Measuring the Potential Impact of PIPEDREAM Malware OPC UA Module, MOUSEHOLE

By Sam Hanson

Published: 2023-10-25 · Archived: 2026-04-05 18:48:32 UTC

This is the second posting in our two-part blog series on PIPEDREAM's OPC UA Module, MOUSEHOLE. To view our first blog in this series, see: Deep Dive Into PIPEDREAM's OPC UA Module, MOUSEHOLE

In April of 2022, Dragos published a whitepaper and hosted a webinar to alert and inform the industrial cybersecurity community of a sophisticated new malware, PIPEDREAM, the seventh known industrial control systems (ICS)-specific malware developed by the CHERNOVITE threat group. Dragos Threat Intelligence followed up with a blog titled, Analyzing PIPEDREAM: Results from Runtime Testing. Continuing this research, we are releasing additional information on the Open Platform Communications Unified Architecture (OPC UA) module, MOUSEHOLE, focusing on further analysis and runtime testing results.

In the final installment of this two-part blog series, Dragos Threat Intelligence analysts present runtime testing results on an experiment named MOUSELAB, a derivation of MOUSEHOLE developed by Dragos to assess the impacts of the tool's capabilities. We also provide OPC UA server security settings and best practice recommendations for ICS/OT defenders.

Dynamic Analysis and Infrastructure Setup

Dragos first analyzed MOUSEHOLE statically, that is, purely by reading the code. However, Dragos quickly determined that dynamic analysis was necessary to test the limitations and full functionality of the malware.

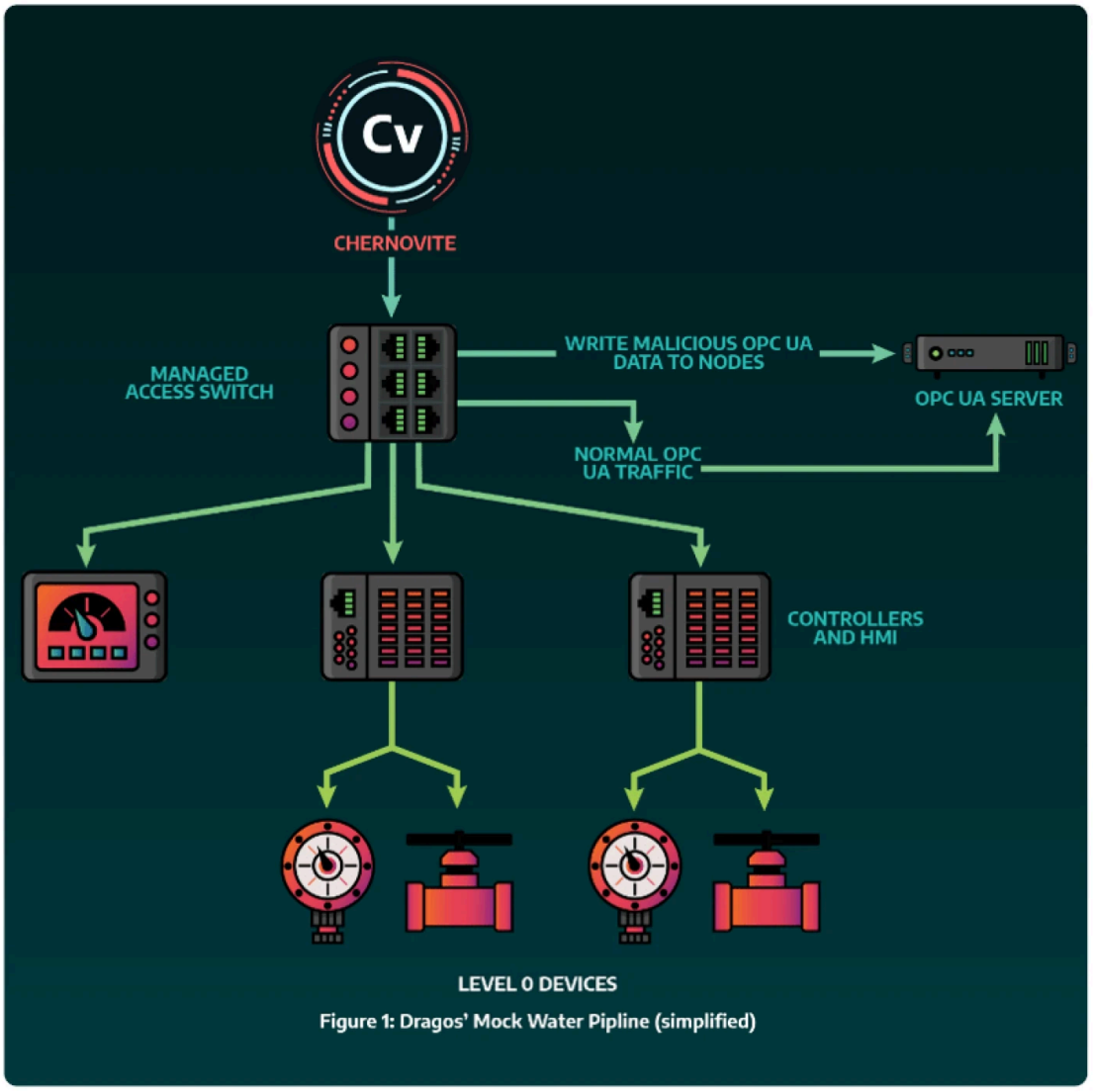

Figure 1: Dragos' Mock Water Pipline (simplified)

Dragos set up a system on our cyber range, as demonstrated in Figure 1, functioning as a mock water pipeline to test against MOUSEHOLE. During testing, it became apparent that each step of the attack required user interaction. While successful in impacting our mock pipeline by increasing the pressure to unsafe conditions, Dragos researchers hypothesized that most of the interaction with the OPC UA server would be automated in a real-world scenario.

A More Realistic Attack Scenario

The following is solely a hypothetical scenario created by Dragos to describe how CHERNOVITE may deploy MOUSEHOLE.

To accomplish an effect on any given industrial process, the operator is required to execute a MOUSEHOLE command at nearly every step: scanning the network; enumerating the endpoints; reading node attributes; and writing node attributes. In a more realistic attack scenario, the adversary deploys and executes MOUSEHOLE to conduct reconnaissance against the OPC UA server. The adversary collects intelligence, such as what nodes map to operational-impacting process values. Once the OPC UA address space is understood, the adversary hardcodes all node identifiers and values they want to modify into a new, automated version of MOUSEHOLE.

Developing an automated version allows the adversary to use MOUSEHOLE as a reconnaissance tool without completely "burning" the capability. Once reconnaissance in the adversary's target environment is complete, then the automated capability would be deployed and "burned."
Analyst note: When an adversary's capability is discovered (colloquially referred to as "burned") defenders will create detections and produce recommendations to help mitigate the threat. Adversaries never want to burn a capability unnecessarily – custom malware takes time, resources, and expertise to develop.

To simulate a more realistic attack tool, Dragos researchers experimented with an automated, victim-specific version of MOUSEHOLE nicknamed MOUSELAB. The goal of this experiment was to create unsafe conditions in the water pipeline by increasing the water pressure. The results of this experiment would provide a better understanding of MOUSEHOLE's effectiveness and help us, as defenders, to create more effective detections and mitigation strategies.

The MOUSELAB Experiment

MOUSELAB consists of two major steps – the first, reconnaissance, authentication requirements, and access controls; the second, writing malicious data to node value attributes.

Step 1: Reconnaissance, Authentication Requirements, and Access Controls

MOUSELAB first scans the network looking for OPC UA servers. Once servers are found, it obtains server endpoints using OPC UA's discovery service and identifies which servers are set up with weak security modes enabled (in this case, a security mode of "none"). MOUSELAB then attempts "anonymous login," that is, a login without any user credentials. If credentials are required, MOUSELAB asks the user for the username and password list and begins brute-forcing the server. Once the username and password are successfully found, MOUSELAB obtains the server structure and enumerates the address space looking for nodes with read or write access enabled. This information is returned to the operator, who must then analyze these nodes and determine which could have an operational impact.

Step 2: Writing Malicious Data to Node Value Attributes

After performing initial reconnaissance on the target OPC UA server, the operator hardcodes the node's identifiers and values to be overwritten into MOUSELAB. Finally, they execute the program, writing the new values to the server and directly impacting operational conditions.

To achieve the goal of creating unsafe conditions in the pipeline, a few OPC UA node values must be modified in the server's address space representing various process information. First, the solenoid valve, represented as a node with a Boolean datatype, must be set to false. This blocks the water from escaping and relieves pressure in the pipe. Next, the water flow rate node value is increased to 100%, pumping water into the pipeline as fast as possible.

However, if we attempt to increase the pressure by only modifying the flow rate and closing the solenoid valve, we will trigger a safety shutdown once the pressure reaches our high-pressure set point node value, currently set to 15 PSI. When a safety shutdown is triggered, the system relieves the pipeline pressure and issues an over-pressure reset. Once the alarm is cleared, the process is ready to resume normal behavior. The safety shutdown can be seen

in our first execution of MOUSELAB as demonstrated in our DEF CON presentation recording: view time stamp: 9:31.

Next, we ran the experiment again, this time increasing the high-pressure set point to 100 PSI, far beyond the maximum safe pressure within the pipeline, as well as closing the solenoid valve and maximizing our flow rate. Setting the high-pressure set point to such a high value effectively disables it, allowing pressure to reach unsafe levels while still staying below the set point. The unsafe conditions shutdown can be seen in the same DEF CON presentation recording referenced above: view time stamp: 11:43.

Analyst note: Our pipeline is set up in such a way that the pump is not physically capable of increasing the pressure to a point of physical destruction. This small-scale experiment was designed to only test MOUSEHOLE's functionality with legitimate OPC UA servers. Whether a MOUSEHOLE-like tool can cause physical disruption or destruction depends on numerous factors in the target environment.

Dragos researchers set up the infrastructure to test MOUSELAB. A threat group such as CHERNOVITE would not have the luxury of such in-depth knowledge. Significant reconnaissance is required to fully understand what nodes would lead to operational impact and unsafe system conditions.

Dragos assesses with high confidence that MOUSEHOLE could impact operations. While MOUSELAB contains various enhancements that improved the attack from the adversary's perspective, these enhancements are not necessary for MOUSEHOLE to achieve an impact and should not be viewed as a limitation. MOUSEHOLE, in its current form, already contains all the functionality required to disrupt industrial processes.

OPC UA Server Security and Best Practices

While it's important to understand how MOUSEHOLE can impact an industrial process, the biggest takeaway for defenders is how to secure OPC UA servers.

Researchers from Germany's Federal Office for Information Security found that, unlike many industrial protocols, OPC UA has security built into its architecture and design, greatly increasing confidentiality, integrity, and availability. However, the security of OPC UA relies on proper setup, configuration, and deployment of the OPC UA server, which can be difficult depending on the configuration software used.

In fact, a significant number of OPC UA deployments have security issues. Researchers from the CISPA Helmholz Center for Information Security and Saarland University scanned the internet for public-facing OPC UA servers and published their results in a paper titled Security Analysis of Vendor Implementation of the OPC UA Protocol for Industrial Control Systems. They found that "…92% of the deployments show issues with security configurations. Among those, 44% of the servers are accessible without any authentication requirements." That is, anonymous access was enabled by default allowing any authenticated client to connect and interact with the OPC UA server.

While OPC UA has security baked into the design of the protocol, proper configuration and deployment by the OT asset owner and security practitioner must be taken seriously. Ensuring the security recommendations below are followed helps mitigate the threat of rogue OPC UA clients such as MOUSEHOLE.

Security Setting Recommendations for OPC UA Server Deployments

- Utilizing OPC UA's "Sign-only" security mode is preferred for ICS environments with network monitoring solutions such as the Dragos Platform. Sign-only security mode sends messages unencrypted with an authentication code that allows receivers to be sure the message came from a trusted sender, protecting against rogue OPC UA clients such as MOUSEHOLE, while still allowing the packets to be inspected by network monitoring solutions.
- Ensure private keys and certificates are stored in a secure location.
- Ensure server utilizes explicit trust lists for certificates and monitor trust list over time.
- If application authentication is enabled (UserNameIdentityToken), ensure long, complex passwords or passphrases are used.
- Ensure the server's Security Policy is *Basic256Sha256*. If your server uses a weaker Security Policy algorithm, change the policy immediately.
- Verify anonymous authentication is disabled. OPC UA server implementations often include an anonymous authentication option, sometimes enabled by default.
- Disable client read or write access for OPC UA nodes that do not require it. Monitor for unnecessarily lenient access controls on a node-by-node basis.

MOUSEHOLE and the other modules contained in PIPEDREAM represent a serious escalation in adversarial capabilities. Defenders must respond to these threats by ensuring assets are properly configured and secured. Dragos researchers hope the insights and intelligence gained from this research and presented in this blog helps push the industry towards a more secure future.

Get the Complete Analysis

Learn more about the discovery and capabilities of CHERNOVITE's PIPEDREAM malware in our whitepaper.

DOWNLOAD WHITEPAPER

Sam Hanson is a Vulnerability Analyst on the Intelligence Research Team. Sam graduated from the University of Minnesota – Twin Cities in 2020 with a Computer Science degree and a focus on Computer Security.
