# Osquery for Endpoint Visibility and Threat Analysis

## IIT Jammu: Sec Workshop
## 22-23rd February 2023

Prof. Manjesh K. Hanawal, IIT Bombay
Arjun Sable, IIT Bombay
Yogesh Jadhav, IIT Bombay

# Cyber security and threats

CERT-IN annual report 2021

| Security Incidents | 2021 |
|---|---|
| Phishing | 523 |
| Unauthorized Network Scanning/ Probing | 432057 |
| Vulnerable Services | 728276 |
| Virus/ Malicious Code | 209110 |
| Website Defacements | 27408 |
| Website Intrusion & Malware Propagation | 1489 |
| Others | 3946 |
| Total | **1402809** |

*Table 2: Breakup of Security Incidents handled*

News / India / AIIMS cyber attack: Delhi Police seeks information on Chinese hackers through Interpol

**AIIMS cyber attack: Delhi Police seeks information on Chinese hackers through Interpol**

In a letter to the CBI, Delhi Police has asked whether the Chinese IP addresses detected in the AIIMS cyber attack were being used by a company or an individual.

DNA हिंदी में पढ़ें

Top News | Russia-Ukraine crisis LIVE: Europe warns airlines not to fly over or near Ukraine

Latest News
Explainers
India
Entertainment
Business
Cricket

Home » World

**National emergency in Ukraine after cyber-attacks bring down many websites**

State of emergency allows authorities to impose restrictions on movement, block rallies and ban political parties in the interest of national security

Reported By: DNA | Edited By: DNA Web Team |Source: DNA Web Desk |Updated: Feb 24, 2022, 07:43 AM IST

**Cyber crimes in India caused Rs 1.25 lakh crore loss in 2019: Official**

*Industry expert said that there are only a few Indian companies who are making some of the cyber security products and there is a big vacuum in the sector.*

Written by PTI
October 20, 2020 8:48:17 pm

Image sources: Internet

# Cyberattacks in India (Top 7 recent attacks)

1. **Razorpay:** Lost Rs. 7.3 crores in unauthorized transactions
2. **Amazon, Flipkart, Airtel & JioMart:** Data of 10 crore users stoles due to cyberattack in Juspay
3. **Oil India Ltd:** Ransomware attack demanded Rs. 58 Crores
4. **Tech Mahindra:** Ransomware attack resulted in loss of Rs. 35 Crores
5. **Mobikwik:** 10 Crore user data put of dark web
6. **Air India:** Reportedly 45 Lac passenger data hacked
7. **Domino's:** 18 cores order are put in public
8. **AIIMS:** ransomware attacks

# Cyberattacks in news

vajra

### FEBRUARY

A water treatment plant in Oldsmar, Florida was compromised when an attacker attempted to poison the water supply.

CD Projekt Red was attacked by HelloKitty ransomware.

### MARCH

Channel Nine in Australia had broadcasts disrupted by cyberattacks.

University of Highlands and Islands was attacked with Cobalt Strike.

CNA Insurance was attacked by Evil Corp.

Buffalo Public Schools in New York were attacked with ransomware.

Microsoft Exchange Servers were attacked by HAFNIUM.

### APRIL

The Houston Rockets basketball team (NBA) was attacked by Babuk.

### MAY

Colonial Pipeline was attacked by DarkSide.

AXA was attacked by Avaddon.

Brenntag (chemical distributor) was attacked by DarkSide.

Acer was attacked by REvil.

JBS Foods was attacked by REvil.

Ireland's Health Service Executive (HSE) was attacked by Conti.

### JULY

Ransomware attacks were launched in Chile, Italy, Taiwan, and the U.K. by the LockBit threat group.

Kaseya suffered a supply chain attack from REvil.

### NOVEMBER

The Robin Hood trading platform was breached and information on seven million user accounts was taken.

### DECEMBER

Log4j vulnerability revealed and exploited by multiple threat actors.

Source: The BlackBerry 2022 Threat Report

# Supply Chain Attacks

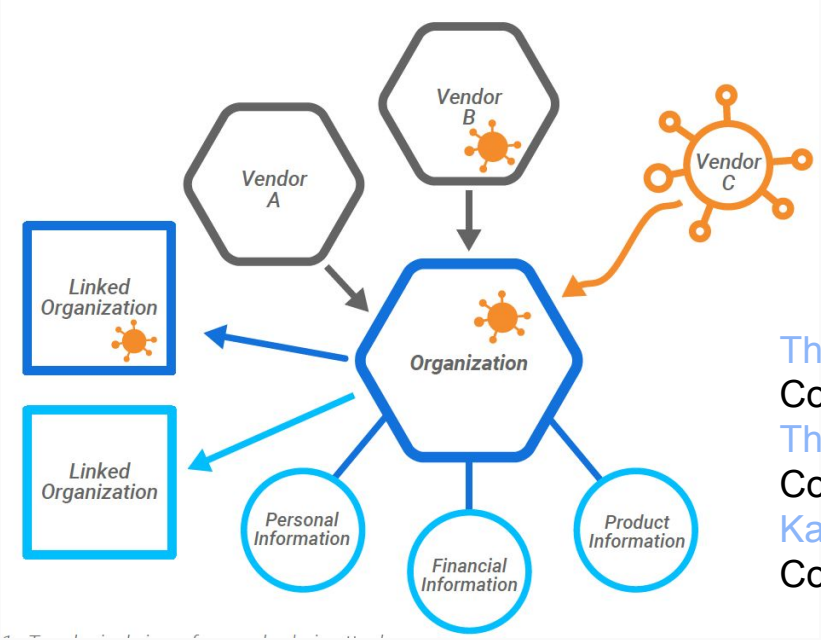



Image: The BlackBerry 2022 Threat Report

- Attackers look for path of least resistance
- Supply chain represent the latest tradecraft
- Trust is exploited!

The NotPetya ransomware attack (2017):
Compromised the Ukrainian tax software MEDoc
The SolarisWinds (2020):
Compromised Orion IT Management and monitoring software
Kaseya (2021):
Compromised Virtual System/Server Administration (VSA) server

# Challenges in Detecting Malwares

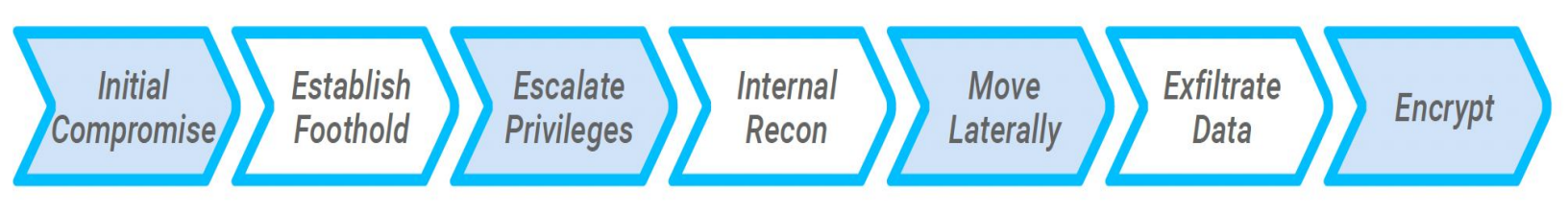
- Evade Antivirus: Changing behavior
- Evade Static analysis: Use obscure programming language (Go, Nim, Rust)
- Evade dynamic analysis: Can detect sandboxed environment
- Hide deep inside: Software getting complex with too many dependencies
- Adversarial training can defeat security tools
- Appear genuine: Emergence of cloud and IoT make it easy to hide and propagate

source: TechRepublic

Way forward: Good network visibility and behavioral analysis can help

# Attack Life Cycle

1. Advanced Persistent Threat  group (low-and-slow)
2. Ransomware attacks (smash and grab)




The attackers users certain Tactics, Techniques and Procedures (TTP) are launch attacks

# MITRE ATT&CK Framework

## Linux Matrix

Below are the tactics and techniques representing the MITRE ATT&CK® Matrix for Enterprise. The Matrix contains information for the Linux platform.

View on the ATT&CK® Navigator

Version Permalink

layout: flat | show sub-techniques | hide sub-techniques | help

| Initial Access | Execution | Persistence | Privilege Escalation | Defense Evasion | Credential Access | Discovery | Lateral Movement | Collection | Command and Control | Exfiltration | Impact |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 techniques | 8 techniques | 16 techniques | 11 techniques | 22 techniques | 15 techniques | 21 techniques | 7 techniques | 14 techniques | 16 techniques | 8 techniques | 13 techniques |
| Drive-by Compromise | Command and Scripting Interpreter (4) | Account Manipulation (1) | Abuse Elevation Control Mechanism (2) | Abuse Elevation Control Mechanism (2) | Adversary-in-the-Middle (2) | Account Discovery (2) | Exploitation of Remote Services | Adversary-in-the-Middle (2) | Application Layer Protocol (4) | Automated Exfiltration | Account Access Removal |
| Exploit Public-Facing Application | Exploitation for Client Execution | Boot or Logon Autostart Execution (2) | Boot or Logon Autostart Execution (2) | Debugger Evasion | Brute Force (4) | Application Window Discovery | Internal Spearphishing | Archive Collected Data (3) | Communication Through Removable Media | Data Transfer Size Limits | Data Destruction |
| External Remote Services | Inter-Process Communication | Boot or Logon Initialization Scripts (1) | Boot or Logon Initialization Scripts (1) | Deobfuscate/Decode Files or Information | Credentials from Password Stores (3) | Browser Bookmark Discovery | Lateral Tool Transfer | Audio Capture | Data Encoding (2) | Exfiltration Over Alternative Protocol (3) | Data Encrypted for Impact |
| Hardware Additions | Native API | Browser Extensions | Create or Modify System Process (1) | Execution Guardrails (1) | Exploitation for Credential Access | Debugger Evasion | Remote Service Session Hijacking (1) | Automated Collection | Data Obfuscation (3) | Exfiltration Over C2 Channel | Data Manipulation (3) |
| Phishing (3) | Scheduled Task/Job (3) | Compromise Client Software Binary | Escape to Host | Exploitation for Defense Evasion | Forge Web Credentials (1) | File and Directory Discovery | Remote Services (2) | Clipboard Data | Dynamic Resolution (3) | Exfiltration Over Other Network Medium (1) | Defacement (2) |
| Supply Chain Compromise (3) | Software Deployment Tools | Create Account (2) | Event Triggered Execution (3) | File and Directory Permissions Modification (1) | Input Capture (3) | Network Service Discovery | Software Deployment Tools | Data from Information Repositories | Encrypted Channel (2) | Exfiltration Over Physical Medium (1) | Disk Wipe (2) |
| Trusted Relationship | System Services | Create or Modify System Process (1) | Exploitation for Privilege Escalation | Hide Artifacts (7) | Modify Authentication Process (2) | Network Share Discovery | Taint Shared Content | Data from Local System | Fallback Channels | Exfiltration Over Web Service (2) | Endpoint Denial of Service (4) |
| Valid Accounts (3) | User Execution (2) | Event Triggered Execution (3) | Hijack Execution Flow (1) | Hijack Execution Flow (1) | Multi-Factor Authentication Interception | Network Sniffing | | Data from Network Shared Drive | Ingress Tool Transfer | Scheduled Transfer | Firmware Corruption |
| | | External Remote Services | Process Injection (3) | Impair Defenses (5) | Multi-Factor Authentication Request Generation | Password Policy Discovery | | Data from Removable Media | Multi-Stage Channels | | Inhibit System Recovery |
| | | Hijack Execution Flow (1) | Scheduled Task/Job (3) | Indicator Removal (7) | Network Sniffing | Peripheral Device Discovery | | Data Staged (2) | Non-Application Layer Protocol | | Network Denial of Service (2) |
| | | Modify Authentication Process (2) | Valid Accounts (3) | Masquerading (5) | OS Credential Dumping (2) | Permission Groups Discovery (2) | | Email Collection (1) | Non-Standard Port | | Resource Hijacking |
| | | Pre-OS Boot (2) | | Modify Authentication Process (2) | Steal or Forge Authentication Certificates | Process Discovery | | Input Capture (3) | Protocol Tunneling | | Service Stop |
| | | Scheduled | | Obfuscated Files or Information (8) | | Remote System Discovery | | Screen Capture | Proxy (4) | | System Shutdown/Reboot |
| | | | | Pre-OS Boot (2) | | Software Discovery (1) | | | Remote Access Software | | |
| | | | | Process Injection (3) | | System Information Discovery | | | | | |
| | | | | Reflective Code Loading | | System Location Discovery (1) | | | | | |

# Good visibility of system logs:

Quality logs are important to to understand and detect threats:

- Process events
- File events
- Socket/Network events
- Hardware events
- Etc.

Default Logging systems

- Linux: Auditd
- Windows: Sysmon and Event Tracer for Windows (ETW)

Windows

- **Collection of required information from endpoints**
- **Collection of Logs to a central location**
- **Contextualization of logs and system state**

# Contextualization of logs

- **Collection of required information from endpoints**
- **Collection of logs to a central location**
- **Contextualization logs and system state**

- Collection of required information from endpoints in the real time.
- Contextualize collected data.
- Customization of data to be collected on the fly.

Proprietary agents that hook kernel for real time logging are available

- FortiEDR
- Singularity by Sentinel
- Carbon black

# Osquery: Endpoint visibility

A data collection agent that is lightweight, configurable, easily accessible,

- Created by Facebook in early 2014 and open sourced in Oct 2014
- Between 2014-2019 Facebook maintained osquery as an open-source project
- June 2019, Osquery project handed over to Linux Foundation
- October 2019, Osquery 4.0.2 become the first stable version from Linux foundation

# Osquery Schemas

Osquery uses SQL tables to represent abstract concepts such as

- running processes
- loaded kernel modules
- open network connections
- browser
- Plugins
- hardware events
- file hashes

Total tables : 277

- For Linux : 158
- For Windows : 104
- For MacOS : 191
- For Free BSD : 53

**process_events** (EVENTED TABLE)

Track time/action process executions.

Improve this Description on Github

| COLUMN | TYPE | DESCRIPTION |
|---|---|---|
| pid | BIGINT | Process (or thread) ID |
| path | TEXT | Path of executed file |
| mode | TEXT | File mode permissions |
| cmdline | TEXT | Command line arguments (argv) |
| cmdline_size | BIGINT | Actual size (bytes) of command line arguments |
| env | TEXT | Environment variables delimited by spaces |
| env_count | BIGINT | Number of environment variables |
| env_size | BIGINT | Actual size (bytes) of environment list |
| cwd | TEXT | The process current working directory |
| auid | BIGINT | Audit User ID at process start |
| uid | BIGINT | User ID at process start |
| euid | BIGINT | Effective user ID at process start |
| gid | BIGINT | Group ID at process start |
| egid | BIGINT | Effective group ID at process start |
| owner_uid | BIGINT | File owner user ID |

# Osquery Advantages

- Exposing an operating system as a high-performance relational database
- It delivers a single-agent solution using a universal query language to collect rich datasets for multiple use cases
- It's a cross-platform application with support for recent versions of macOS, Windows 10, CentOS, and Ubuntu.
- Running queries no longer requires specialized expertise. One can write SQL-based queries to explore data across all operating systems.

"SQL-powered operating system instrumentation, monitoring, and analytics"

# Running Osquery

You can install osquery by compiling it from source, or from package manager.

- **osqueryi:** The interactive osquery shell, for performing ad-hoc queries

- **osqueryd:** A daemon for scheduling and running queries in the background.

- osqueryi and osqueryd are independent tools.
- They don’t communicate.
- One can use one without the other.
- Most of the flags and options needed to run each are the same

# Configuring Osquery

- Osquery is not a plug-and-play application. Whether one intend to use the interactive shell (Osqueryi) or the daemon (Osqueryd)

- One can pass some flags and options, either from the command line or via a configuration file.

- Dozens of command-line flags and configuration options are available

```
osquery configuration options (set by config or CLI flags):

    --audit_allow_config                            Allow the audit publisher to change auditing configuration
    --audit_allow_fim_events                        Allow the audit publisher to install filesystem-related rules
    --audit_allow_process_events                    Allow the audit publisher to install process-related rules
    --audit_allow_sockets                           Allow the audit publisher to install socket-related rules
    --audit_allow_user_events                       Allow the audit publisher to install user-related rules
```

- Creating a configuration file makes it easier to run osqueryi. It is located in /etc/osquery/osquery.conf. It is also available to the daemon.

# Osquery Configuring File /etc/osquery/osquery.conf

vajra

- **Options:** A list of daemon options and feature settings. Used by Osqueryi and Osqueryd.

- **Scheduled queries:** A list of queries and when they should run.

- **Decorators:** Special queries which prepend data to other scheduled queries.
  For example: UUID of the host running osquery and the username of the user to every scheduled query.

- **Query Pack:** A list of packs to be used to conduct more specific scheduled queries.

# Example Configuring File

```
{
  "options": {
    "disable_events": "false"
  },
  "schedule": {
   "bpf_file_events": {
      "interval": 10,
      "query": "select * from bpf_file_events",
      "snapshot": false
    },
 "bpf_process_events": {
      "interval": 10,
      "query": "select * from bpf_process_events",
      "snapshot": false
    },
 "bpf_socket_events": {
      "interval": 10,
      "query": "select * from bpf_socket_events",
      "snapshot": false
    }
  },
  "file_paths": {
    "all_writable": [
      "/tmp/%%",
      "/dev/%%",
      "/var/%%",
    "/bin/%%",
"/boot/%%",
"/dev/%%",
"/etc/%%",
"/home/%%",
    ],
    "bin_dirs": [
      "/usr/bin/%",
      "/usr/sbin/%",
      "/bin/%",
      "/sbin/%"
    ],
    "pam_unix": [
      "/usr/lib/x86_64-linux-gnu/security/pam_unix.so"
    ],
  }

"decorators": {
    "load": [
      "SELECT uuid AS host_uuid FROM system_info;",
      "SELECT user AS username FROM logged_in_users ORDER BY time DESC LIMIT 1;"
    ]
  },
}

"packs": {
    "osquery-monitoring": "/usr/share/osquery/packs/osquery-monitoring.conf",
    "incident-response": "/usr/share/osquery/packs/incident-response.conf",
    "it-compliance": "/usr/share/osquery/packs/it-compliance.conf",
    "vuln-management": "/usr/share/osquery/packs/vuln-management.conf"
  }
```

# Use of Osqueryi

- In an interactive shell allowing users to run SQL queries on OS
- Used for ad-hoc queries and  debugging
- Do not persist between two sessions

```
(base) mkh@Manjeshs-MacBook-Air /etc % osqueryi
Using a virtual database. Need help, type '.help'
osquery> select * from logged_in_users ;
+------+------+---------+------+------------+-------+
| type | user | tty     | host | time       | pid   |
+------+------+---------+------+------------+-------+
| user | mkh  | console |      | 1676806789 | 144   |
| user | mkh  | ttys000 |      | 1676974998 | 10509 |
+------+------+---------+------+------------+-------+
osquery> select * from last ;
+----------+---------+-------+------+--------------+------------+------+
| username | tty     | pid   | type | type_name    | time       | host |
+----------+---------+-------+------+--------------+------------+------+
| mkh      | ttys000 | 10509 | 7    | user-process | 1676974998 |      |
| mkh      | console | 144   | 7    | user-process | 1676806789 |      |
| mkh      | ttys000 | 9526  | 8    | dead-process | 1676103353 |      |
| mkh      | ttys000 | 9526  | 7    | user-process | 1676103353 |      |
| mkh      | console | 4572  | 7    | user-process | 1675903215 |      |
| mkh      | console | 144   | 8    | dead-process | 1675885710 |      |
| mkh      | console | 144   | 7    | user-process | 1675674071 |      |
| mkh      | console | 432   | 7    | user-process | 1675008685 |      |
| mkh      | console | 116   | 7    | user-process | 1675006803 |      |
+----------+---------+-------+------+--------------+------------+------+
```

```
osquery> select command, path from crontab ;
osquery> select * from suid_bin ;
+----------------------+----------+-----------+-------------+
| path                 | username | groupname | permissions |
+----------------------+----------+-----------+-------------+
| /bin/ps              | root     | wheel     | S           |
| /usr/bin/write       | root     | tty       | G           |
| /usr/bin/top         | root     | wheel     | S           |
| /usr/bin/atq         | root     | wheel     | S           |
| /usr/bin/crontab     | root     | wheel     | S           |
| /usr/bin/atrm        | root     | wheel     | S           |
| /usr/bin/newgrp      | root     | wheel     | S           |
| /usr/bin/su          | root     | wheel     | S           |
| /usr/bin/batch       | root     | wheel     | S           |
| /usr/bin/at          | root     | wheel     | S           |
| /usr/bin/quota       | root     | wheel     | S           |
| /usr/bin/sudo        | root     | wheel     | S           |
| /usr/bin/login       | root     | wheel     | S           |
| /usr/sbin/traceroute6 | root     | wheel     | S           |
| /usr/sbin/postqueue  | root     | _postdrop | G           |
| /usr/sbin/traceroute | root     | wheel     | S           |
| /usr/sbin/postdrop   | root     | _postdrop | G           |
+----------------------+----------+-----------+-------------+
```

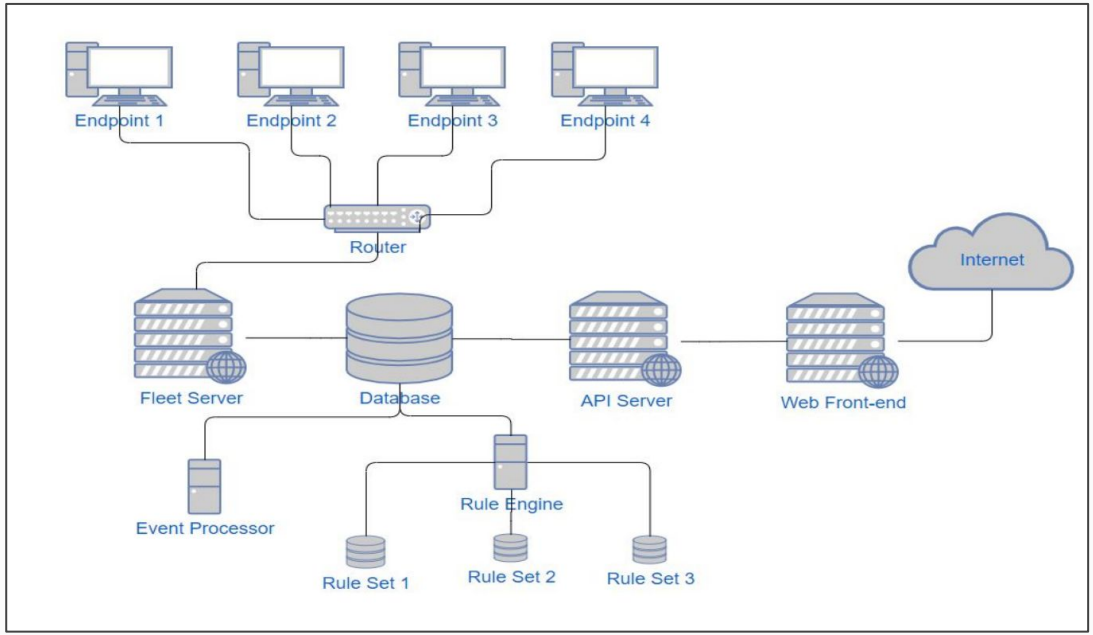
```
osquery> select *  from listening_ports limit 10;
+-----+-------+----------+--------+---------+----+--------+------+
| pid | port  | protocol | family | address | fd | socket | path |
+-----+-------+----------+--------+---------+----+--------+------+
| 144 | 0     | 17       | 2      | 0.0.0.0 | 6  | 0      |      |
| 389 | 0     | 0        | 0      |         | 0  | 3      |      |
| 392 | 0     | 0        | 0      |         | 0  | 3      |      |
| 392 | 0     | 0        | 0      |         | 0  | 5      |      |
| 401 | 0     | 17       | 2      | 0.0.0.0 | 3  | 0      |      |
| 401 | 0     | 17       | 2      | 0.0.0.0 | 4  | 0      |      |
| 401 | 0     | 0        | 0      |         | 0  | 5      |      |
| 401 | 61144 | 6        | 2      | 0.0.0.0 | 6  | 0      |      |
| 401 | 61144 | 6        | 10     | ::      | 7  | 0      |      |
| 401 | 0     | 0        | 0      |         | 0  | 8      |      |
+-----+-------+----------+--------+---------+----+--------+------+
```

# Use of Osqueryd

- Osquery is a daemon that runs continuously in the background
- Real time monitoring and security analysis
- Store data in database for later analysis
- Required configuration details of what data to collect and where to store it
- By default, data is published in file /var/log/osquery/osqueryd.results.log
- It supports collection of logs from multiple devices at a

# Centralized Processing and Monitoring

- Endpoints with Osquery'd
- Fleet server
- Database
- Event processor
- API server
- Rule Engine
- Web Management Console



# Performance impact of Osquery on endpoints

- Better visibility helps in threat
- Better visibility is possible by enabling the daemon to collect data from more table. But this consumes more resources and loads the endpoints
- Osquery is built for performance but it is easy to schedule queries that will significantly effect the performance
- Osquery provides option of Watchdog to limit the load
- Watchdog kills scheduled queries that consumes resournce above some threshold and blacklists them.

# Thank you

**Contact: mhanawal@iitb.ac.in**