

# VMRay Sandbox Analyzer Function for IBM Resilient


## Table of Contents



- [app.config.settings](#)
- [Function Inputs](#)
- [Function Output](#)
- [Pre-Process Script](#)
- [Post-Process Script](#)

## - Rules

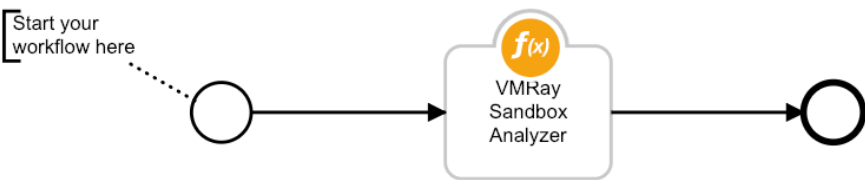
This package contains a function that executes a VMRay Sandbox Analyzer of an Attachment or Artifact and returns the Analysis Report to IBM Resilient.

Workflows / Example: VMRay Sandbox Analyzer [Attachment]

 [Cancel](#) [Save & Close](#) [Save](#)

Name *	Example: VMRay Sandbox Analyzer [Attachment]	Creator	 Resilient Sysadmin
API Name *	example_vmray_sandbox_analyzer_attachment	Last Modified	01/29/2019 08:27
Description	An example of having an attachment sample analyzed by VMRay Sandbox Analyzer	Last Modified By	 Resilient Sysadmin
Object Type *	Attachment	Associated Rules	<b>Example: VMRay Sandbox Analyzer [Attachment]</b>

Start your workflow here



```
graph LR; Start(( )) --> VMRay[VMRay Sandbox Analyzer]; VMRay --> End(( ))
```

- Supports an attachment or artifact that is a file.
- In future it will allow users to select the type of report, PDF, HTML, or JSON, which is returned from

VMRay Sandbox, now only support JSON.

- Supports a proxy. Just add your proxy details to the `proxy` section in `app.config` file.

## app.config settings:

---

```
# Your VMRay Analyzer API Key

vmray_api_key=

# Your VMRay Server URL, using https://cloud.vmrays.com if empty.
vmray_analyzer_url=https://cloud.vmrays.com

# Amount of time in seconds to wait until checking if the report is ready again.
vmray_analyzer_report_request_timeout=60
```

---

## Function Inputs:

---

Function Name	Type	Required	Example	Info
<code>incident_id</code>	<code>Number</code>	Yes	<code>1001</code>	The ID of the current Incident
<code>attachment_id</code>	<code>Number</code>	No	<code>5</code>	The ID of the Attachment to be analyzed
<code>artifact_id</code>	<code>Number</code>	No	<code>6</code>	The ID of the Artifact to be analyzed
<code>analyzer_report_status</code>	<code>Boolean</code>	Yes	<code>False</code>	Has the analysis report generated successfully. Options are: <code>True</code> or <code>False</code>

---

## Function Output:

---

```
results = {  
    "analysis_report_status": analysis_report_status,  
    "incident_id": incident_id,  
    "artifact_id": artifact_id,  
    "attachment_id": attachment_id,  
    "sample_final_result": sample_final_result  
}
```

---

## Pre-Process Script:

Example: VMRAY Sandbox Analyzer [Attachment]

```
inputs.incident_id = incident.id  
inputs.attachment_id = attachment.id
```

Example: VMRAY Sandbox Analyzer [Artifact]

```
inputs.incident_id = incident.id  
inputs.artifact_id = artifact.id
```

---

## Post-Process Script:

This example adds a Note to the Incident and color codes the `analysis_status` depending if it was **malicious** or **clean**

```
def font_color(vti_score,sample_severity):  
    color = "green"  
    try:  
        if sample_severity in ["malicious"] or int(vti_score) >= 75:  
            color = "red"  
        elif sample_severity in ["blacklisted","suspicious"] or int(vti_score) >= 50:  
            color = "yellow"  
    except:  
        pass  
    return color  
  
noteText = u"" "Successful submit <b>{}</b> to VMRay Analyzer.Check the results below:  
<br>"" ".format(attachment.name)
```

```

for sample in results.sample_final_result:
    noteText += u"""-----
----"""
    color = font_color(sample["sample_report"]["sample_score"],sample["sample_report"]["sample_last_reputation_severity"])
    noteText += u"""<br>VMRay Sandbox Analysis: <b>{sample_filename}</b> complete.<br>
VMRAY Online Attachment: <a href={sample_online_report}>{sample_online_report}</a><br>
VMRay Analyzer result: VTI Score: <b style= "color:{color}">{sample_vti_score}</b>, Severity: <b style= "color:{color}">{sample_severity}</b> <br>
""".format(sample_filename=sample["sample_report"]["sample_filename"],
sample_online_report=sample["sample_report"]["sample_webif_url"],
color = color,
sample_vti_score = sample["sample_report"]["sample_score"],
sample_severity = sample["sample_report"]["sample_last_reputation_severity"])

    noteText += u"""<br>| analysis_id | analysis_job_started | analysis_vti_score | analysis_severity |<br>"""

    for analysis in sample["sample_analysis_report"]:
        color = font_color(analysis["analysis_vti_score"],analysis["analysis_severity"])
        noteText += u"""| <a href={analysis_link}> {analysis_id} </a> | {analysis_job_started} | <b style= "color:{color}"> {analysis_vti_score}</b> | <b style= "color:{color}">{analysis_severity}</b> |<br>
""".format(analysis_link=analysis["analysis_webif_url"],
analysis_id=analysis["analysis_id"],
analysis_job_started=analysis["analysis_job_started"],
analysis_vti_score=analysis["analysis_vti_score"],
analysis_severity=analysis["analysis_severity"],
color=color)

    reputations = [str(reputation["reputation_lookup_severity"]) for reputation in sample["sample_reputation_report"]]

    if "malicious" in reputations:
        color = "red"
        reputation_lookup_severity = "malicious"
    elif "suspicious" in reputations:
        color = "yellow"
        reputation_lookup_severity = "suspicious"
    elif "blacklisted" in reputations:
        color = "yellow"
        reputation_lookup_severity = "blacklisted"

```

```

elif "not_suspicious" in reputations:
    color = "green"
    reputation_lookup_severity = "not_suspicious"
elif "whitelisted" in reputations:
    color = "green"
    reputation_lookup_severity = "whitelisted"
else:
    color = "green"
    reputation_lookup_severity = "unknown"

noteText += u"""Reputation lookup result: <b style= "color:{color}">{reputation_lo
okup_severity} </b> <br>""".format(color=color, reputation_lookup_severity=reputation
_lookup_severity)

incident.addNote(helper.createRichText(noteText))

```

### Example of adding a incident note from post-processing scripts:

 Resilient Sysadmin added a note to the *Incident* 01/29/2019 07:50



Successful submit **0655d58db2798ad8336f92dd580f988312f37f3e52b405c9c71d3afd2bd2c290** to VMRay Analyzer. Check the results below:

-----  
VMRay Sandbox Analysis: **0655d58db2798ad8336f92dd580f988312f37f3e52b405c9c71d3afd2bd2c290.rtf** complete.

VMRAY Online Attachment: <https://cloud.vmrays.com/user/sample/view?id=2996559>

VMRay Analyzer result: VTI Score: **100**, Severity: **blacklisted**

analysis_id	analysis_job_started	analysis_vti_score	analysis_severity
2668927	2019-01-30T15:41:48	100	malicious
2668919	2019-01-30T15:37:27	100	malicious
2668886	2019-01-30T15:32:43	100	malicious
2668854	2019-01-30T15:28:45	100	malicious
2668848	2019-01-30T15:24:45	100	malicious

Reputation lookup result: **blacklisted**

## Rules

Rule Name	Object Type	Workflow Triggered
Example: Joe Sandbox Analysis [Artifact]	Artifact	Example: VMRay Sandbox Analyzer [Artifact]
Example: VMRay Sandbox Analyzer [Attachment]	Attachment	Example: VMRay Sandbox Analyzer [Attachment]

