Identity Protection Connector

Last updated by | Naser Mohammed | Mar 19, 2023 at 7:18 AM CDT

Alerting on Identity Protection events

Identity protection recommendations are used to monitor identity activities in a subscription. They're designed so that the customer can take proactive measures before the incident takes place or reactive measure to stop an attack attempt.

Alerting to Sentinel

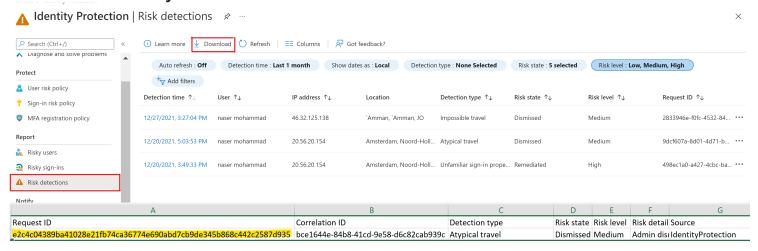
The configuration of the Data Connector is done solely by enabling the Connector with the Connect button. The events coming in from this Data Connector will be stored as part of the SecurityAlert table. Not every single event may be turned into an alert from the Identity Protection end. This depends upon the configuration of the alerts from the Identity Protection Portal. Under Notify – Users at risk detected alerts – the customer may choose under "Alert on user risk level at or above" from alerting at high, medium, or low. The default is set to high. So, if the customer does not see in Sentinel alerts some of the events from Identity Protection, it may be that this setting is filtering such events and not converting these into alerts that will be sent over the Data Connector. In which case they may want to modify this setting to generate alerts for lower severity events.



Verifying if logs were received in Sentinel

We can confirm if certain logs are being sent to Sentinel using the **Request Id** from the CSV export of the Risk Detections:

From Azure AD Identity Protection ==> Risk Detections Download the risk detections:



In the Sentinel Log Analytics Workspace, you can see the same value under VendorOrignalId field:

TimeGenerated [UTC]	7 DisplayName		√ AlertSeverity	∇ Description	
ProviderName	IPC				
VendorName	Microsoft				
··· VendorOriginalId	VendorOriginalId <u>e2c4c04389ba41028e21fb74ca36774e690abd7cb9de345b868c442c2587d935</u>				
SystemAlertId	cf33dbaf-8	212-949a-3b9f-e433	bea75b55		
AlertTvne	Impossible	Travel			

Azure Active Directory Identity Protection Incidents

Incidents ingested from Identity Protection can be ingested in an initially closed state. This can be confirmed by examining the following code: https://msazure.visualstudio.com/One/git/ASI-Cases?
https://msazure.visualstudio.com/One/git/ASI-Cases?
https://msazure.visualstudio.com/One/git/ASI-Cases?
https://msazure.visualstudio.com/One/git/ASI-Cases?
https://msazure.visualstudio.com/One/git/ASI-CaseService%2FCaseCreation%2FCaseCreationLogic.cs&a=contents&version=GBdevelop">https://msazure.visualstudio.com/One/git/ASI-CaseService%2FCaseCreation%2FCaseCreationLogic.cs&a=contents&version=GBdevelop
https://msazure.visualstudio.com/One/git/ASI-CaseService%2FCaseCreation%2FCaseCreationLogic.cs&a=contents&version=GBdevelop
https://msazure.visualstudio.com/One/git/ASI-CaseService%2FCaseCreation%2FCaseCreationLogic.cs&a=contents&version=GBdevelop
https://msazure.visualstudio.com/One/git/ASI-CaseService%2FCaseCreation%2FCaseCreationLogic.cs&a=contents&version=GBdevelop
https://msazure.visualstudio.com/One/git/ASI-CaseService%2FCaseCreation%2FCaseCreationLogic.cs&a=contents&version=GBdevelop
https://msazure.visualstudio.com/One/git/ASI-CaseCreationLogic.cs&a=contents&version=GBdevelop
https://msazure.visualstudio.com/One/git/ASI-CaseCreation
https://msazure.visualstudio.com/One/git/ASI-CaseCreation
https://msazure.vis

```
public const string ResolvedAtSourceCommentText = "Resolved at source";
public const string DismissedAtSourceCommentText = "Dismissed at source";
 . . . . . . .
if (alert.Status == AlertStatus.Resolved)
                                                        Set Case Status To Closed With Reason And Add Comment (new Case, alert Data Model, Resolved At Source Comment To Commen
                                          if (alert.Status == AlertStatus.Dismissed)
                                                        SetCaseStatusToClosedWithReasonAndAddComment(newCase, alertDataModel, DismissedAtSourceComment
                                          }
private void SetCaseStatusToClosedWithReasonAndAddComment(Case @case, AlertDataModel alertDataModel, string co
                            {
                                          // Set case status to closed and closure reason to benign positive
                                          @case.Status = CaseStatus.Closed;
                                          @case.CloseReason = CaseCloseReason.BenignPositive;
                                          //Add a closing comment explaining the closure reason
                                          @case.ClosedReasonText = comment;
                            }
```