is executed (eventID 600) and in less than one minute a service is installed (eventID 4697) and in less than 10 minutes the audit settings are changed (eventID 4907).

- Server Risk Firewall:
  - Case 1: A Powershell event is executed (eventID 600) and in less than one minute the firewall is deactivated (eventID 4950, extMessage contains “Habilitar” or “Enable” and “No”)
  - Case 2: A Powershell event is executed (eventID 600) and in less than one minute a service is installed (eventID 4697) and in less than 10 minutes the firewall is deactivated (eventID 4950, extMessage contains “Habilitar” or “Enable” and “No”)

In this case, the alert message contains the differential point of the branch, indicating that it was after the suspicious activity occurred (deactivation of the FW):

![Figure 37. Deactivation of Firewall Alert.](image-url)
Bibliography


About the authors

Pablo González Pérez  
Security Researcher  
CDO - Telefonica

Santiago Hernández Ramos  
Security Researcher  
CDO - Telefonica

Santiago R. Urbano López de Meneses  
Global Product Manager – ElevenPaths
About ElevenPaths

In ElevenPaths, Telefónica’s Cyber Security department, we believe that challenging the current state of security should always be an aspect of technology. We continuously question the relationship between security and people in the process of creating innovative products that are capable of transforming the concept of security. In doing so, we strive to stay one step ahead of our attackers, who are ever more present in our digital life.

More information

www.elevenpaths.com
@ElevenPaths
blog.elevenpaths.com