a request containing both the Content-Length and Transfer-Encoding headers. For instance:

- The front-end server uses the Content-Length header to determine the size of the request.
- The back-end server uses the Transfer-Encoding header to interpret the body as chunked data, leading to a new request being processed.

This discrepancy can allow unauthorized actions, such as injecting a malicious second request that bypasses security controls.

Issues with Incorrect Content-Length

If the Content-Length header does not match the actual size of the request body, servers may only process part of the request. For instance:

- If the body is 24 bytes long but the Content-Length header specifies 10 bytes, only the first 10 bytes are processed.

This can result in:

- Data loss or incomplete requests being processed.
- Failed attempts at request smuggling due to truncated payloads.

Defense Against CL.TE Request Smuggling

1. **Eliminate Ambiguity**:
    - Ensure consistency between how front-end and back-end servers process headers.
    - Disable the use of Transfer-Encoding if Content-Length is specified.
2. **Validate Headers**:
    - Drop requests that include both Content-Length and Transfer-Encoding headers.
    - Validate incoming headers before forwarding them.
3. **Update Software**:
    - Use the latest versions of server software with security patches for request smuggling vulnerabilities.
4. **Regular Testing**:
    - Conduct pentesting using tools like Burp Suite or OWASP ZAP to identify vulnerabilities in the web application infrastructure.

By addressing these vulnerabilities, organizations can mitigate the risks of CL.TE request smuggling and enhance their application security.