Machine-to-Machine (M2M) Technology Design Issues and Implementation Vulnerabilities

Terms:
- Security Intelligence
  - Federico Maggi (Senior Threat Researcher)

In our latest research paper, titled “The Fragility of Industrial IoT’s Data Backbone: Security and Privacy Issues in MQTT and CoAP Protocols,” and written with Rainer Vosseler and Davide Quarta, we discussed how MQTT and CoAP are affected by design issues and implementation vulnerabilities that can allow attackers to subvert devices enabled by these two machine-to-machine (M2M) protocols. During our research, we found hundreds of thousands of misconfigured hosts that exposed credentials, sensitive information, and industry-related process data.

In this blog post, we delve into the protocol security issues that may crop up from a technology perspective. The scarce awareness that we’ve observed around the current state of MQTT and CoAP can enable attackers in achieving their goals, ranging from reconnaissance and lateral movement to remote control and targeted attacks.

Technology behind M2M communication

Message Queuing Telemetry Transport (MQTT) is a mature standard and a publish-subscribe protocol that handles one-to-many communication mediated by brokers. We found security issues in the protocol itself and its software implementations. This presents a problem as MQTT is widely adopted commercially for automation and in industrial applications, which could prove crucial for mission-critical M2M.

Constrained Application Protocol (CoAP), on the other hand, follows a client-server model, allowing the creation of the equivalent of HTTP for constrained nodes. Despite being relatively new, CoAP is already implemented in many pieces of internet-of-things (IoT) and industrial-internet-of-things (IIoT) software today. Security issues that we highlight are primarily due to the “connectionless” nature of User Datagram Protocol (UDP), on which it is based. Although the risks are well highlighted in the CoAP Request for Comments (RFC), in this research we empirically measured the bandwidth amplification potential of CoAP services.

MQTT issue in handling topic strings

The MQTT standard has some corner cases that could subject implementations to vulnerabilities. One of the cases involves the Unicode handling in topic strings. The standard leaves it up to the developer to close the connection upon failing the validation of disallowed UTF-8 code points. So if it happens that a broker does not check for disallowed UTF-8 code points, a malicious client could take advantage of the discrepancy to disconnect other clients through invalidly encoded strings.

![Figure 1. Context (top left); a broker that implements the check as in the standard (top right); both client and broker do not implement the check (bottom left); the broker lets the invalid character pass, while the client follows the standard (bottom right)](image)

Of course, if the broker follows the standard, it wouldn’t be an issue. Interestingly, if both the broker and the client do not follow the standard, the issue is also avoided. But if the broker does not follow the standard and the client does, the client will disconnect upon receiving invalid content.
Figure 2. A malicious client could exploit the discrepancy between validating versus non-validating nodes to keep nodes offline. Notably, in this case, not following the standard appears as the best approach for a client.

If the broker does not follow the standard, a malicious client could keep all the clients offline with a single "retain message" set to true and the quality of service (QoS) to 2. The clients will then disconnect upon receiving the invalid message and not acknowledge back to the broker. This, in turn, will have the broker assume that the clients never received the message and thus will keep flooding them with the same retained message over and over because of the "retain message" and "QoS = 2".

**Implementation vulnerabilities that attackers can take advantage of**

As an example, we here discuss one of the vulnerabilities that we found in some of the MQTT implementations. We also explain the effect of an amplification attack we demonstrated on CoAP.

**MQTT Payload Remaining Length (CVE-2018-17614)**

The vulnerability is an unbounded write-in caused by a missing check on the "remaining length" field in a popular MQTT library. This allows an attacker to execute arbitrary code on vulnerable devices that implement an MQTT client. An attacker here must either control a rogue MQTT broker, or the broker must be missing proper checks for the remaining length field and just relay MQTT packets "as they are" from publishers to subscribers. The vulnerability can be triggered during the parsing routine for an MQTT PUBLISH packet, and precisely when reading the "remaining length" and "topic length" fields.

Moreover, the overwrite in the callback field can be triggered as many times as an attacker wishes, which could lead to a persistent denial-of-service (DoS) condition. If the broker is malicious (or if a client can force a broker to relay a malicious payload), all vulnerable devices will be under a persistent DoS state, or will continue to execute arbitrary code "pushed" by the attacker.

There was a fix for this vulnerability. However, the developers could not agree on the best way to implement it. Once we notified the developers through a ZDI advisory, the fix was applied. As mentioned earlier, it is up to the developer whether to close the connection or to discard the packet and leave the connection open. We believe that this case is a good example of how corner cases in the specifications could lead to confusion.

**CoAP: IP address spoofing on UDP and the risk of amplification (PoC)**

As previously mentioned, CoAP is UDP-based, and thus inherently susceptible to Internet Protocol (IP) spoofing. Unlike transmission control protocol (TCP), UDP has no handshake phase, in which two endpoints agree on a sequence number that identifies a connection. This means that if an attacker sends a UDP packet with a spoofed source IP address to an endpoint, that endpoint will have no way to verify where the packet came from. This also means that if autonomous systems have no specific countermeasures against spoofing, an endpoint will have to trust the UDP packet header and act accordingly.
Another risk with CoAP is the possibility of amplification. In our experiment on a test network with CoAP clients and servers, we were able to launch an amplification attack with increasing payload size and estimate the maximum bandwidth amplification factor (BAF).

Aside from being UDP-based, CoAP is based on a request-response scheme. CoAP responses can be significantly larger than requests. In our estimate, CoAP can reach up to 32x amplification factor, which is roughly between the amplification of Domain Name System (DNS) and Simple Service Discovery Protocol (SSDP). An attacker who has access to a 1-Mbps link would be able to hit a target at 32 Mbps.

A further look at the protocol specification reveals that CoAP supports "block-wise transfers," which essentially means that a large response can be divided into smaller responses. Interestingly, “both sides have a say in the block size that actually will be used.” This means that an attacker can craft a request packet asking for the maximum block size, hence bringing the amplification factor up to 32x, which is substantial, considering that CoAP nodes are usually resource-constrained (e.g., running on battery).

A more detailed explanation on these issues and our other findings can be read in our research, “The Fragility of Industrial IoT’s Data Backbone: Security and Privacy Issues in MQTT and CoAP Protocols,” which includes an attack demonstration video. Our full paper also discusses the applications of M2M technology, impact in various sectors, and security considerations.

The post “Machine-to-Machine (M2M) Technology Design Issues and Implementation Vulnerabilities” appeared first on [source].