Upgrade CURL libraries for HTTP client

A libcurl upgrade is a feature that

- updates the embedded libcurl library from version 7.30 to 8.17.0 to align with upstream security and protocol improvements,
- preserves specific customizations such as VRF awareness, XR SSL integrations, and TCP performance tuning, and
- ensures backward compatibility and automatic benefit for all dependent applications without user intervention.

Table 1: Feature History Table

Feature Name	Release Information	Feature Description
Upgrade CURL libraries for HTTP client	Release 26.3.1	Introduced in this release on: NCS 5500 fixed routers. You gain enhanced security and improved reliability through an upgraded embedded libcurl library. This transparent update preserves existing functionality and continues to support HTTP/1.0 and HTTP/1.1.

- Key benefits for libcurl upgrades, on page 1
- Support and restrictions for libcurl upgrade, on page 2
- How libcurl upgrade works, on page 2

Key benefits for libcurl upgrades

The following are significant attributes and considerations for the libcurl upgrade:

- Updates the libcurl version to 8.17.0 for the underlying HTTP stack.
- Maintains compatibility for all existing APIs, so no changes are required for dependent applications.
- Retains and independently manages platform-specific patches, such as VRF, SSL, and TCP optimizations, by rebasing and validating them against the upgraded libcurl version, simplifying future maintenance.

- Supports major features such as Smart Licensing, Shell utility (Copy), Install Manager, CEPKI, SKS, and Zero Touch Provisioning (ZTP).
- Is not user-configurable and cannot be disabled; no new CLI or configuration is introduced.

Table 2: Libcurl upgrade attribute comparison

Attribute	Previous implementation	Upgraded implementation
Libcurl version	7.30	8.17.0
Supported HTTP protocols	HTTP, HTTPS	HTTP, HTTPS
XR-specific customizations	VRF support, SSL integration (XR libcrypto), TCP tuning	Maintained
API compatibility	Legacy APIs supported	All existing APIs maintained
Configuration impact	None	None (not configurable)

Note The upgrade is completely transparent to users and feature owners. No CLI changes or configuration are required, and the previous libcurl version cannot be restored by configuration.

Support and restrictions for libcurl upgrade

Provides an overview of the Cisco platforms that support the libcurl upgrade, details operational or configuration restrictions, and explains how the enhancement integrates with system infrastructure in supported environments.

The following points summarize platform support and key restrictions for the libcurl upgrade:

- The upgrade applies only to platforms that run Cisco IOS XR software, including supported hardware and virtual platforms that use the upgraded HTTP infrastructure.
- Users cannot enable, configure, or roll back the upgrade. The system image manages the upgrade as a transparent infrastructure enhancement.
- The upgrade preserves VRF awareness and compatibility with XR-specific socket, SSL, and networking features across supported platform variants.
- Existing HTTP client APIs and applications continue to operate without change on supported platforms.

How libcurl upgrade works

The libcurl upgrade is an internal infrastructure improvement that replaces the embedded libcurl library version 7.30 with version 8.17.0. This change aligns the HTTP client stack with modern standards for protocols, security, and maintainability. The upgrade is transparent to users; no changes to commands or configuration are required. Application components continue to function as before, preserving backward compatibility.

Security updates, platform-specific features, and operational observability are maintained through integration of upstream fixes and Cisco patches.

- The upgrade is internal and does not alter configuration workflows.
- All existing APIs and application interactions continue to be supported.
- Security and feature enhancements are introduced through targeted patches to the upgraded library.

Summary

The HTTP client infrastructure processes HTTP and HTTPS transactions through coordinated components that use a stable API and asynchronous communication.

- Application component: Initiates HTTP or HTTPS operations by calling the HTTP client library API.
- HTTP client library: Serves as the API interface and manages communication with the HTTP client process.
- HTTP client process: Handles request and response states, session management, concurrency, and calls libcurl.
- libcurl library: Executes protocol operations and network communication, with enhancements for XR integration.

Workflow

These stages describe the transaction flow for HTTP requests by using the upgraded libcurl library:

1. An application component, such as Smart Licensing, Install Manager, Zero Touch Provisioning, or the Copy utility, starts an HTTP or HTTPS request by calling the HTTP client library API.
   - The API remains unchanged, so existing feature components continue to work without modification.
   - The HTTP client library receives and processes the request.
2. The HTTP client library packages the request, adds required parameters, and sends the request asynchronously to the HTTP client process.
   - Asynchronous interprocess communication prevents operations from blocking.
   - Request details, such as VRF information and metadata, are included for execution.
   - The HTTP client process receives the prepared request.
3. The HTTP client process manages the request by queuing transactions and maintaining session state for multiple requests.
   - The process uses the libcurl multi interface to scale concurrent requests efficiently.
   - The process performs error handling and allocates required system resources.
   - Requests are sequenced and validated before dispatch to the communication layer. The process then prepares to call libcurl for protocol execution.
4. The HTTP client process calls the `curl_multi_perform` API for efficient asynchronous network operations.
   - Handles XR-custom socket management, VRF context, and SSL using Cisco extensions.

- Manages concurrency by using `select` and the libcurl multi interface.

This stage provides reliable, efficient transaction processing. Network protocol negotiation and TLS setup occur in the updated, security-patched libcurl library.

5. The libcurl library establishes the network connection, negotiates SSL or TLS as needed, and completes the HTTP transaction by using HTTP/1.0 or HTTP/1.1.
   - Maintains socket management, SSL via `libssl_xr`/`libcrypto_xr`, and TCP settings.
   - Security patches and protocol improvements are applied in this stage.
   - The library sends response and transaction state information to the HTTP client process.
6. The HTTP client process parses the libcurl response, updates transaction status, logs actions or errors, and tracks operational state.
   - Uses standardized error codes and logging to simplify troubleshooting and monitoring.
   - Makes status information available through show commands.

   Transaction results are passed to the client library. The HTTP client library supplies the response and status to the application.
7. The application receives the response from the HTTP client library and continues normal operation.

   The HTTP transaction process completes, and resources are released. No user or application-level awareness or adaptation is needed for the upgrade.

Result

The system benefits from a secure, up-to-date HTTP client infrastructure with improved protocol support, stronger security, and better operational efficiency, with no change to applications or user experience.