Trellix

DATA SHEET

Trellix DRAY

Key Features

- Development platform and documentation included built into the platform
- 120-plus reusable UI component library for building front-end views in JavaScript
- Server Sent Events system
- Plugin framework written in Python
- Event, query, and command inter plugin message bus
- Database model, access, and migration framework
- Sandboxed file system operations
- Built-in email notifications back-end
- Front-end settings framework
- Time-based and interval job scheduling back-end
- Extensible command line interface (CLI) framework
- Quick and easy backup and restore via CLI
- Unified front-end and back-end role-based access control (RBAC)
- User and group access management.
- API-first design with Rest API access to plugins and core platform functionality
- Vault for secrets management
- Vertically scalable to meet workflow demands
- Lightweight footprint
- Deployable in denied, disrupted, intermittent, and limited impact (DDIL) environments

Trellix DRAY is a flexible platform that enables the rapid development of custom security operations workflows and integrations unified under one platform.

The Problem

- Teams interact with multiple security solutions daily.
- The security solutions are provided by different vendors.
- Security operations workflows are separated between solutions.
- There is no unified way to interact with multiple solutions.

The Solution

One platform with infinite securities operations workflow and integration possibilities.

A workflow is a collection of backend plugins and frontend UI views that collectively provide the desired functionality and interactivity for a desired security operation.

Plugins provide logic, API access, data transformations, database and filesystem interactions and integrations.

Views provide the visual user interactions to the plugins and are quickly composable using a large library of reusable components.

The plugins are designed using Python, and views are designed using JavaScript. Both languages were chosen for their flexibility and vast community support as two of the most popular modern programming languages available.

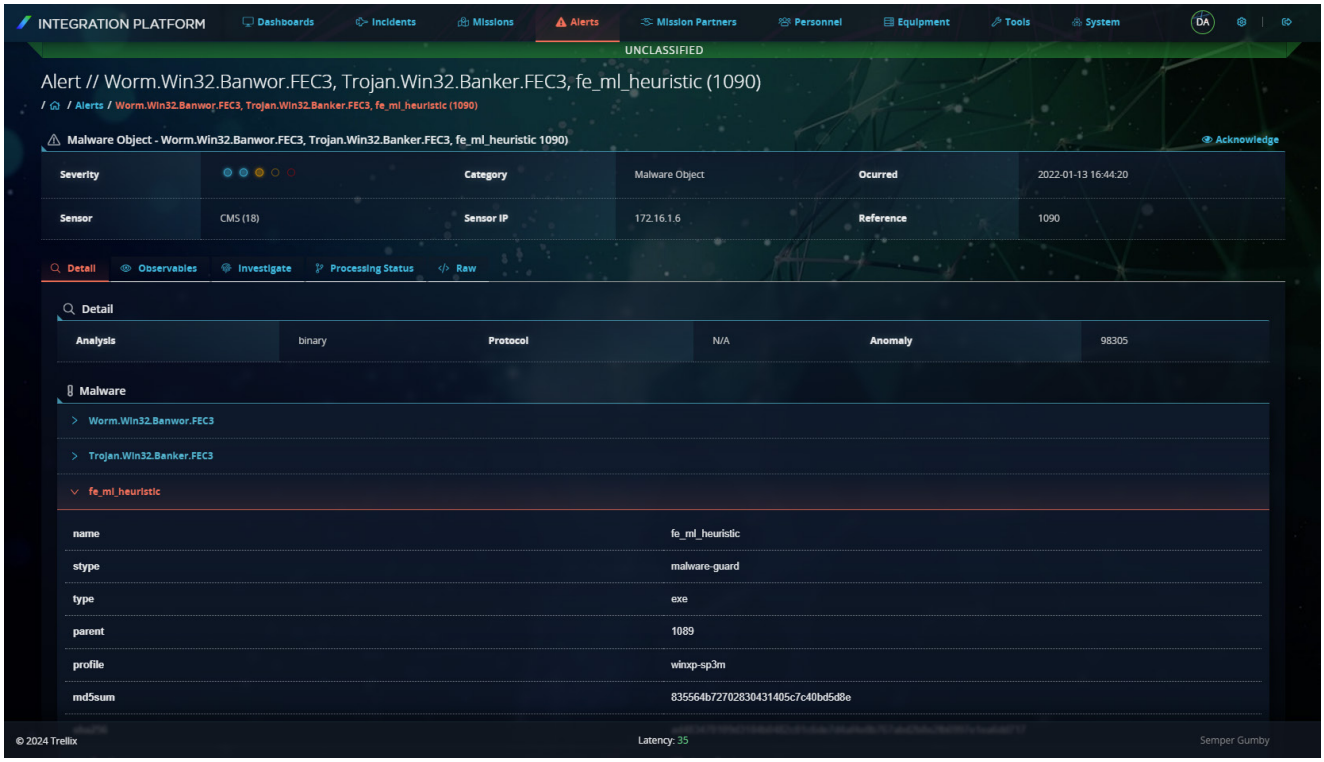

Figure 1: Alerts in Trellix Dray.

The Workflow Concept

A workflow is a collection of server plugins and client UI views that collectively provide the functionality and interactivity for a desired workflow.

Plugins provide logic, API access, data transformations, database and filesystem interactions and integrations.

Views provide the visual interactions with data from the server plugins and are quickly composable using a large library of reusable components.

Plugins

Plugins are written in Python and come in three flavors.

Core Plugins

Core plugins provide the base features of the platform and are always available. They include:

- Authentication
- Configurations
- Directed Acyclic Graphs
- Development
- Events
- Filesystem
- Health
- Logs
- Notifications
- Ping
- Plugin Manager
- Scheduler
- Settings
- Statistics
- System
- Tags
- Tools
- Users
- Vault

Shared Plugins

Shared plugins are workflow-agnostic and provide useful tools. They include:

- ChatGPT
- CyberChef
- Draw
- Knowledge Base

Workflow Plugins

Workflow plugins provide the RESTful API or data interactions for a custom workflow and are developed by the user.

Views

Views are written in JavaScript and React and can be quickly composed using an extensive library of 120-plus reusable components.

Core Views

Core views provides the foundation for menus, navigation, layout, and built-in views for interacting with core plugins for system management.

Workflow Views

Workflow views provide the interactions for users to display and interact with data provided by the workflow plugins through the REST API.

Architecture

The platform is a headless architecture consisting of a web-based client and a server.

Client

The client is a performance-focused extensible Single Page Application (SPA) that is implemented completely on the client side.

The client provides core features such as protected data access to the server APIs, a security model that is in lock step with the server, and an extensive library of utilities and reusable components to compose new workflows.

Server

The server provides a robust plugin architecture to implement new APIs, data transformations, or integrations with other security solutions for custom workflows.

The server was designed with an API-first approach with the REST API forming the foundation of the platform that enables integration flexibility where data can be both consumed and created not just by the client, but also by other security solutions.

Beyond the flexible plugin system, the server provides core functionality for database access, sandboxed filesystems, CLI interactions, and a scheduling engine.