Velociraptor

Prepared By:
Kazim Ali Obad

Supervisor:

Anmar Mohammed

MOHAMMED .B. HASSAN

Contents

1 What is Velociraptor?

1.1 Definition

Velociraptor is an advanced, open-source endpoint monitoring and Digital Forensics and Incident Response (DFIR) platform. Despite being open source, it is a professional-grade tool used in real enterprise environments for serious investigations.

1.2 Core Capabilities

- Endpoint Monitoring: Continuously monitors all connected endpoint machines in real time.
- Incident Response (IR): When an attack or anomaly is detected, Velociraptor allows you to immediately push artifact collection tasks to the affected machine.
- Digital Forensics: Conduct thorough forensic investigations by collecting structured evidence (artifacts) from any endpoint, remotely and at scale.
- Adversary Hunting: Hunt across your entire fleet for traces of advanced attackers using VQL queries to search for specific indicators.
- Malware Analysis Support: Collect running process data, perform memory dumps, and gather the raw evidence that malware analysts need.

1.3 VQL The Velociraptor Query Language

VQL is the built-in query language that powers Velociraptor's artifact system. It is a framework for writing queries that collect, filter, and analyze endpoint data. Every prebuilt artifact in Velociraptor is written in VQL, meaning the platform is fully customizable. If the built-in artifacts do not cover your exact scenario, you can write your own.

2 Understanding Artifacts

2.1 The Real-World Analogy

The concept of an artifact is the foundation of all digital forensics work. To understand it properly, start with a physical analogy. When a crime occurs and an investigator arrives at the scene, everything they find that can tell them what happened is called evidence fingerprints, footprints, broken items, and so on. In digital forensics, the exact same concept applies. Every action taken on a computer by a legitimate user or by an attacker leaves traces behind.

2.2 Categories of Digital Artifacts

The following are the primary categories of digital artifacts relevant to Windows endpoint investigations:

- **Registry Keys:** The Windows Registry stores configuration data for the OS and applications. Attackers frequently modify registry keys to establish persistence.
- **Running Processes:** The list of active processes at any given moment. Malicious software trojans, keyloggers, C2 agents must run as a process.
- **Files on Disk:** Recently downloaded, recently modified, or files in unusual locations and temporary directories.
- **Network Activity:** Every network connection leaves a trace IP addresses contacted, DNS queries sent, open ports, and active sessions.
- **Attack Patterns and Privilege Escalation Indicators:** Specific system behaviors matching known attack techniques.
- User Activity: Login/logout events, command history, recently opened documents, browser history.

2.3 How Velociraptor Handles Artifacts

Velociraptor ships with a large library of prebuilt artifact definitions organized by category, operating system, and investigation type. The key capability is scale: Velociraptor can collect artifacts from one machine or from thousands of machines simultaneously using the same interface and the same commands.

> ***NOTE: An artifact is any trace left on a digital system that can serve as evidence. Collecting artifacts is the core activity of digital forensics and incident response.***

3 Velociraptor Architecture

3.1 Server–Client Model

Velociraptor operates on a server–client (server–agent) model. There are two components:

- **Velociraptor Server:** The central management point. It runs the web-based administration interface, manages all connections from agents, stores collected artifacts, and dispatches investigation tasks.
- **Velociraptor Client (Agent):** A lightweight agent installed on each endpoint you want to monitor and investigate. The agent receives instructions from the server, executes artifact collection locally, and returns results.

3.2 Network Ports and Communication

- **Port 8089 (HTTPS):** The administration portal the port you use to access the Velociraptor web interface as an administrator.
- **Port 8000 (HTTPS/TLS):** The agent communication channel all data exchange between the server and client agents flows through this port.

- **TLS Encryption:** All communication between the server and clients is TLS encrypted. This protects against interception.

> ***NOTE: Always use https:// to access the web interface. The server is configured to reject plain HTTP connections.***

3.3 Deployment Types

- **Self-hosted / Instant Mode:** The server and client run on the same single machine. This is the lab and learning setup.
- **Separate Server and Clients:** A dedicated server machine runs the Velociraptor server. Client agents are installed separately on each endpoint.
- **Cloud Deployment:** The Velociraptor server runs on a cloud virtual machine (AWS, Azure, GCP, etc.). On-premise clients connect outbound to the cloud server.

4 Docker Concepts and Purpose

4.1 The Problem Docker Solves

A persistent problem in software deployment is dependency conflicts. An application requires specific versions of libraries, runtimes, and OS components. Docker solves this by packaging the software together with its entire runtime environment the operating system base, every library, every configuration into a single portable unit. This eliminates the 'it works on my machine' problem entirely.

4.2 Docker Concepts

Image: A Docker Image is a static, read-only snapshot of a configured environment. It is the blueprint. For example, an Ubuntu image with Velociraptor pre-installed.

Container: A Container is a running instance of an image. When Docker starts a container from an image, it creates an isolated runtime environment. Multiple containers can run from the same image simultaneously.

Docker Compose: A tool for defining and running multi-container applications using a YAML configuration file (usually docker-compose.yml). Docker Compose automates the startup, networking, and configuration of all containers in a project with a single command.

5 Installing and Deploying Velociraptor

5.1 Getting the Files

Velociraptor is open source and available from its official GitHub repository. There are two approaches to deployment:

- **Docker-based deployment:** The docker-compose.yml file handles all configuration and startup automatically.
- **Manual / source deployment:** Clone the repository, generate a server configuration file (server.config.yaml), and run the binary directly.

5.2 Step-by-Step Docker Installation

Step 1 Install Docker Engine:

```
$ sudo apt install docker.io -y
```

Step 2 Start and enable the Docker service:

```
$ sudo systemctl enable --now docker
```

Step 3 Verify Docker is running :

```
$ sudo systemctl status docker
```

Step 4 Test Docker :

```
$ sudo docker ps
```

Step 5 Navigate to the Velociraptor Docker project directory:

```
$ cd ~/Desktop/velociraptor-docker
```

Step 6 Verify the project files are present (you should see docker-compose.yaml, Dockerfile, entrypoint, README.md, .env):

```
$ Ls
```

Step 7 Start Velociraptor using Docker Compose:

```
$ sudo docker compose up -d
```

Step 8 Verify the container is running:

```
$ docker ps
```

Step 9 View container logs (optional, useful for troubleshooting):

```
$ docker compose logs -f
```

Step 10 Open the Velociraptor Web Interface in your browser:

```
$ https://localhost:8889
```

> *NOTE: Always type https:// not http://. The server rejects plain HTTP. Your browser will show a certificate warning because the certificate is self-signed.*

5.3 Logging In

Use the default credentials: Username: admin / Password: admin

5.4 The Web Interface After Login

After logging in, the main dashboard is displayed. Key areas:

- **Clients / Devices:** Lists all machines currently connected as agents. On a fresh install this will be empty no clients have been added yet.
- **Collected Artifacts:** Where artifact collection results are stored and viewed.
- **Hunt Manager:** For running artifact collections across many clients at once.
- **Server Information:** Deployment health and configuration overview.

6 Adding a Client Agent (Windows)

6.1 How Client Configuration Works

Before an agent can connect to the Velociraptor server, it needs a configuration file that tells it where the server is and how to authenticate. This configuration file contains:

- The server address and port
- TLS certificates and cryptographic keys for encrypted communication
- Authentication tokens proving the agent is legitimate

This approach ensures the TLS connection is cryptographically sound. The server embeds its own certificate information into the client config so the client can verify it is talking to the correct server.

6.2 Entering the Docker Container

Since Velociraptor runs inside a Docker container, you need to enter the container to access the binary and generate the client config:

```
$ sudo docker exec -it velociraptor bash
```

Explain:

- **docker exec** execute a command inside a running container
- **-i** interactive mode (keeps standard input open)
- **-t** allocates a terminal (pseudo-TTY) so you get a proper shell prompt
- **velociraptor** the container name
- **bash** opens a bash shell inside the container

You can also write the long form:

```
$ sudo docker exec --interactive --tty velociraptor bash
```

When you are done working inside the container, exit with:

```
$ exit
```

6.3 Finding the Velociraptor Binary Inside the Container

```
$ find / -name velociraptor 2>/dev/null
```

The binary is typically located at /opt/velociraptor/velociraptor. Navigate there directly:

```
$ cd /opt/velociraptor
```

6.4 Finding the Server Configuration File

```
$ find / -name 'server.config.yaml' 2>/dev/null
```

Or search for all YAML files:

```
$ find / -name '*.yaml' 2>/dev/null
```

6.5 Generating the Client Configuration File

With both the binary and server config located, generate the client configuration:

```
$ ./velociraptor config client -c server.config.yaml > client.conf.yaml
```

This command reads the server configuration (which contains the TLS certificates and server identity) and produces a new file called client.conf.yaml. Verify the file was created:

```
$ ls -la
```

6.6 Transferring the Configuration File to Windows

Method 1 Python HTTP Server: Start a simple HTTP server from inside the directory containing the client config:

```
$ python3 -m http.server 4444
```

Then on the Windows machine, open a browser and go to

Error! Hyperlink reference not valid.. A directory listing will appear.

Click client.conf.yaml to download it. To find the Linux machine's IP:

```
$ ifconfig
```

Method 2 Docker Copy: Copy the file directly from the container to the host machine:

```
  sudo docker cp
$ velociraptor:/opt/velociraptor/client.conf.yaml
  ./client.conf.yaml
```

6.7 Downloading the Windows Velociraptor Executable

In addition to the configuration file, the Windows machine needs the Velociraptor executable. Download it from the official GitHub Releases page:

- Go to the Releases section of the Velociraptor GitHub repository
- Under the latest release, find the Windows executable
- Download the amd64 (64-bit) .exe file

6.8 Installing the Windows Agent as a Service

Step 1 Open Administrator Command Prompt and navigate to Downloads:

```
$ cd C:\Users\<your_username>\Downloads
```

Step 2 Verify both files are present (the Velociraptor .exe and client.conf.yaml):

```
$ dir
```

Step 3 Install the service:

```
$ velociraptor.exe service install --config
client.conf.yaml
```

On success, output will confirm the service was installed. The agent immediately starts and begins attempting to connect to the Velociraptor server.

Step 4 Verify the connection: Go back to the Velociraptor web interface and refresh the Clients page. Within a few seconds, the Windows machine will appear in the client list. To confirm you are looking at the correct machine, run the following on the Windows machine and cross-reference with the Velociraptor interface:

```
$ whoami
```

7 Collecting Artifacts from an Endpoint

7.1 The Investigation Scenario

The practical work begins with a realistic scenario: a machine in your organization may have been compromised. You do not know what happened. You need to investigate without alerting the user or the attacker, and without physically touching the machine a Windows Server 2019 machine is used as the target client.

7.2 Navigating to Artifact Collection

- In the Velociraptor web interface, go to the Clients list
- Click on the target machine
- Inside the client view, go to the Collected Artifacts section
- Click the button to add a new artifact collection
- A search interface appears you can browse by category or search by name

7.3 General System Information

The first artifact to run on any new client is a **general information collector**. This establishes a baseline of what you are looking at before any deeper investigation.

- **What it collects:** OS version, hostname, hardware specifications, user accounts, installed applications, uptime
- **Why it matters**: Before you investigate anything specific, you need to know what kind of system you are dealing with, who the users are, and what software is installed

7.4 Downloaded Files Analysis

After a machine is compromised, attackers almost always download something onto it tools, scripts, payloads, loaders. Finding these downloaded files is a high-priority task.

Artifact to use:

```
$ Windows.Analysis.DownloadedFiles
```

What it returns:

- The name and full file path of every downloaded file
- The source URL or network path the file was downloaded from
- The timestamp of the download event
- Files transferred via copy-paste are also captured not just browser downloads

Running it:

- Add a new artifact collection
- Search for and select Windows.Analysis.DownloadedFiles
- Click Launch and wait for the collection to complete on the remote client
- Open the Results tab

7.5 Running Process List

The process list is one of the most important live artifacts in incident response. Every piece of malicious software trojans, C2 agents, keyloggers, ransomware must run as a process.

Artifact to use:

```
$ Windows.System.Pslist
```

What it returns for each process:

- Process Name the executable name
- Process ID (PID) unique numeric identifier
- Parent Process ID (PPID) which process launched this one
- Username which user account the process is running under
- Command Line the exact command used to start the process
- Executable Path full path on disk to the executable

What to look for: The critical field to examine is Username. On a properly configured Windows system, core system processes run as SYSTEM. If a process looks like a system component but is running under a regular user account, that is a red flag.

> ***NOTE : A process running under a non-SYSTEM, non-service user account that has a name resembling a system component is a strong indicator of compromise. Take the suspicious file path and submit it to a multi-engine antivirus scanning service such as VirusTotal for immediate cross-referencing.***

7.6 Other Available Pre-Built Artifacts

- Windows.System.Users All user accounts on the machine, including hidden accounts
- Windows.Network. Open connections, listening ports, recent network sessions, DNS cache
- Windows.Forensics. Persistence mechanisms, scheduled tasks, services, event log analysis
- Windows.Registry. Query specific registry keys known to be used by attackers for persistence
- Windows.Detection. Pre-written detections for known attacker techniques and attack patterns
- Linux equivalents All major artifact categories have corresponding Linux versions

Custom artifacts can be written in VQL for any investigation .

8 Memory Dumping Process Memory Analysis

8.1 What is a Memory Dump?

When a suspicious process is identified, the next investigation step is to capture what that process had in memory at a specific point in time. Think of it like taking a forensic photograph of a running process. At the moment you trigger the dump, everything stored in that process's memory space is saved to a file with the extension .DMP.

This .DMP file is then given to a malware analyst or memory forensics specialist who examines it using tools such as Volatility, Rekall, or WinDbg. From the memory dump, the analyst can determine:

- What code was executing inside the process
- Whether the process was injected with malicious code by another process (process injection)
- What network connections the process was managing
- What strings were present in memory URLs, commands, credentials, file paths
- Whether shellcode or an unpacked malware payload was present in memory

8.2 Why Memory Dumping is Critical

Many modern attacks use fileless malware or process injection. The attacker injects malicious code directly into the memory of a legitimate process no files are written to disk. This means disk-based detection misses it entirely. The memory dump reveals what the process was actually doing at runtime, regardless of what the disk-based view shows.

8.3 Performing a Memory Dump in Velociraptor

Step 1 Have the target process name ready from the process list

Step 2 In the client artifact collection interface, add a new collection and search for:

```
$ Windows.Memory.ProcessDump
```

Step 3 Configure Parameters:

- Click Configure Parameters
- Find the ProcessName field
- Remove any placeholder text
- Enter the exact name of the suspicious process as it appeared in the process list

Step 4 Click Launch. Velociraptor pushes the task to the client. The agent finds the running process by name, dumps its memory to a file, and uploads the dump back to the server.

Step 5 When the collection completes, open the Results tab. The .DMP file is available for download this is your raw forensic evidence.

8.4 Connecting Memory Dumps to the Broader Investigation

Each artifact you collect answers a different question. Downloaded files tell you what was brought onto the machine. The process list tells you what is running. The memory dump tells you what a suspicious process was actually doing. Together, these three artifacts form the foundation of most incident response investigations.

9 The DFIR Investigation Workflow

9.1 Overview

Everything covered in these sessions maps to a standard Digital Forensics and Incident Response methodology. Understanding where each technique fits in the broader workflow is essential for working effectively in a real incident.

9.2 Acquisition

Acquisition is the evidence collection phase. It must be done first, correctly, and carefully. The goal is to gather all relevant artifacts from affected endpoints before any cleanup or response action that could alter the evidence state.

In practice with Velociraptor:

- Deploy agents on all endpoints that may be involved
- Collect artifact sets files, processes, network data, registry data, event logs
- Perform memory dumps of suspicious processes
- Document every collection action with timestamps

9.3 Processing

Raw collected data must be processed before it can be analyzed. Processing involves:

- Cross-referencing artifacts match process PIDs against network connection records to see what a suspicious process was communicating with
- Running VQL queries against collected data to surface specific indicators
- Filtering noise and focusing on anomalies

9.4 Analysis

The analysis phase is where forensic expertise is applied. With processed data in hand:

- Reconstruct the attack timeline what happened first, what followed
- Map attacker techniques to frameworks like MITRE ATT&CK
- Analyze memory dumps for injected code, shellcode, stolen credentials
- Determine the scope how many machines were affected, how far did the attacker get
- Identify the initial access vector how did the attacker first get in

9.5 Containment and Response

Once the scope and nature of the incident are understood, containment begins. This phase which includes:

- Isolating affected machines from the network without tipping off the attacker
- Blocking identified C2 addresses and malicious domains
- Removing persistence mechanisms the attacker installed
- Resetting compromised credentials
- Running the incident response meeting who is involved, what decisions need to be made, who communicates what to management

10 Working Inside Docker Containers

10.1 Entering and Exiting the Container

```
$ sudo docker exec -it velociraptor bash
```

Options: -i (interactive) keeps stdin open; -t (tty) allocates a pseudo-terminal; velociraptor is the container name (verify with docker ps); bash is the shell to open.

Exit the container when finished:

```
$ exit
```

10.2 Searching for Files Inside the Container

Find the Velociraptor binary:

```
$ find / -name velociraptor 2>/dev/null
```

Find all YAML configuration files:

```
$ find / -name '*.yaml' 2>/dev/null
```

10.3 Copying Files Between the Container and Host

Copy from container to host:

```
$ sudo docker cp <container_name>:<path_in_container>
  <path_on_host>
```

Example copy client config to current directory:

```
$ sudo docker cp velociraptor:/opt/velociraptor/client.conf.yaml ./client.conf.yaml
```

Copy from host to container:

```
$ sudo docker cp ./localfile.txt velociraptor:/opt/velociraptor/localfile.txt
```

10.4 YAML Configuration Files

- **server.config.yaml** The server configuration file. Contains TLS certificates, the data store path, API settings, and the server's identity. Lives inside the Docker container.
- **client.conf.yaml** The client configuration. Generated by the server from the server config. Contains the server's public certificate and connection address.

For a local lab where the server and client are on the same network, the client config needs to point to the correct server IP. Open client.conf.yaml and set the correct hostname and port:

```
$ API_config:   hostname: 192.168.168.38   # Replace with your server's actual IP   port: 8000
```

11 Real-World Enterprise Deployment Context

11.1 What a Production Deployment Looks Like

In a real organization, a Velociraptor deployment looks like this:

- A dedicated server physical or virtual running the Velociraptor server process. In larger organizations this may be a cloud VM to allow centralized access across multiple office locations.
- Client agents installed on every endpoint: workstations, servers, critical systems.
- Integration with the organization's SIEM so that events and alerts from Velociraptor feed into the centralized security monitoring system.

11.2 When To Use Velociraptor

- **Suspected Compromise:** An alert fires from an AV, a SIEM rule, a network IDS. Without physically touching the machine or alerting the user, we immediately deploy artifact collection.
- **Proactive Threat Hunting:** No specific alert has triggered, but you run periodic hunts across all endpoints searching for known indicators of compromise.
- **Post-Incident Forensic Investigation:** After containing an incident, the forensic investigation begins. Velociraptor is used to reconstruct the full attack timeline.
- **Compliance and Verification:** Auditing endpoint state checking that no unauthorized software is installed, no unusual accounts exist, no unexpected services are running.

15 Summary

Velociraptor is a professional, open-source DFIR platform for endpoint monitoring, incident response, and digital forensics. It goes beyond standard EDR by enabling deep forensic artifact collection, memory dumping, VQL-based hunting, and enterprise-scale investigations.

Artifacts are digital traces left on computer systems. They are the evidence base for every investigation the same way physical traces are the evidence base in a crime scene investigation.

VQL is Velociraptor's built-in query language. Prebuilt artifacts cover most common investigation scenarios. It allows Custom artifacts to be written for anything not covered.

The architecture is server–client. The server runs on ports 8089 (admin) and 8000 (agents). All communication is TLS-encrypted. The admin interface requires https://.

Client configuration files are server-generated never created manually. They contain the cryptographic material needed for TLS authentication.

Memory dumping captures what a process had in memory at a specific moment. This is critical for detecting fileless malware and process injection attacks that leave no disk artifacts.

The DFIR workflow flows through: Acquisition → Processing → Analysis → Containment and Response. Velociraptor handles Acquisition. Specialized tools handle the rest.