**Downgrade attack**

**Context: the *downgrade* vulnerability in QUIC v1**

In the early versions of QUIC (including v1), there was a risk that an attacker could intercept and modify version negotiation packets to force the client and server to **fall back** to an older protocol version (or another protocol like HTTP/2 over TCP).

Typically, a QUIC client sends a list of the versions it supports (QUIC v1, v2, etc.), and the server replies with the actual chosen version. In an attack scenario, a malicious intermediary could alter these messages to **trick** the client and server into believing that the other side did not support QUIC v1 or v2, thus forcing a *fallback* to a less secure or older protocol.

**QUIC v2: authenticated version negotiation**

In QUIC v2, the version negotiation mechanism has been redesigned to thwart this type of attack:

1. **Cryptographic authentication**:

Version negotiation messages are now protected so that both the client and server can be sure, by the end of the handshake, that **no one has tampered** with the chosen version en route.

Concretely, a token and/or a similar cryptographic element is used to **sign or validate** the version.

1. **Mutual validation**:

The client remembers the negotiated version and, upon receiving the server’s response, checks that this same version is indeed supported and accepted.

If there is any discrepancy (e.g., an unexpected version not listed by either the client or server), the connection fails.

1. **Explicit prevention of fallback**:

If an attacker tries to force a fallback to HTTP/2 (which runs over TCP + TLS, not QUIC), the client (or the server) will detect that the QUIC negotiation no longer aligns with what was announced.

The result is a **connection failure** rather than a silent fallback.

**Outcome: no more *downgrades* to HTTP/2 or other versions :**

Thanks to this protection, it is no longer possible (or at least *extremely* difficult) to trick the client and server into **switching to HTTP/2** (or QUIC v1, or another less secure protocol) **without their knowledge**.

Of course, if the server truly does not support QUIC v2 or HTTP/3, the connection may ultimately fall back to HTTP/2 or HTTP/1.1 — but only because the server legitimately indicates that it cannot support these newer protocols, not because of a malicious intermediary.

In summary, QUIC v2 establishes a secure version negotiation process that addresses most of the *downgrade* attacks encountered in earlier versions of QUIC. It is now much harder for an attacker to force a return to an older protocol version, such as HTTP/2 (or an outdated version of QUIC), because the final version is cryptographically verified end to end.