Malware Handling

What is an incident timeline?

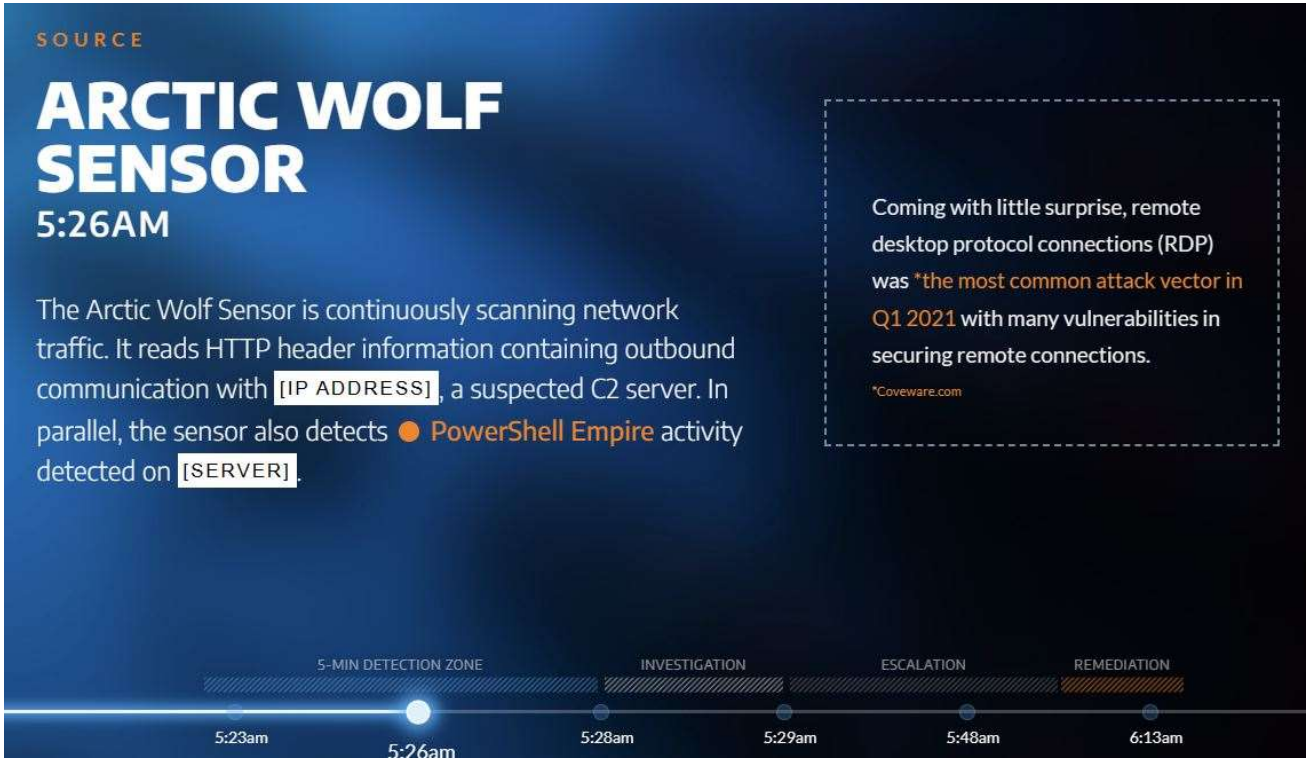
- An incident timeline is a complete real-time record of an incident. It often includes manual entries (chat), consolidated records of pages, alerts, and acknowledgements, and automatic system updates (for example, notification that someone has changed the severity level of an incident or marked it as resolved). It's also often synced with chat or a Slack channel.
- The timeline is there to keep the team on the same page, get new team members up to speed quickly, and simplify the process of incident postmortems.

Importance of incident timeline

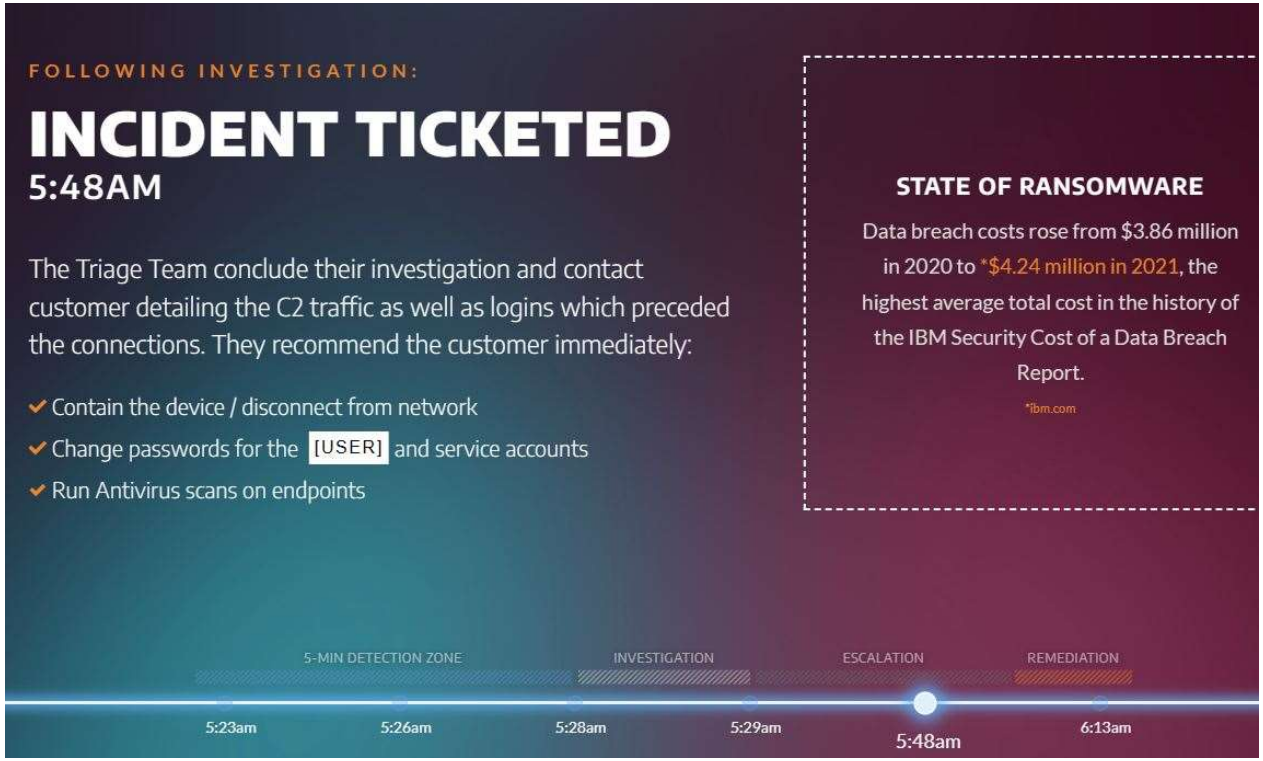
- A single real-time view
- One of the quickest ways for an incident to get out of control is a lack of communication between teams or stakeholders. An incident timeline mitigates this risk by giving everyone the same information in a single view in real time. This means everyone—from the developers working on the incident to the communications team responsible for updating users to c-suite stakeholders—can stay up to speed without complicated games of telephone or multiple disconnected email threads, phone calls, and chats.
- Digging deeper into KPIs
- An incident timeline often helps teams get to the bottom of a single incident, but its usefulness doesn't stop there. It can also be used alongside timelines for similar incidents to help teams spot patterns and diagnose larger problems with important KPIs.

Incident timeline : Example

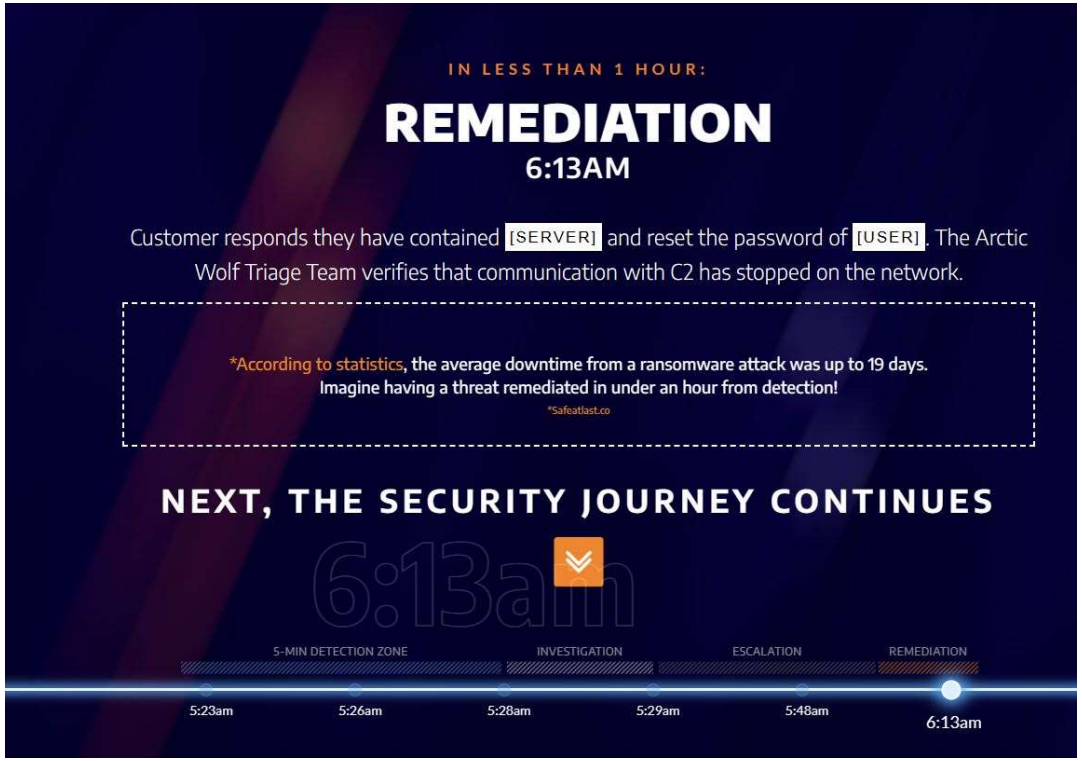
Incident timeline : Example



Incident timeline : Example



Incident timeline : Example



Incident timelines vs. ChatOps

- Incident timelines are typically provided by and used within incident management systems like Opsgenie to centralize all incident information.
- ChatOps for incident management has the same goal. The only difference is that instead of being housed in an incident management system, ChatOps typically centralizes the timeline in a chat program like Slack, which syncs with and pulls in information from incident management platforms like Opsgenie and any other relevant sources.
- The benefits of ChatOps—access to the same information across teams, real-time conversations and updates, less context-switching, no more games of telephone, and a built-in record for postmortems—are the same benefits that an incident timeline promises. The core difference is simply the location and amount of information. For most incident teams, the ChatOps feed typically has a lot of noise surrounding the important information. It's helpful to pull the rich details into your incident timeline, while retaining the chat log for future reference should you ever need it.

Botnet Identification

- The word “botnet” is an amalgamation of two terms: robot (bot) and network. A botnet is a network of computers, called “bots”, which are controlled by a single attacker, called “bot herder” or “bot master”.
- To control the machines in the network, the bots are infected with malware that places them under the control of the bot herder. The bot herder can then command all bots to do its bidding, which is typically to carry out attacks. Infected bots are often referred to as “zombies”.
- The bot herder acts remotely, sending updates to infected machines, manipulating their actions as needed. When created on a large scale, botnets containing millions (or even billions!) of bots can cause massive damage. This enables the owners to rent access to parts of their botnets on the black market.

How Does a Botnet Attack Work?

- There are numerous ways to create a botnet. For example, bot herders often create simple Command and Control (C2) botnets. In this case, they first set up the backend, which is a server that provides the C2 structure.
- For the structure, the herder can use a web application stacked on top of a Linux, Apache, MySQL and PHP (LAMP) environment using PHP and MySQL. Next, they create a bot builder, which packs a malware payload and then embeds it with the address of the C2 and relevant configuration information.
- Not all bot herders design their botnets from scratch. In fact, there are many botnet kits, like Neutrino and Ice IX (ICE9), available on the dark web, offered for sale or as Software as a Service (SaaS).

Indicators for Botnet Activity

- There are various signs that may indicate whether your resources are under a botnet attack. Organizations operating Security Operations Centers (SOC) are typically alerted by their security tooling. An Endpoint Detection and Response (EDR) system, for example, monitors endpoints and can compare normal traffic with abnormal traffic, sending alerts and responding to attacks as needed.

Indicator #1: abnormally high web-server CPU load

- If your web-server CPU load is abnormally high, there might be a process using too many server resources. In this case, you need to quickly investigate the matter to check if it is a legitimate service or some malware injected into your systems by threat actors.

Indicator #2: excessive network traffic that cause either full or partial network blockage

- Typically, during a network blockage, users are not able to access web-based resources. In this case, users will receive error codes like 504, 503, 502, 408, or 404.
- **Too much incoming traffic**—may indicate a DDoS attack. This typically means the bot herder commanded the botnet to overwhelm your systems until they drop.
- **Too much outbound traffic**—on the other hand, might indicate your system was hijacked. During these attacks, attackers reroute traffic elsewhere, which is why users experience downtime.

Indicators for Botnet Activity

Indicator #3: excessive memory usage

- A single botnet process typically needs to consume massive amounts of system resources, and may even consume all available system memory.

Indicator #4: non-native traffic profiles

- Abnormal network traffic may also indicate a botnet attack, especially if the traffic occurs over interfaces, ports, or protocols without being implemented by your known services.

Malware Handling

- Malware, also known as malicious code, refers to a program that is covertly inserted into another program with the intent to destroy data, run destructive or intrusive programs, or otherwise compromise the confidentiality, integrity, or availability of the victim's data, applications, or operating system.
- Malware is the most common external threat to most hosts, causing widespread damage and disruption and necessitating extensive recovery efforts within most organizations.

Malware Handling

- **Forms of Malware**
- Viruses. A virus self-replicates by inserting copies of itself into host programs or data files.
- Worms. A worm is a self-replicating, self-contained program that usually executes itself without user intervention
  - Network Service Worms. A network service worm takes advantage of a vulnerability in a network service to propagate itself and infect other hosts. –
  - Mass Mailing Worms. A mass mailing worm is similar to an email-borne virus but is selfcontained, rather than infecting an existing file.
- Trojan Horses. A Trojan horse is a self-contained, nonreplicating program that, while appearing to be benign, actually has a hidden malicious purpose. Trojan horses either replace existing files with malicious versions or add new malicious files to hosts.
- Malicious Mobile Code. Malicious mobile code is software with malicious intent that is transmitted from a remote host to a local host and then executed on the local host, typically without the user's explicit instruction. Popular languages for malicious mobile code include Java, ActiveX, JavaScript, and VBScript.
- Blended Attacks. A blended attack uses multiple infection or transmission methods. For example, a blended attack could combine the propagation methods of viruses and worms

Malware Handling

- **Attacker Tools**
- Backdoors. A backdoor is a malicious program that listens for commands on a certain TCP or UDP port.
- Keystroke Loggers. A keystroke logger monitors and records keyboard use. Some require the attacker to retrieve the data from the host, whereas other loggers actively transfer the data to another host through email, file transfer, or other means.
- Rootkits. A rootkit is a collection of files that is installed on a host to alter its standard functionality in a malicious and stealthy way. A rootkit typically makes many changes to a host to hide the rootkit's existence, making it very difficult to determine that the rootkit is present and to identify what the rootkit has changed.
- Web Browser Plug-Ins. A web browser plug-in provides a way for certain types of content to be displayed or executed through a web browser. Malicious web browser plug-ins can monitor all use of a browser.
- E-Mail Generators. An email generating program can be used to create and send large quantities of email, such as malware and spam, to other hosts without the user's permission or knowledge.
- Attacker Toolkits. Many attackers use toolkits containing several different types of utilities and scripts that can be used to probe and attack hosts, such as packet sniffers, port scanners, vulnerability scanners, password crackers, and attack programs and scripts.

Malware Prevention Policy

- malware prevention–related policy considerations include the following:
- Requiring the scanning of media from outside of the organization for malware before they can be used
- Requiring that email file attachments be scanned before they are opened
- Prohibiting the sending or receipt of certain types of files (e.g., .exe files) via email
- Restricting or prohibiting the use of unnecessary software, such as user applications that are often used to transfer malware (e.g., personal use of external instant messaging and file sharing services)
- Restricting the use of removable media (e.g., flash drives), particularly on hosts that are at high risk of infection, such as publicly accessible kiosks
- Specifying which types of preventive software (e.g., antivirus software, content filtering software) are required for each type of host (e.g., email server, web server, laptop, smart phone) and application (e.g., email client, web browser), and listing the high-level requirements for configuring and maintaining the software (e.g., software update frequency, host scan scope and frequency)
- Restricting or prohibiting the use of organization-issued and/or personally-owned mobile devices on the organization's networks and for telework/remote access.

Malware Incident Response

- Preparation
- Building and Maintaining Malware-Related Skills
- Facilitating Communication and Coordination
- Acquiring Tools and Resources

- Detection and Analysis
- Identifying Malware Incident Characteristics
- Malware category (e.g., virus, worm, Trojan horse)
- Services, ports, protocols, etc. that are attacked
- Vulnerabilities that are exploited (e.g., software flaws, misconfigurations, social engineering)
- Malicious filenames, sizes, content, and other metadata (e.g., email subjects, web URLs)
- Which versions of operating systems, devices, applications, etc., may be affected
- How the malware affects the infected host, including the names and locations of affected files, altered configuration settings, installed backdoor ports, etc.

Malware Incident Response

- Identifying Infected Hosts
- Forensic Identification
  - DNS Server Logs.
  - Network Forensic Tools.
  - Network Device Logs.
- Active Identification
  - Custom Network-Based IPS or IDS Signature.
  - Packet Sniffers and Protocol Analyzers.
  - Manual Identification

Malware Incident Response

- Prioritizing Incident Response
- How the malware entered the environment and what transmission mechanisms it uses
- What type of malware it is (e.g., virus, worm, Trojan horse)
- Which types of attacker tools are placed onto the host by the malware
- What networks and hosts the malware is affecting and how it is affecting them
- How the impact of the incident is likely to increase in the following minutes, hours, and days if the incident is not contained.

Malware Incident Response

- Malware Analysis
- Containment

- Containment of malware has two major components: stopping the spread of the malware and preventing further damage to hosts. Nearly every malware incident requires containment actions. In addressing an incident, it is important for an organization to decide which methods of containment to employ initially, early in the response. Containment of isolated incidents and incidents involving noninfectious forms of malware is generally straightforward, involving such actions as disconnecting the affected hosts from networks or shutting down the hosts. For more widespread malware incidents, such as fast-spreading worms, organizations should use a strategy that contains the incident for most hosts as quickly as possible; this should limit the number of machines that are infected, the amount of damage that is done, and the amount of time that it will take to fully recover all data and services

Malware Incident Response

- Containment Through Automated Detection
  - Content Filtering
  - Network-Based IPS Software.
  - Executable Blacklisting
- Containment Through User Participation
- Containment Through Disabling Services
- Containment Through Disabling Connectivity

Malware Incident Response

- Eradication
- organizations should rebuild any host that has any of the following incident characteristics, instead of performing typical eradication actions (disinfection):
- One or more attackers gained administrator-level access to the host.
- Unauthorized administrator-level access to the host was available to anyone through a backdoor, an unprotected share created by a worm, or other means.
- System files were replaced by a Trojan horse, backdoor, rootkit, attacker tools, or other means.
- The host is unstable or does not function properly after the malware has been eradicated by antivirus software or other programs or techniques. This indicates that either the malware has not been eradicated completely or that it has caused damage to important system or application files or settings.
- There is doubt about the nature of and extent of the infection or any unauthorized access gained because of the infection.

Malware Incident Response

- Recovery
- Lessons Learned

- Security Policy Changes. Security policies might be modified to prevent similar incidents. For example, if connecting personally owned mobile devices to organization laptops caused a serious infection, modifying the organization's policies to secure, restrict, or prohibit such device connections might be advisable.
- Awareness Program Changes. Security awareness training for users might be changed to reduce the number of infections or to improve users' actions in reporting incidents and assisting with handling incidents on their own hosts.
- Software Reconfiguration. OS or application settings might need to be changed to support security policy changes or to achieve compliance with existing policy.
- Malware Detection Software Deployment. If hosts were infected through a transmission mechanism that was unprotected by antivirus software or other malware detection tools, an incident might provide sufficient justification to purchase and deploy additional software.
- Malware Detection Software Reconfiguration. Detection software might need to be reconfigured in various ways, such as the following: – Increasing the frequency of software and signature updates – Improving the accuracy of detection (e.g., fewer false positives, fewer false negatives) – Increasing the scope of monitoring (e.g., monitoring additional transmission mechanisms, monitoring additional files or file systems) – Changing the action automatically performed in response to detected malware – Improving the efficiency of update distribution.