

# Yara Rules

## What is Yara?

Yara is a powerful engine that helps to identify malware (mostly).

Below is the short description of the engine by its developers taken from their [github](#):

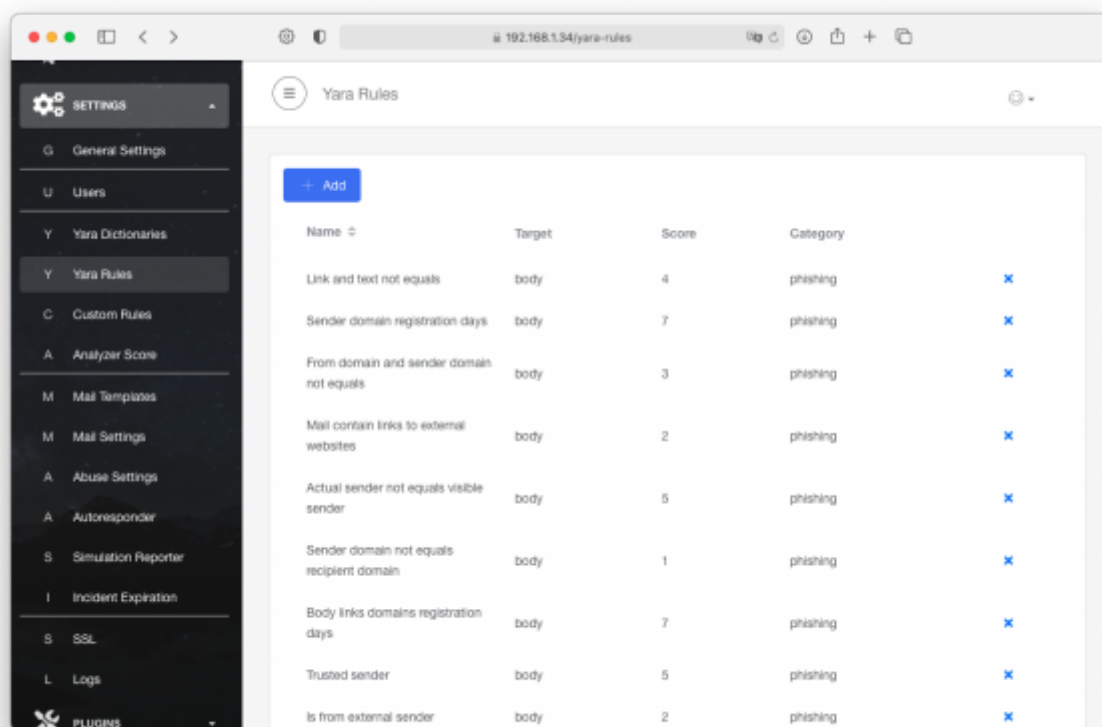
*YARA is a tool aimed at (but not limited to) helping malware researchers to identify and classify malware samples. With YARA you can create descriptions of malware families (or whatever you want to describe) based on textual or binary patterns. Each description, a.k.a rule, consists of a set of strings and a boolean expression which determine its logic.*

In the Screener, Yara is used to analyze \*.eml files.

In case if the received report is \*.msg, it'll be automatically converted into \*.eml.

## Yara Rules

In the Settings → Yara Rules there is a list of predefined Rules that can be Modified.



Name	Target	Score	Category	
Link and text not equals	body	4	phishing	✕
Sender domain registration days	body	7	phishing	✕
From domain and sender domain not equals	body	3	phishing	✕
Mail contain links to external websites	body	2	phishing	✕
Actual sender not equals visible sender	body	5	phishing	✕
Sender domain not equals recipient domain	body	1	phishing	✕
Body links domains registration days	body	7	phishing	✕
Trusted sender	body	5	phishing	✕
Is from external sender	body	2	phishing	✕

Let us review the very first Yara rule as an example.

The screenshot shows a web browser window with the URL `192.168.1.34/yara-rules/1`. On the left is a dark sidebar menu with the following items: SETTINGS (with a gear icon), General Settings, Users, Yara Dictionaries, Yara Rules (highlighted), Custom Rules, Analyzer Score, Mail Templates, Mail Settings, Abuse Settings, Autoresponder, Simulation Reporter, Incident Expiration, SSL, Logs, and PLUGINS (with a gear icon). The main content area is titled 'Link and text not equals' and contains the following fields:

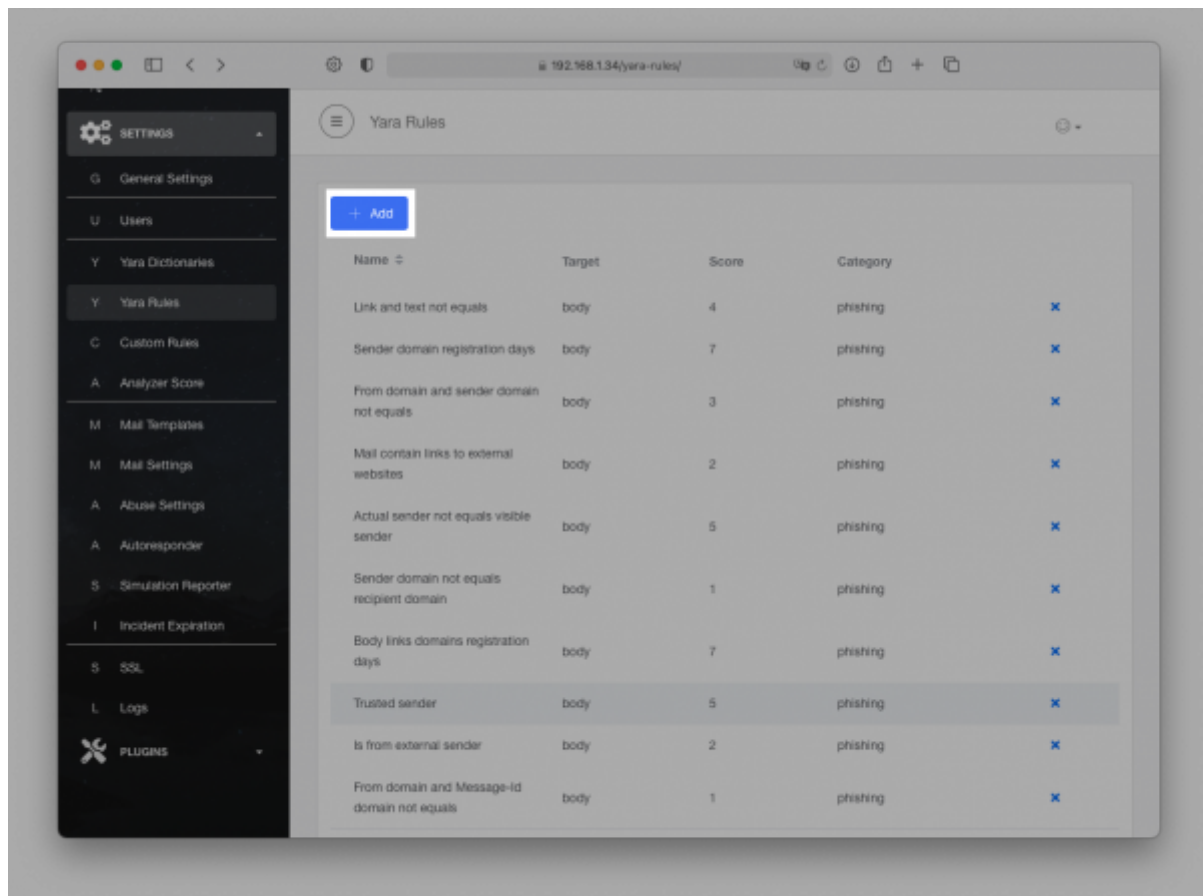
- ENABLED**: A checkbox that is checked.
- NAME**: A text input field containing 'Link and text not equals'.
- TARGET**: Radio buttons for 'BODY' (selected) and 'ATTACHMENT'.
- RULE CONSTRUCTOR**: A section with the text 'Condition Not Supported'.
- YARA RULE TEXT**: A text area containing the following YARA rule:

```
for any i in (D eml.number_of_http_links) : { eml.http_links[i].text matches /[a-zA-Z]*<\/[a-zA-Z0-9~%&\/_@-]*> and  
eml.http_links[i].link != eml.http_links[i].text }
```
- DESCRIPTION**: A text area containing the text 'Is the link display name (the visible link vs link when you hover over) different from the actual link?'.
- SCORE**: A numeric input field with the value '4'.
- CATEGORY**: A text input field containing 'phishing'.
- Buttons**: 'Save' and 'Cancel' buttons at the bottom.

- Link and text not equal - The name of the Rule.
- Enabled\Disabled - Status of the rule. Whether it is active\inactive.
- Name - New name of the Rule. Applied after a save.
- Target (Body\Attachement) - What to scan, the body of an \*.eml file or attachment in it.
- Rule Constructor - In the example, it can not display too complex Yara rule.
- Yara Rule Text - Text of the rule itself.
- Description - Description that helps a user to understand the purpose of the rule.
- Score - Amount of score that is going to be assigned to an incident.
- Category - To what category the rule is applied.

## Add a Rule

To add a new rule, please use the +Add button.



The New Yara Rule page allows building a new rule upon user's requirements.

The screenshot shows the 'New Yara Rule' form. It includes a 'YARA RULE FILE' section with a 'Choose File' button and 'no file selected' text. Below this is an 'ENABLED' checkbox. The 'NAME' field is labeled 'Name'. The 'TARGET' section has radio buttons for 'BODY' (selected) and 'ATTACHMENT'. The 'RULE CONSTRUCTOR' section has an 'Add Condition Block' button. The 'YARA RULE TEXT' field is a large text area. Below it is a 'DESCRIPTION' text area. The 'SCORE' field is a numeric input with a spinner. The 'CATEGORY' field is a text input. At the bottom are 'Save' and 'Cancel' buttons.

## Rule Constructor

The most interesting part of the page is the Rule Constructor. It is separated into Condition Bloks. Each block has:

Field	Condition	Value or Field
Delivered To	Equals To	Delivered To
Subject	Not Equals To	Subject
Date	Contains	Date
From	Not Contains	From
To		To
CC		To
BCC		BCC
Actual Sender Email		Actual Sender Email
Visible Sender Email		Visible Sender Email
Recipient Email		Recipient Email
Origin Server Hostname		It can be configured to have any string or digital value

In case if multiple expressions are set in the same block. a "Rule Operator" option will appear. The created Conditional Block will assign some spam score to an incident only if any of the conditions matched\all of the conditions matched.

[Let us review an example below.](#)

RULE OPERATOR

all

RULE OPERATOR

any

FIELD	CONDITION	VALUE OR FIELD	ACTIONS
Subject	equals to	"Ipad"	✗
From	not equals to	"apple@apple.com"	✗

Add Condition Delete

FIELD	CONDITION	VALUE OR FIELD	ACTIONS
To	equals to	"trusty@victim.com"	✗

Add Condition Delete

Add Condition Block

YARA RULE TEXT

( eml.subject == "Ipad" or eml.from != "apple@apple.com" ) and ( eml.to == "trusty@victim.com" )

The rule will work only if both Conditional Blocks are triggered, since it is set to "All". Conditional Block #1 will be triggered if any of the conditions are triggered, which are Subject equals to "Ipad" and the email is not from "apple@apple.com". Conditional Block #2 will be triggered if the only condition is true, the email is purposed to a guy with email "trusty@victim.com".

As the result, the constucture has generated the following Yara rule:

```
( eml.subject == "Ipad" or eml.from != "apple@apple.com" ) and ( eml.to == "trusty@victim.com" )
```

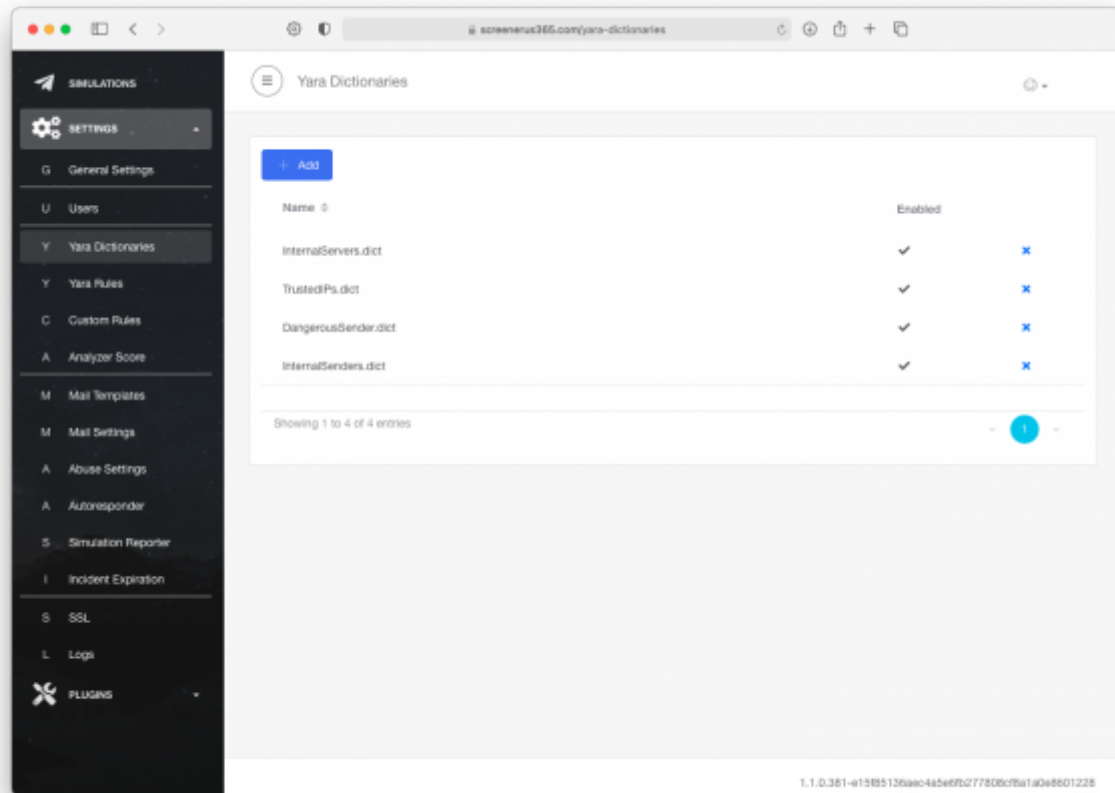
## Useful Example

Let us add a Yara rule that would be pretty useful.

In this example, we will create a [dictionary](#) of trusted domain names, then we will use the dictionary to give a negative spam score to an incident.

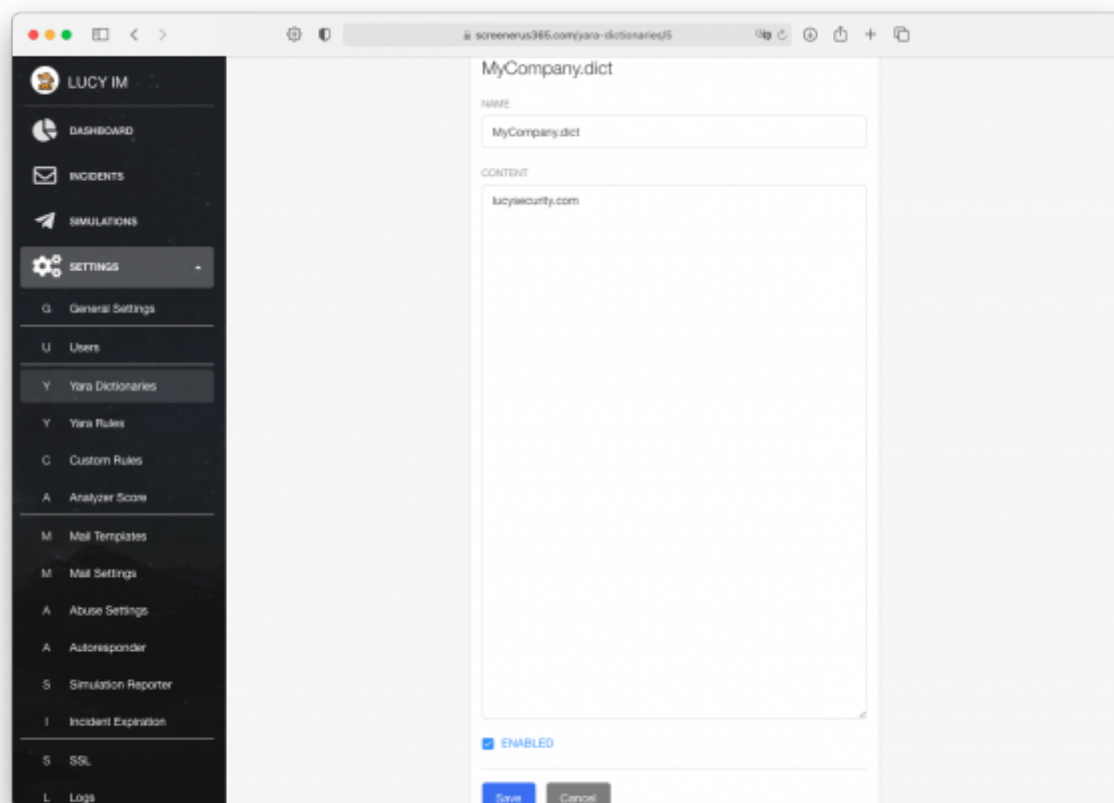
### Step 1: Create the dictionary

[Go to Settings](#) → [Yara Dictionaries](#) → [+Add](#).



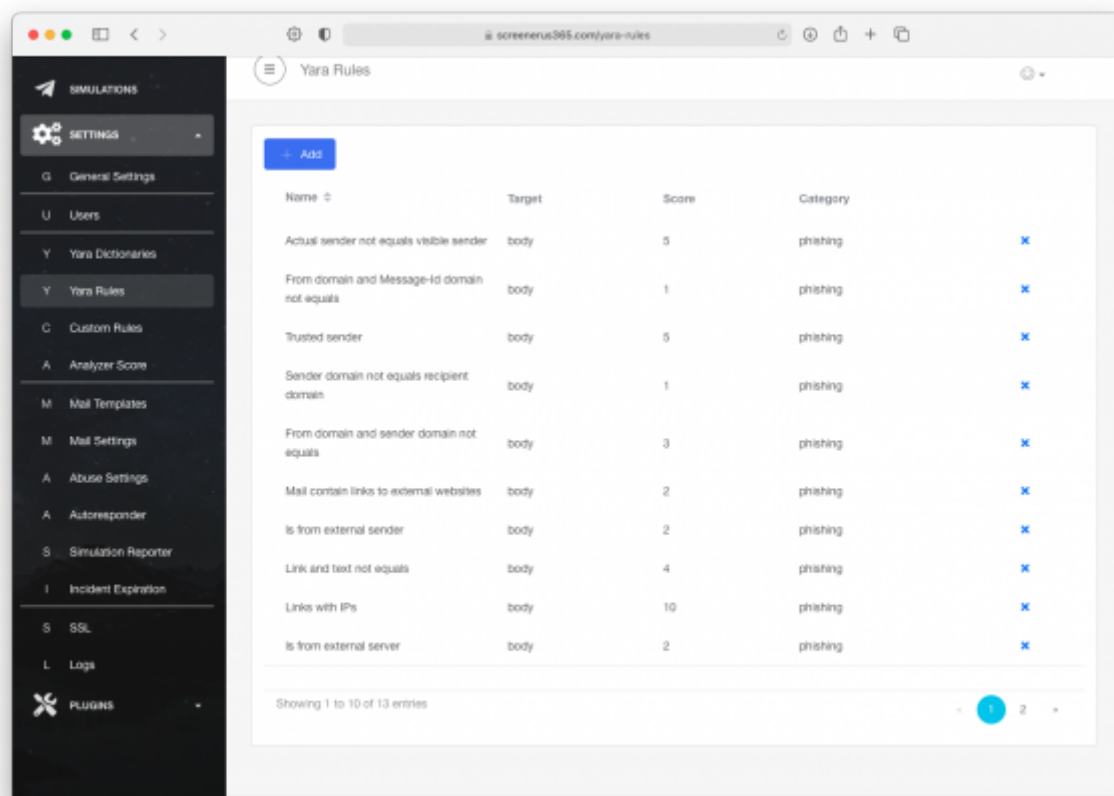
[Set the name and add a domain name of your company.](#)

In my case it is MyCompany.dict and the domain name is lucysecurity.com Set it to be "Enabled" and press the "Save" button.



## Step 2: Create the Yara Rule

Go to Settings → Yara Rules → +Add.



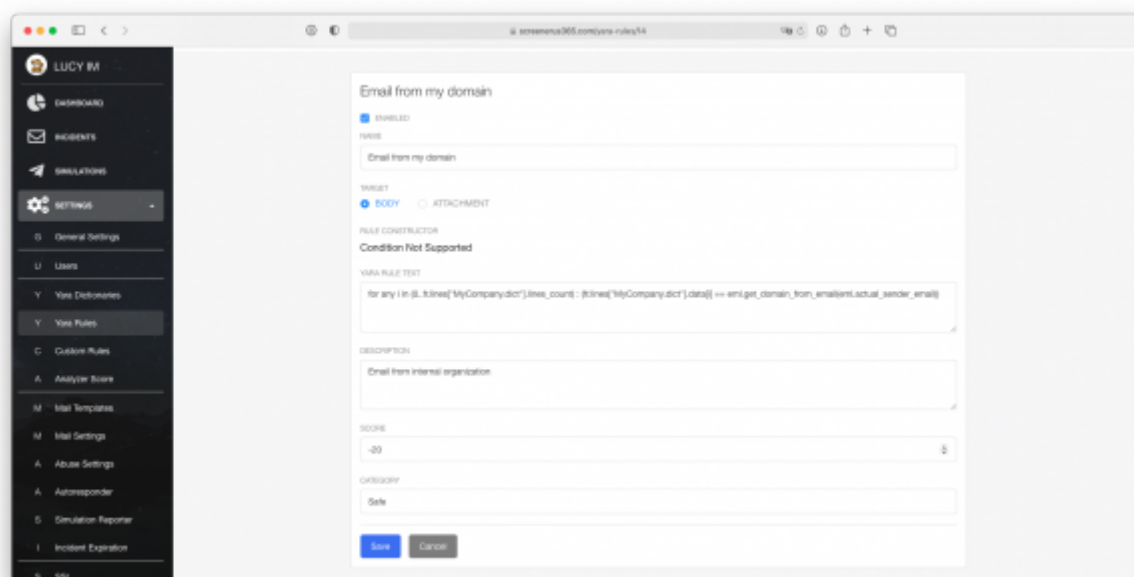
Set it to be Enabled, Set the name, and input the Yara Rule below.

```
for any i in (0..fr.lines["MyCompany.dict"].lines_count) :  
(fr.lines["MyCompany.dict"].data[i] ==  
eml.get_domain_from_email(eml.actual_sender_email))
```

This means that **for** (cycle) any line from 0 to maximum of existing lines in the dictionary (**lines\_count**) there should be a comparison, if a line matches with the domain name in the sender email domain (**eml.get\_domain\_from\_email(eml.actual\_sender\_email)**)  
Where MyCompany.dict is the name of the dictionary.

The important trick, set the score to be a negative value, for example -20.

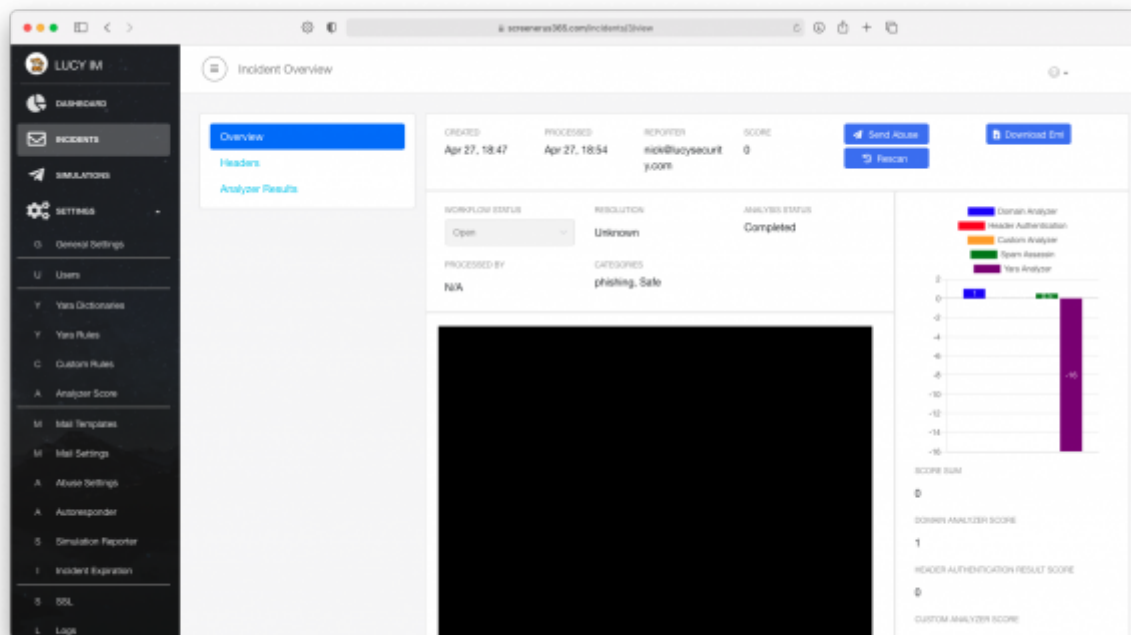
With this configuration set all of the emails from domains that are in the dictionary will be considered as safe and the negative value will cover the false positives. The category is up to you. It'll be assigned to the incident in the overview section.



### Step 3: Review the result

It can be seen that the category is assigned according to the internal rules, including the "safe" one. Preconfigured Yara rule in step 2 gave the incident -20 scores.





Yara Results

Overview Headers Analyzer Results

Domain Yara Header Authentication Custom Rules Spam Assassin UI Analyzer Results

Target	Name	Description	Category	Score
body	is from external server	View	phishing	2
body	Mail contain links to external websites	View	phishing	2
body	Email from my domain	View	Safe	-20

So from now on it is enough to add a domain name into the dictionary we created for it automatically considered as safe.

## Deep into Yara inside of Screener

### EML Module

We use the "eml" module for the Yara engine. It is designed to process eml files. It can be found in the Yara rule example below:

```
eml.get_domain_from_email(eml.headers["message-id"]) !=
```

```
eml.get_domain_from_email(eml.headers["from"])
```

It is possible to build your own complex Yara rules with its models and endpoints available in the table below:

Members	Type	Description
delivered_to	string	Extracts information from the header "Delivered_to"
message_id	string	Extracts information from the header "Message_id"
subject	string	Extracts information from the header "Subject"
date	string	Extracts information from the header "Date"
x_mailer	string	Extracts information from the header "X_mailer"
from	string	Extracts information from the header "From"
to	string	Extracts information from the header "To"
cc	string	Extracts information from the header "CC"
bcc	string	Extracts information from the header "bcc"
actual_sender_email	string	Extracts information from the email regarding the True Sender
visible_sender_email	string	Extracts information about the Visible Sender Email
recipient_email	string	Extracts information regarding the recipient email address
origin_server_hostname	string	Extracts information regarding server hostname
headers	string_dictionary	example: eml.headers["headername"]="custom text"
number_of_domains	integer	Number of domain names in the email
domains	string_array	The domain names in the email
number_of_ip_addresses	integer	Number of IP addresses in the email
ip_addresses	string_array	The IP addresses in the email
number_of_http_links	integer	Number of HTTP links in the email
http_links	struct (link, text)	example: eml.http_links[1].text="domain" and eml.http_links[1].link=" <a href="https://domain.com">https://domain.com</a> ". It can be used together or separately

Functions	Return Type	Example	Description
"capture"	string	Example: eml.capture("hello (world)", "hey hello world yeah") == "world", Example 2: eml.capture("hello (world)") == "world"	Parses a particular text from an email
"get_domain_from_link"	String	Example: eml.get_domain_from_link(" <a href="https://google.com/test">https://google.com/test</a> ") == "google.com" and eml.get_domain_from_link(" <a href="http://amazon.com">http://amazon.com</a> ") == "amazon.com"	Parse a domain name from a link inside of email
"get_domain_from_email"	String	Example: eml.get_domain_from_email("john@gmail.com") == "gmail.com" and eml.get_domain_from_email("joe@live.com") == "live.com"	Parse domain name from the email
"get_ip"	String	eml.get_ip("localhost") == "127.0.0.1" and eml.get_ip("google.com") == "172.217.21.174"	The function returns 1 IP from the resolve request

Functions	Return Type	Example	Description
"domain_registration_days"	Integer	eml.domain_registration_days("google.com") != 0	Gets the amount of days that has passed after the domain registration date

## FR Module

We also use fr (file-read) module that is designed to process dictionaries in the Screener.

Members	Type	Description
files_count	integer	The number of active dictionaries
files	string array	Names of the active dictionaries
lines	struct	<b>fr.lines["Dictionary_name.dict"].lines_count == 10 and fr.lines["Dictionary_name_2.dict"].lines_count == 11</b> Or, it is possible to look for a particular line <b>fr.lines["Dictionary_name.dict"].data[0] == "value_on_1st_string" and fr.lines["Dictionary_name.dict"].data[9] == "value_on_a_string_9" and fr.lines["Dictionary_name.dict"].data[4] == "value_on_a_string_4"</b>

From:

<https://wiki.lucysecurity.com/> - LUCY

Permanent link:

[https://wiki.lucysecurity.com/doku.php?id=yara\\_rules\\_screener](https://wiki.lucysecurity.com/doku.php?id=yara_rules_screener)

Last update: **2021/04/28 10:29**

