Technology Stack Research

Evaluate C++ Frameworks (Boost, Poco, Qt):

A. Boost C++ Libraries

Boost is one of the most mature and comprehensive C++ library collections.

Why Boost is relevant for an EDR agent: (EDR = Endpoint Detection and Response):-

- **Filesystem operations** → scanning directories, detecting new file creations
- **Asio (Networking)** → asynchronous HTTP(S), TCP communication
- **Process handling** → ability to inspect running processes
- **Threading & concurrency** → telemetry batching, background monitoring loops
- **Cross-platform support** → Linux + Windows portability
- **Smart pointers & utilities** → safer memory handling, efficient performance

Pros

- Very mature & widely adopted
- Extremely high performance
- STL-like API (becomes easier once learned)
- Many modules useful for system-level programming

Cons

- Large library—can be heavy to compile
- Learning curve is high for beginners

Does Boost Work Well for Your EDR? → YES (Highly Recommended)

Why It Works

✓ EDR agents require continuous process monitoring — Boost.Process helps.
✓ Fast hashing & file scanning need efficient filesystem access — Boost.Filesystem works well.
✓ Telemetry batching needs threaded background workers — Boost.Thread excels.
✓ mTLS communication can be built using Boost.Asio + OpenSSL.
✓ Cross-platform: You can port to Windows easily later.

Limitations

❌ Complex API, difficult for new programmers
❌ Compilation increases build time
❌ Asio TLS integration requires manual OpenSSL handling

Suitability Score for BESS EDR: 9/10

→ Best suited for low-level, high-performance agent code.

B. Poco C++ Libraries

Poco is a modern, lightweight framework aimed at building networked applications.

Why Poco is relevant for your EDR

- **HTTP/HTTPS client** → sending telemetry to CMS
- **mTLS support** → secure communication
- **JSON handling** → telemetry serialization
- **Threading + Tasks** → periodic background checks
- **File and process utilities** → system inspection
- Multi-threaded background services
- Poco internally uses OpenSSL for secure communication.

Pros

- Smaller & cleaner compared to Boost
- Excellent for REST communication
- Easier to learn
- Built-in SSL/TLS support

Cons

- Less feature-rich than Boost
- Smaller community

Does Poco Work Well for Your EDR? → YES (Best Option for Communications)

Why It Works

✓ Telemetry needs HTTPS + JSON → Poco::Net + Poco::JSON works out of the box.
✓ mTLS authentication → supported by Poco via SSLManager.
✓ Very fast for REST-based communication with CMS.
✓ Simpler than Boost for HTTP/HTTPS operations.
✓ Lightweight, ideal for a daemon/service.

Limitations

✗ Not as powerful as Boost for low-level OS interactions
✗ Community is smaller
✗ Harder to integrate with asynchronous designs compared to Boost.Asio

Suitability Score: 8.5/10

→ Best suited for a clean, REST-oriented, lightweight agent.

C. Qt Framework

Qt is primarily a GUI framework but also includes networking and system APIs.

Qt is a full application framework that includes:

- GUI system
- Network modules
- Database layers
- Thread and event loop
- Cross-platform deploy system

Qt applications run using a large runtime environment (~40–100 MB)

Why Qt is *less ideal* for your EDR

- Heavy-weight for background system services
- GUI features not needed for agent service
- Larger runtime dependency

Pros

- Great cross-platform support
- Good networking utilities

Cons

- Very heavy compared to Boost/Poco
- Unnecessary for a CLI/daemon-based agent

Does Qt Work Well for Your EDR? → NO (Not Suitable for an Agent)

Why It Does NOT Work

❌ Too heavy for a background agent
❌ Requires Qt runtime libraries → increases agent size
❌ GUI modules unnecessary
❌ Slower startup for system monitoring
❌ Overkill for daemon-style development

Only useful if you build a GUI dashboard, not for an endpoint agent.

Suitability Score: 3/10

→ **Not recommended for EDR agent unless you plan a GUI dashboard.**

Overall:

✓ **Poco + selected Boost libraries** is the best combination.
✓ Qt should be avoided for agent development.