Quantum Readiness Scanner - User Guide

Version 2.0.1.47
January 23, 2026

Document Revision Log			
Software Version	**Document ID**	**Date**	**Change Description**
2.0.1.47	QRUG20147_001	01/23/2026	• Removed “TYCHON” from required license format in the License Configuration section. • Added newly supported flags for Command Reference > Output and Reporting. • New Output Feature section within Appendix A: Output Formats • New Output Features: Split Outputs & Details Levels section within the Usage Examples section.

Table of Contents

What is Quantum Readiness Scanner?

Quantum Readiness Scanner is a cross-platform executable that you download and run on your endpoints to scan for cryptographic assets. The binary supports multiple scan modes (network, filesystem, memory, VPN/IPSec detection), outputs results in various formats (JSON, CBOM, HTML, NDJSON), and can send findings directly to integration platforms like Splunk, Elasticsearch, AWS S3, Cloudflare R2, Kafka, or save locally for custom processing.

The Post-Quantum Cryptography Crisis

Quantum computers will break current encryption within the next 10-15 years. When this happens, every RSA key, ECDSA signature, and Diffie-Hellman key exchange protecting your organization today will become instantly vulnerable. This isn't a theoretical future problem—it's an imminent business risk requiring immediate action.

The "Harvest Now, Decrypt Later" Attack

Nation-state adversaries and sophisticated threat actors are already collecting encrypted data today, storing it until quantum computers become available to decrypt it. Your sensitive communications, financial data, and intellectual property from today could be compromised years from now without proper post-quantum preparation. The data you're protecting right now has a shelf life that extends well beyond the quantum timeline.

Immediate Business Risks

- Financial Systems: Banking, payment processing, and financial communications vulnerable
- Healthcare Records: Patient data and medical systems at risk of future exposure
- Intellectual Property: Trade secrets, R&D data, and competitive advantages compromised
- Government Contracts: NIST compliance requirements and security clearance implications
- Supply Chain: Partner communications and vendor integrations vulnerable
- Customer Trust: Brand damage from future data breaches of today's encrypted data

Hidden Cryptographic Debt

- Legacy Applications: Hardcoded crypto in custom software
- Embedded Systems: IoT devices with unfixable crypto implementations
- Third-Party Software: Vendor applications with unknown crypto dependencies
- Cloud Services: Multi-tenant platforms with shared crypto infrastructure
- Mobile Applications: Certificate pinning and embedded keys in mobile apps
- Database Encryption: TDE, column-level, and application-layer crypto

Compliance & Regulatory Timeline

Federal agencies and critical infrastructure must transition to post-quantum cryptography by2035per NIST guidelines. Many industries will face earlier requirements:

How Quantum Readiness Scanner Solves This

The Quantum Readiness Scanner provides the comprehensive cryptographic asset discovery and analysis capabilities organizations need to prepare for the post-quantum transition. By identifying every cryptographic implementation across your infrastructure—from network

services to embedded applications—you can prioritize migration efforts, ensure regulatory compliance, and maintain security during the critical transition period.

Discovery Phase

- Complete Asset Mapping: Find every crypto implementation
- Hidden Dependencies: Discover embedded and inherited crypto
- Risk Prioritization: Identify most critical vulnerable systems
- Compliance Baseline: Document current state for auditors

Analysis Phase

- Quantum Vulnerability: Assess PQC readiness across systems
- Migration Planning: Understand replacement complexity
- Business Impact: Model risks and timeline requirements
- Cost Estimation: Budget for cryptographic upgrades

Transition Phase

- Progress Tracking: Monitor migration completion
- Continuous Monitoring: Detect new vulnerable deployments
- Validation Testing: Verify post-quantum implementations
- Compliance Reporting: Demonstrate regulatory adherence

Primary Use Cases

- Post-Quantum Readiness: Identify quantum-vulnerable cryptographic implementations
- Certificate Discovery: Track X.509 certificates, expiration dates, and trust chains
- Risk Assessment: Evaluate cryptographic strength and identify weak implementations
- Security Assessments: Discover all cryptographic assets across network infrastructure
- Compliance Auditing: Generate comprehensive crypto inventories for regulatory requirements

Key Capabilities

- Network Scanning: TLS/SSL cipher suite enumeration and certificate discovery
- Filesystem Analysis: Discover certificates, private keys, and crypto files
- Memory Inspection: Identify loaded cryptographic libraries in running processes
- SSH Key Discovery: Enumerate SSH host keys and analyze key strength
- VPN Client Detection: Discover installed enterprise VPN clients with PQC assessments
- IPSec Tunnel Analysis: Detect and analyze IPSec tunnel configurations and security
- Multi-Platform: Native support for Windows, Linux, and macOS environments

Why Quantum Readiness Scanner Matters

As organizations face the imminent threat of quantum computing breaking current cryptographic standards, maintaining complete visibility into cryptographic assets has become critical. Quantum Readiness Scanner provides the comprehensive discovery and analysis capabilities needed to prepare for the post-quantum transition, ensure regulatory compliance, and maintain robust security postures across complex enterprise environments.

Quick Start

This user guide reviews multiple options for configuring your Quantum Readiness scans, optimizing your performance, and viewing your data. However, for those users who want to quickly begin scanning, below are the most common scan configurations with the simplest output options.

Remote Scanning

Scan external network infrastructure by connecting to remote hosts via TLS/SSL and SSH protocols. Ideal for network security assessments, external certificate monitoring, and infrastructure audits. Supports CIDR ranges, port ranges, wildcard domain enumeration, and bulk host scanning.

POWERSHELL:

```
# Basic remote scan
.\certscanner-windows-amd64.exe -license-key "..." -host example.com

# With cipher enumeration
.\certscanner-windows-amd64.exe -license-key "..." -host example.com -cipherscan

# Multiple hosts and ports
.\certscanner-windows-amd64.exe -license-key "..." -host example.com,google.com
-ports 443,22,8443

# Wildcard domain enumeration
.\certscanner-windows-amd64.exe -license-key "..." -host "*.company.com" -ports
443
```

BASH:

```
# Basic remote scan
./certscanner-linux-x64 -license-key "..." -host example.com

# With cipher enumeration
./certscanner-linux-x64 -license-key "..." -host example.com -cipherscan

# Multiple hosts and ports
./certscanner-linux-x64 -license-key "..." -host example.com,google.com -ports
443,22,8443

# Wildcard domain enumeration
./certscanner-linux-x64 -license-key "..." -host "*.company.com" -ports 443
```

BASH:

```
# Basic remote scan (Intel)
./certscanner-darwin-amd64 -license-key "..." -host example.com

# Basic remote scan (Apple Silicon)
./certscanner-darwin-arm64 -license-key "..." -host example.com

# With cipher enumeration
./certscanner-darwin-amd64 -license-key "..." -host example.com -cipherscan

# Multiple hosts and ports
```

```
./certscanner-darwin-amd64 -license-key "..." -host example.com,google.com -
ports 443,22,8443

# Wildcard domain enumeration
./certscanner-darwin-amd64 -license-key "..." -host "*.company.com" -ports 443
```

Local Scanning

Analyze the local system for cryptographic assets including filesystem certificates, running process memory, active network connections, and Outlook archives. Perfect for endpoint compliance validation, incident response, and comprehensive system crypto inventory.

POWERSHELL:

```
# Basic local scan
.\certscanner-windows-amd64.exe -license-key "..." -mode local

# Comprehensive local scan (single flag)
.\certscanner-windows-amd64.exe -license-key "..." -mode local -fullscan

# Alternative: Manual comprehensive scan
.\certscanner-windows-amd64.exe -license-key "..." -mode local -scanfilesystem -
scanmemory -scanconnected -scanoutlookarchives

# VPN & IPSec Detection
.\certscanner-windows-amd64.exe -license-key "..." -mode local -detect-vpn-
clients -detect-ipsec
```

BASH:

```
# Basic local scan
./certscanner-linux-x64 -license-key "..." -mode local

# Comprehensive local scan (single flag)
./certscanner-linux-x64 -license-key "..." -mode local -fullscan

# Alternative: Manual comprehensive scan
./certscanner-linux-x64 -license-key "..." -mode local -scanfilesystem -
scanmemory -scanconnected -scanoutlookarchives

# VPN & IPSec Detection
./certscanner-linux-x64 -license-key "..." -mode local -detect-vpn-clients -
detect-ipsec
```

BASH:

```
# Basic local scan (Intel)
./certscanner-darwin-amd64 -license-key "..." -mode local

# Basic local scan (Apple Silicon)
./certscanner-darwin-arm64 -license-key "..." -mode local

# Comprehensive local scan (single flag)
./certscanner-darwin-amd64 -license-key "..." -mode local -fullscan

# Alternative: Manual comprehensive scan
```

```
./certscanner-darwin-amd64 -license-key "..." -mode local -scanfilesystem -
scanmemory -scanconnected -scanoutlookarchives

# VPN & IPSec Detection
./certscanner-darwin-amd64 -license-key "..." -mode local -detect-vpn-clients -
detect-ipsec
```

Basic Installation

Below are the basic steps for getting started with Quantum Readiness. For more advanced installation steps or integrations, please contact your account representative.

Download Binary

Download the latest binary for your platform from the Partner Portal. The scanner is available for the following platforms:

- macOS- Intel (amd64) and Apple Silicon (arm64)
- Windows- 64-bit (amd64)
- Linux- 64-bit (x64)

Contact your account representative for Partner Portal access credentials.

License Configuration

A valid license key is required to activate the full scanning capabilities of Quantum Readiness Scanner. The license serves as both an authentication mechanism and feature enabler, ensuring authorized deployment across enterprise environments while unlocking comprehensive cryptographic discovery, remote scanning, third-party integrations, and advanced reporting capabilities. Without a license, the utility operates in a restricted trial mode suitable only for basic evaluation.

Trial Mode vs Licensed Mode

The scanner operates in trial mode by default with limited functionality. A license key unlocks all features, enterprise capabilities, and removes trial restrictions.

Trial Mode (No License)

- Single port scan only (port 443)
- Local mode only (no remote scanning)
- No filesystem scanning
- No memory scanning
- No VPN/IPSec detection
- No quantum readiness scoring
- JSON output only

Licensed Mode

- Remote scanning- Network infrastructure
- Filesystem scanning- All certificate stores
- Memory scanning- Running processes
- VPN/IPSec detection- Tunnel configuration
- Quantum readiness- PQC assessment
- Outlook archives- PST/OST scanning
- All output formats- JSON, HTML, CBOM, SIEM
- Unlimited hosts- Based on license tier

How to Apply Your License Key

The scanner supports multiple methods for license activation, listed in priority order:

Command-line Flag (Highest Priority)

Pass the license key directly as a command-line argument. Best for testing or one-time scans.

Windows:

```
.\certscanner-windows-amd64.exe -license-key "xxxxxxxxxxxxxxxxxxxx" -host
example.com
```

Linux/macOS:

```
./certscanner-linux-amd64 -license-key "xxxxxxxxxxxxxxxxxxxx" -host example.com
```

Environment Variable

Set the LICENSE_KEY environment variable. Recommended for automation and container deployments.

⚠️ **Windows Users Important Note:** Use $env:LICENSE_KEY = "..." for immediate use in the current session. If you use [Environment]::SetEnvironmentVariable(), you MUST close PowerShell completely and open a NEW window for the variable to be available. The permanent method does NOT work in the same session where you set it.

Windows (PowerShell):

```
# Option 1: Temporary (current session only) - Works immediately
$env:LICENSE_KEY = "xxxxxxxxxxxxxxxxxxxx"

# Option 2: Permanent (current user) - REQUIRES RESTART OF POWERSHELL
[Environment]::SetEnvironmentVariable("LICENSE_KEY", "xxxxxxxxxxxxxxxxxxxx",
"User")
# ⚠️ IMPORTANT: Close PowerShell completely and open a NEW window for this to
take effect!

# Option 3: Permanent AND works immediately (RECOMMENDED)
$key = "xxxxxxxxxxxxxxxxxxxx"
[Environment]::SetEnvironmentVariable("LICENSE_KEY", $key, "User")
$env:LICENSE_KEY = $key  # Also set for current session

# Verify it's set:
$env:LICENSE_KEY
```

Linux/macOS (Bash):

```
# Temporary (current session)
export LICENSE_KEY="xxxxxxxxxxxxxxxxxxxx"

# Permanent (add to ~/.bashrc or ~/.zshrc)
echo 'export LICENSE_KEY="xxxxxxxxxxxxxxxxxxxx"' >> ~/.bashrc
source ~/.bashrc
```

Docker/Kubernetes:

```
# Docker (Remote Mode Only)
docker pull tychoncorp/cryptographic-analyzer
docker run -e LICENSE_KEY="xxxxxxxxxxxxxxxxxxxx" tychoncorp/cryptographic-
```

```
analyzer

# Kubernetes Secret
kubectl create secret generic license --from-literal=license-
key="xxxxxxxxxxxxxxxxxxxx"
```

Docker Hub: see tychoncorp/cryptographic-analyzer

User Configuration File

Store the license in your home directory. Best for individual user workstations.
Location:

- Windows:C:\Users\YourName\.the scanner\license.key
- Linux/macOS:~/.the scanner/license.key

Setup:

```
# Windows (PowerShell)
New-Item -ItemType Directory -Force -Path "$env:USERPROFILE\.tychon"
Set-Content -Path "$env:USERPROFILE\.tychon\license.key" -Value
"xxxxxxxxxxxxxxxxxxxx"

# Linux/macOS
mkdir -p ~/.tychon
echo "xxxxxxxxxxxxxxxxxxxx" > ~/.tychon/license.key
chmod 600 ~/.tychon/license.key
```

System Configuration File (Lowest Priority)

System-wide license for all users. Requires administrator/root privileges. Best for enterprise deployments.
Location:

- Windows:C:\ProgramData\the scanner\license.key
- Linux/macOS:/etc/the scanner/license.key

Setup:

```
# Linux/macOS (requires root)
sudo mkdir -p /etc/tychon
echo "xxxxxxxxxxxxxxxxxxxx" | sudo tee /etc/tychon/license.key
sudo chmod 644 /etc/tychon/license.key
```

License Information

- Grace Period: Licenses include a 60-day grace period after expiration date
- License Format: Keys are 52 characters
- Validation: License keys are validated locally - no internet connection required

Troubleshooting Windows Environment Variables

If the LICENSE_KEY environment variable is not working on Windows, follow these steps:

1. VERIFY THE ENVIRONMENT VARIABLE IS SET

Open a new PowerShell window and check if the variable exists:

```
# Check if the variable is set
$env:LICENSE_KEY

# Should output your license key, e.g.: xxxxxxxxxxxxxxxxxxxx
# If it outputs nothing, the variable is not set
```

Most Common Issue: If you used [Environment]::SetEnvironmentVariable(), you are likely still in the same PowerShell session where you set it. This method sets the registry value but does NOT update the current session. You MUST close all PowerShell windows and open a completely new one.

2. COMMON ISSUES AND SOLUTIONS

- **Issue**: Variable only works in the current PowerShell session# Solution: Set permanently for current user
  [Environment]::SetEnvironmentVariable("LICENSE_KEY", "xxxxxxxxxxxxxxxxxxxx", "User")

  # Then close and reopen PowerShell to apply changes

- **Issue**: Running executable from Command Prompt (cmd.exe) instead of PowerShell# Solution: Set system-wide (requires Administrator privileges)
  # Open PowerShell as Administrator, then run:
  [Environment]::SetEnvironmentVariable("LICENSE_KEY", "xxxxxxxxxxxxxxxxxxxx", "Machine")

  # Or set in cmd.exe for current session:
  set LICENSE_KEY=xxxxxxxxxxxxxxxxxxxx

- **Issue**: Variable not available after setting it permanently# Solution: You MUST close and reopen your terminal/PowerShell window
  # Environment variables are loaded when the shell starts
  # Simply setting it in one window won't affect other windows already open

- **Issue**: Running from a different shell than where you set it# Solution: Use the -license-key flag instead
  .\certscanner-windows-amd64.exe -license-key "xxxxxxxxxxxxxxxxxxxx" -host example.com

Best Practice for Windows: If environment variables continue to cause issues, use the file-based method instead (Method 3: User Configuration File). This is more reliable across different shells and terminal sessions.

Command Reference

When you are ready to begin scanning, you can configure your scan with various commands to customize the scope, output, features, and more. The following section provides reference tables with the available commands.

Core Options

Flag	Description	Environment Variable	Default
-mode	Scan mode: 'remote' or 'local'	SCAN_MODE	remote
-license-key	License key for full feature access (trial mode if not provided)	LICENSE_KEY	-
-version	Display version information and exit	-	-
-config	Enter configuration mode to securely store credentials	-	-
-disable-quantum-readiness	Disable quantum readiness assessment	-	false

Scanning Options (Both Modes)

Flag	Description	Environment Variable	Default
-ports	Ports to scan (comma-separated, ranges like 1000-1024)	PORTS	443 (remote)
-quickscan	Quick scan mode: only report preferred connection per port	-	true
-cipherscan	Perform detailed cipher suite enumeration (for TLS connections)	-	false
-experimental-go-probes	**Preferred:** Use hybrid Go-based cipher probes for faster scans (2x speed, 95% reduction in OpenSSL executions, reduces EDR alerts). Automatically falls back to legacy scan on error.		false
-cputhrottle	CPU throttling level: none, low, medium, high (controls concurrency for resource management)	-	medium
-include-empty-ports	Include ports even if no TLS or SSH detected	-	false
-disable-quantum-readiness	Disable host (os + hardware) quantum readiness assessment	-	false

Remote Mode Features

Flag	Description	Environment Variable	Default
-host	Target host(s)/IP(s)/CIDR/Range/Wildcard domains	HOST	-
-exclude-file	Path to file containing IPs/ranges/CIDRs to exclude from scan	-	-

Local Mode Features

Flag	Description	Environment Variable	Platform Support
-fullscan	Enable comprehensive local scanning: cipherscan, scanmemory, scanfilesystem, scanoutlookarchives, detect-vpn-clients, detect-ipsec	-	All platforms
-scanmemory	Scan process memory for cryptographic libraries	-	Windows, Linux (basic)
-scanconnected	Scan active external connections	-	All platforms
-scanfilesystem	Scan filesystem for certificate files	-	All platforms
-scanoutlookarchives	Scan for encrypted Outlook archives (.pst, .ost, .pab)	-	All platforms
-arpscan	Scan IPs from local ARP table (22,443,8443,etc.)	-	All platforms
-detect-vpn-clients	Discover enterprise VPN clients with PQC assessments PRE-RELEASE	-	All platforms
-detect-ipsec	Detect IPSec tunnel configurations and security analysis PRE-RELEASE	-	All platforms

Note: Features marked as PRE-RELEASE may have inaccuracies or incomplete functionality.

Output & Reporting

Flag	Description	Environment Variable	Default
-output	File to save report	-	scan_report.json
-outputformat	json, flatndjson, cbom, the scanner, eventlog, html	OUTPUT_FORMAT	json

-split-outputs	Split output into separate files per dataset (quantum, network, memory, filesystem, keystore, outlook, vpn, ipsec)		False
-keep-consolidated	Keep consolidated file when using -split-outputs (creates both split files AND main file)		false
-detail-level	Output verbosity: 'full' (all fields, 0% reduction), 'standard' (~30-40% smaller), 'minimal' (~60-70% smaller)		full
-logfile	Path to detailed log file	-	-
-tags	Custom tags (comma-separated: prod,webserver,critical)	-	-

Integration & Tracking

Flag	Description	Environment Variable	Default
-disable-database	Disable tracking for active/inactive status	-	false
-bolt-path	Path to BoltDB tracking database file	-	./scan_tracking.db
-posttoelastic	Post report to Elasticsearch	-	false
-elasticnode	Elasticsearch URL (https://localhost:9200)	ELASTIC_HOST	-
-elasticapikey	Elasticsearch API Key	ELASTIC_USERNAME:ELASTIC_PASSWORD	-
-elasticindex	Elasticsearch index suffix (adds to pattern: the scanner-pqc-{dataset}_{elasticindex})	-	the scanner-pqc-{dataset}
-posttokafka	Post report to Kafka	-	false
-kafkabrokers	Kafka broker addresses (broker1:9092,broker2:9092)	KAFKA_BROKERS	-
-kafkatopic	Kafka topic name for events	KAFKA_TOPIC	tychon-crypto-assets
-kafkausername	Kafka SASL username	KAFKA_USERNAME	-
-kafkapassword	Kafka SASL password	KAFKA_PASSWORD	-
-kafkasecurityprotocol	Security protocol (PLAINTEXT, SSL,	KAFKA_SECURITY_PROTOCOL	-

	SASL_PLAINTEXT, SASL_SSL)		
-kafkasaslmechanism	SASL mechanism (PLAIN, SCRAM-SHA-256, SCRAM-SHA-512)	KAFKA_SASL_MECHANISM	PLAIN
-kafkasslcalocation	Kafka SSL CA certificate file path	KAFKA_SSL_CA_LOCATION	-
-kafkasslcertlocation	Kafka SSL client certificate file path	KAFKA_SSL_CERT_LOCATION	-
-kafkasslkeylocation	Kafka SSL client private key file path	KAFKA_SSL_KEY_LOCATION	-
-kafkasslkeypassword	Password for encrypted SSL client private key	KAFKA_SSL_KEY_PASSWORD	-
-kafkasslkeystorelocation	Kafka SSL keystore file path (JKS format)	KAFKA_SSL_KEYSTORE_LOCATION	-
-kafkasslkeystorepassword	Password for Kafka SSL keystore	KAFKA_SSL_KEYSTORE_PASSWORD	-
-kafkassltruststorelocation	Kafka SSL truststore file path (JKS format)	KAFKA_SSL_TRUSTSTORE_LOCATION	-
-kafkassltruststorepassword	Password for Kafka SSL truststore	KAFKA_SSL_TRUSTSTORE_PASSWORD	-
-kafkasslenabledprotocols	Comma-separated list of enabled SSL protocols	KAFKA_SSL_ENABLED_PROTOCOLS	TLSv1.2,TLSv1.3
-kafkasslendpointidentificationalgorithm	SSL endpoint identification algorithm	KAFKA_SSL_ENDPOINT_IDENTIFICATION_ALGORITHM	-
-kafkaclientid	Kafka client ID (defaults to hostname)	-	-
-posttosplunk	Post report to Splunk HEC	-	false
-splunkurl	Splunk server URL (https://splunk.company.com:8088)	SPLUNK_URL	-
-splunktoken	Splunk HEC authentication token	SPLUNK_TOKEN	-
-splunkusername	Splunk basic auth username (alternative to token)	SPLUNK_USERNAME	-
-splunkpassword	Splunk basic auth password (alternative to token)	SPLUNK_PASSWORD	-
-splunkindex	Splunk index name for events	SPLUNK_INDEX	tychon-crypto
-splunksource	Splunk source name	SPLUNK_SOURCE	tychon-scanner

-splunksourcetype	Splunk source type	SPLUNK_SOURCETYPE	the scanner-acdi:crypto_assets
-splunkbatch	Batch size for HEC events	SPLUNK_BATCH	100
-splunktimeout	HEC request timeout in seconds	SPLUNK_TIMEOUT	30
-upload-s3	Upload report file to S3	-	false
-s3bucket	S3 bucket name for uploads	S3_BUCKET	-
-s3region	S3 region for bucket access	S3_REGION	us-east-1
-s3prefix	S3 key prefix for organization	S3_PREFIX	-
-s3accesskey	AWS Access Key ID for S3 authentication	AWS_ACCESS_KEY_ID	-
-s3secretkey	AWS Secret Access Key for S3 authentication	AWS_SECRET_ACCESS_KEY	-
-s3endpoint	Custom S3 endpoint URL (for R2, MinIO, etc.)	-	-
-insecure	Skip SSL certificate verification for Elasticsearch, Kafka, and Splunk connections	-	false

Secure Configuration (FIPS 140-3)

Flag	Description	Environment Variable
-config	Configure and encrypt credentials for reuse	-
-config-elasticnode	Elasticsearch node URL to store encrypted	-
-config-elasticapikey	Elasticsearch API Key to store encrypted	-
-config-kafkabrokers	Kafka broker addresses to store encrypted	-
-config-kafkausername	Kafka SASL username to store encrypted	-
-config-kafkapassword	Kafka SASL password to store encrypted	-
-config-kafkasecurityprotocol	Kafka security protocol (PLAINTEXT, SSL, SASL_PLAINTEXT, SASL_SSL) to store encrypted	-
-config-kafkasaslmechanism	Kafka SASL mechanism (PLAIN, SCRAM-SHA-256,	-

	SCRAM-SHA-512) to store encrypted	
-config-kafkasslcalocation	Kafka SSL CA certificate path to store encrypted	-
-config-kafkasslcertlocation	Kafka SSL client certificate path to store encrypted	-
-config-kafkasslkeylocation	Kafka SSL client private key path to store encrypted	-
-config-kafkasslkeypassword	Kafka SSL client private key password to store encrypted	-
-config-kafkasslkeystorelocation	Kafka SSL keystore file path to store encrypted	-
-config-kafkasslkeystorepassword	Kafka SSL keystore password to store encrypted	-
-config-kafkassltruststorelocation	Kafka SSL truststore file path to store encrypted	-
-config-kafkassltruststorepassword	Kafka SSL truststore password to store encrypted	-
-config-kafkasslenabledprotocols	Kafka SSL enabled protocols list to store encrypted	-
-config-kafkasslendpointidentificationalgorithm	Kafka SSL endpoint identification algorithm to store encrypted	-
-config-s3region	S3 region to store encrypted	-
-config-s3accesskey	S3 Access Key to store encrypted	-
-config-s3secretkey	S3 Secret Key to store encrypted	-
-config-s3endpoint	S3 endpoint URL to store encrypted	-
-config-webapikey	Web API Key to store encrypted	-

Usage Examples

This section provides examples for common scans combining the available command options. All commands below assume a valid license key is configured.

Network Security Assessment

Scan remote hosts and networks to identify TLS configurations, detect post-quantum vulnerable cipher suites, and assess quantum readiness across your infrastructure. Supports CIDR ranges, multiple hosts, and comprehensive cipher enumeration.

POWERSHELL:

```
# Comprehensive network scan with PQC detection
.\certscanner-windows-amd64.exe -host 192.168.1.0/24 -ports 443,22,8443,993,995
-cipherscan -outputformat html

# Quick security assessment
.\certscanner-windows-amd64.exe -host example.com,mail.example.com -quickscan -
outputformat json

# Resource-friendly cipher scan (low CPU usage)
.\certscanner-windows-amd64.exe -host example.com -cipherscan -cputhrottle low -
outputformat json

# Generate CBOM for compliance
.\certscanner-windows-amd64.exe -host example.com -cipherscan -outputformat cbom
-output compliance-report.cbom.json
```

BASH:

```
# Comprehensive network scan with PQC detection
./certscanner-linux-x64 -host 192.168.1.0/24 -ports 443,22,8443,993,995 -
cipherscan -outputformat html

# Quick security assessment
./certscanner-linux-x64 -host example.com,mail.example.com -quickscan -
outputformat json

# Resource-friendly cipher scan (low CPU usage)
./certscanner-linux-x64 -host example.com -cipherscan -cputhrottle low -
outputformat json

# Generate CBOM for compliance
./certscanner-linux-x64 -host example.com -cipherscan -outputformat cbom -output
compliance-report.cbom.json
```

BASH:

```
# Comprehensive network scan with PQC detection
./certscanner-darwin-amd64 -host 192.168.1.0/24 -ports 443,22,8443,993,995 -
cipherscan -outputformat html

# Quick security assessment
./certscanner-darwin-amd64 -host example.com,mail.example.com -quickscan -
outputformat json
```

```
# Resource-friendly cipher scan (low CPU usage)
./certscanner-darwin-amd64 -host example.com -cipherscan -cputhrottle low -
outputformat json

# Generate CBOM for compliance
./certscanner-darwin-amd64 -host example.com -cipherscan -outputformat cbom -
output compliance-report.cbom.json
```

Local System Audit

Inventory all certificates and cryptographic libraries on the local system. Scans certificate stores, filesystem paths, running process memory, Outlook archives (PST/OST), and active TLS connections. Ideal for endpoint compliance and vulnerability assessment.

POWERSHELL:

```
# Complete system audit
.\certscanner-windows-amd64.exe -mode local -scanfilesystem -scanmemory -
scanconnected -scanoutlookarchives -outputformat html

# Focus on cryptographic libraries
.\certscanner-windows-amd64.exe -mode local -scanmemory -outputformat flatndjson

# Certificate inventory
.\certscanner-windows-amd64.exe -mode local -scanfilesystem -outputformat cbom
```

BASH:

```
# Complete system audit
./certscanner-linux-x64 -mode local -scanfilesystem -scanmemory -scanconnected -
scanoutlookarchives -outputformat html

# Focus on cryptographic libraries
./certscanner-linux-x64 -mode local -scanmemory -outputformat flatndjson

# Certificate inventory
./certscanner-linux-x64 -mode local -scanfilesystem -outputformat cbom
```

BASH:

```
# Complete system audit
./certscanner-darwin-amd64 -mode local -scanfilesystem -scanconnected -
scanoutlookarchives -outputformat html

# Certificate inventory
./certscanner-darwin-amd64 -mode local -scanfilesystem -outputformat cbom

# Note: Memory scanning not available on macOS
```

SIEM Integration

Stream scan results directly to security information and event management (SIEM) platforms. Supports Elasticsearch, Splunk, Kafka, and native Windows EventLog. Use flat NDJSON format for log aggregators and real-time streaming for monitoring dashboards.

POWERSHELL:

```
# Stream to Elasticsearch
.\certscanner-windows-amd64.exe -host example.com -posttoelastic -elasticnode
```

```
"https://elastic.company.com:9200" -elasticapikey "key"

# Stream to Kafka (real-time events)
.\certscanner-windows-amd64.exe -host example.com -posttokafka -kafkabrokers
"kafka1:9092,kafka2:9092" -kafkatopic "crypto-events"

# Windows EventLog integration
.\certscanner-windows-amd64.exe -mode local -outputformat eventlog

# Flat format for log aggregation
.\certscanner-windows-amd64.exe -host example.com -outputformat flatndjson -
output C:\logs\crypto-scan.ndjson
```

BASH:

```
# Stream to Elasticsearch
./certscanner-linux-x64 -host example.com -posttoelastic -elasticnode
"https://elastic.company.com:9200" -elasticapikey "key"

# Stream to Kafka (real-time events)
./certscanner-linux-x64 -host example.com -posttokafka -kafkabrokers
"kafka1:9092,kafka2:9092" -kafkatopic "crypto-events"

# Flat format for log aggregation
./certscanner-linux-x64 -host example.com -outputformat flatndjson -output
/var/log/crypto-scan.ndjson
```

BASH:

```
# Stream to Elasticsearch
./certscanner-darwin-amd64 -host example.com -posttoelastic -elasticnode
"https://elastic.company.com:9200" -elasticapikey "key"

# Stream to Kafka (real-time events)
./certscanner-darwin-amd64 -host example.com -posttokafka -kafkabrokers
"kafka1:9092,kafka2:9092" -kafkatopic "crypto-events"

# Flat format for log aggregation
./certscanner-darwin-amd64 -host example.com -outputformat flatndjson -output
/var/log/crypto-scan.ndjson
```

Secure Configuration (FIPS 140-3)

Securely store API keys, credentials, and secrets using FIPS 140-3 certified encryption. Configure once with the-configflag to encrypt and save credentials locally, then run scans without exposing sensitive data in command-line arguments or environment variables.

POWERSHELL:

```
# One-time credential setup (FIPS 140-3 encrypted storage)
# Example: Configuring Elasticsearch integration
.\certscanner-windows-amd64.exe -config `
  -config-elasticnode "https://elastic.company.com:9200" `
  -config-elasticapikey "your-elastic-api-key"

# Simplified usage afterwards - credentials loaded automatically
```

```
.\certscanner-windows-amd64.exe -host example.com -posttoelastic -elasticindex
"production"
```

BASH:

```
# One-time credential setup (FIPS 140-3 encrypted storage)
# Example: Configuring Elasticsearch integration
./certscanner-linux-x64 -config \
  -config-elasticnode "https://elastic.company.com:9200" \
  -config-elasticapikey "your-elastic-api-key"

# Simplified usage afterwards - credentials loaded automatically
./certscanner-linux-x64 -host example.com -posttoelastic -elasticindex
"production"
```

BASH:

```
# One-time credential setup (FIPS 140-3 encrypted storage)
# Example: Configuring Elasticsearch integration
./certscanner-darwin-amd64 -config \
  -config-elasticnode "https://elastic.company.com:9200" \
  -config-elasticapikey "your-elastic-api-key"

# Simplified usage afterwards - credentials loaded automatically
./certscanner-darwin-amd64 -host example.com -posttoelastic -elasticindex
"production"
```

Output Features: Split Outputs & Detail Levels

Control output organization and size with split outputs (separate files per dataset) and detail levels (reduce verbosity). Perfect for real-time monitoring, bandwidth-constrained environments, and SIEM integration.

Windows

```
# Split outputs - separate files per dataset
.\certscanner-windows-amd64.exe -mode local -split-outputs -output report.json

# Result: Creates 8 separate files (skips empty datasets)
#   report_quantum.json, report_network.json, report_memory.json, etc.

# Minimal detail level - ~60-70% size reduction
.\certscanner-windows-amd64.exe -mode local -detail-level minimal -output
report.json

# Combined: Maximum efficiency for real-time monitoring
.\certscanner-windows-amd64.exe -mode local -split-outputs -detail-level minimal
-output report.json

# Standard detail for SIEM integration (~30-40% smaller)
.\certscanner-windows-amd64.exe -host example.com -split-outputs -detail-level
standard -output report.json

# Keep both split and consolidated files
.\certscanner-windows-amd64.exe -mode local -split-outputs -keep-consolidated -
output report.json
```

Linux

```
# Split outputs - separate files per dataset
./certscanner-linux-x64 -mode local -split-outputs -output report.json

# Result: Creates 8 separate files (skips empty datasets)
#   report_quantum.json, report_network.json, report_memory.json, etc.

# Minimal detail level - ~60-70% size reduction
./certscanner-linux-x64 -mode local -detail-level minimal -output report.json

# Combined: Maximum efficiency for real-time monitoring
./certscanner-linux-x64 -mode local -split-outputs -detail-level minimal -output
report.json

# Standard detail for SIEM integration (~30-40% smaller)
./certscanner-linux-x64 -host example.com -split-outputs -detail-level standard
-output report.json

# Keep both split and consolidated files
./certscanner-linux-x64 -mode local -split-outputs -keep-consolidated -output
report.json
```

macOS

```
# Split outputs - separate files per dataset
./certscanner-darwin-arm64 -mode local -split-outputs -output report.json

# Result: Creates 8 separate files (skips empty datasets)
#   report_quantum.json, report_network.json, report_memory.json, etc.

# Minimal detail level - ~60-70% size reduction
./certscanner-darwin-arm64 -mode local -detail-level minimal -output report.json

# Combined: Maximum efficiency for real-time monitoring
./certscanner-darwin-arm64 -mode local -split-outputs -detail-level minimal -
output report.json

# Standard detail for SIEM integration (~30-40% smaller)
./certscanner-darwin-arm64 -host example.com -split-outputs -detail-level
standard -output report.json

# Keep both split and consolidated files
./certscanner-darwin-arm64 -mode local -split-outputs -keep-consolidated -output
report.json
```

PQC Discovery Details

Quantum Readiness Scanner provides comprehensive detection and analysis of post-quantum cryptographic implementations, preparing your infrastructure for the quantum-safe future.

Key Exchange Algorithms (ML-KEM / Kyber)

Pure ML-KEM

- MLKEM512
- MLKEM768
- MLKEM1024
- KYBER*(legacy)

MLKEM512 Hybrids

- X25519MLKEM512
- P256MLKEM512
- P384MLKEM512
- P521MLKEM512
- SECP*MLKEM512

MLKEM768 Hybrids

- X25519MLKEM768
- P256MLKEM768
- P384MLKEM768
- P521MLKEM768
- SECP*MLKEM768

MLKEM1024 Hybrids

- X25519MLKEM1024
- P256MLKEM1024
- P384MLKEM1024
- P521MLKEM1024
- SECP*MLKEM1024

Classical Algorithms

- P-256, P-384, P-521
- X25519, X448
- DH-1024 to DH-4096
- SECP curves

Key Analysis Features

- Public key size tracking
- Hybrid key composition
- Underscore normalization
- Pattern-based extraction

Digital Signature Algorithms

Falcon (NIST FIPS 206)

- falcon512
- falcon1024
- falconpadded512
- falconpadded1024

Dilithium (NIST FIPS 204)

- dilithium2
- dilithium3
- dilithium5

MAYO (Multivariate)

- mayo1
- mayo2
- mayo3
- mayo5

SPHINCS+ (Hash-based)

- sphincssha2*
- sphincsshake*
- 128/192/256 variants
- fsimple/ssimple modes

Complete Cipher Suite Intelligence Database

Quantum Readiness online documentation provides a comprehensive, searchable database of 300+ cipher suites with detailed security analysis, quantum readiness assessments, and compliance information.

Cipher Suite Discovery Details

Overview

Cipher Suite Enumeration

The-cipherscanflag enables comprehensive cipher suite enumeration for TLS connections and key exchange algorithm detection for SSH connections. This feature performs active testing to discover all supported cryptographic configurations on remote services.

TLS Detection

87+ cipher suites across TLS 1.0-1.3, including modern AEAD and legacy CBC modes

SSH Detection

Key exchange, host key algorithms, encryption ciphers, and MAC algorithms

PQC Support

Native detection of ML-DSA, ML-KEM, and hybrid post-quantum algorithms

Important: Cipher scanning performs multiple connection attempts to enumerate all supported cipher suites. This may trigger security monitoring systems and should only be performed with proper authorization.

What Gets Detected

TLS Connections

- Supported cipher suites (TLS 1.0-1.3)
- Key exchange algorithms (ECDHE, DHE, RSA, PQC)
- Ephemeral key lengths
- Negotiated groups (curves)
- Signature algorithms
- Protocol versions
- Session parameters

SSH Connections

- Key exchange algorithms (KEX)
- Host key algorithms
- Encryption ciphers
- MAC algorithms
- Compression algorithms
- SSH protocol version

Usage Examples

Basic Cipher Scanning

BASH:

```
# Basic cipher enumeration
./certscanner -host example.com -cipherscan

# Scan specific ports
```

```
./certscanner -host example.com -ports 443,8443,993 -cipherscan

# Scan multiple hosts with cipher detection
./certscanner -host servers.txt -cipherscan -outputformat json
```

Performance Management

BASH:

```
# Conservative resource usage (recommended for production)
./certscanner -host example.com -cipherscan -cputhrottle low

# Balanced performance (default)
./certscanner -host example.com -cipherscan -cputhrottle medium

# Maximum performance
./certscanner -host example.com -cipherscan -cputhrottle high
```

Tip: Use-cputhrottle low for production environments to minimize resource impact. Cipher scanning can be resource-intensive as it tests multiple cipher suite combinations.

Quantum Readiness Scoring

Learn how Quantum Readiness Scanner evaluates your organization's readiness for the post-quantum cryptography transition through comprehensive scoring algorithms.

Overview

The Quantum Readiness Scoring System evaluates a system's preparedness for the post-quantum cryptography (PQC) era. Using a comprehensive 100-point scale, it assesses hardware capabilities, operating system support, cryptographic libraries, and network readiness to provide actionable insights and upgrade recommendations.

Local Mode Only

Quantum readiness assessment is only available during local mode scans (-mode local). It can be disabled with the -disable-quantum-readiness flag.

Assessment Categories

- **Hardware Assessment**: CPU, memory, security hardware (40 points)
- **Operating System**: Version support, crypto frameworks (30 points)
- **Crypto Libraries**: OpenSSL, system crypto support (25 points)
- **Network Readiness**: Bandwidth, protocol capabilities (5 points)

Readiness Levels

- **Ready**: 88-92+ points
- **Partially Ready**: 68-75+ points
- **Update Required**: 45-55+ points
- **Not Ready**: <45-55 points

Scoring Methodology

Hardware Assessment (40 Points)

CPU Capabilities (20 Points)

- **Architecture**: 64-bit required (0 points for 32-bit)
- **Instruction Sets**: AES-NI, AVX2, BMI support
- **Core Count**: Multi-core performance scaling
- **CPU Generation**: Modern architecture support

Memory Capacity (15 Points)

- **8GB+**: 8-10 points
- **16GB+**: 12-13 points
- **32GB+**: 15 points (server threshold)
- **64GB+**: 15 points (enterprise)

Security Hardware (5 Points)

- **TPM Module**: Hardware security support
- **Secure Boot**: Boot integrity verification
- **Hardware RNG**: True random number generation
- **Enclave Support**: Intel SGX, ARM TrustZone

Operating System Assessment (30 Points)

Version Support (20 Points)

macOS Scoring:

- **15.0+ (Sequoia)**: 20 points
- **14.0+ (Sonoma)**: 15 points
- **13.0+ (Ventura)**: 12 points
- **12.0+ (Monterey)**: 8 points
- **11.0+ (Big Sur)**: 5 points
- **10.15+ (Catalina)**: 2 points
- **Older versions**: 0 points

Windows Scoring:

- **Windows 11 24H2+**: 20 points
- **Windows 11**: 15 points
- **Windows 10 22H2**: 10 points
- **Windows 10 older**: 5 points
- **Windows 8.1 or older**: 0 points

Linux Scoring:

- **Kernel 6.0+**: 20 points
- **Kernel 5.15+**: 15 points
- **Kernel 5.4+**: 10 points
- **Kernel 4.x**: 5 points
- **Older kernels**: 0 points

Crypto Framework (10 Points)

- **Modern Framework**: Native PQC APIs available
- **Crypto Libraries**: System-level crypto support
- **API Compatibility**: PKCS#11, CNG, Security.framework
- **Hardware Integration**: HSM and TPM support

Crypto Libraries Assessment (25 Points)

OpenSSL Version (15 Points)

- **3.4.0+**: 15 points (Full PQC support)
- **3.3.0+**: 12 points (Experimental PQC)
- **3.2.0+**: 10 points (Limited PQC)
- **3.1.x**: 8 points
- **3.0.x**: 6 points
- **1.1.1**: 3 points (Legacy)
- **1.1.0 or older**: 0 points

System Crypto (10 Points)

Platform-Specific:

- **macOS**: Security.framework, CommonCrypto
- **Windows**: CNG, CAPI, Schannel
- **Linux**: libgcrypt, NSS, kernel crypto

Assessment Factors:

- Library version and PQC readiness
- Algorithm support coverage

- Performance optimization level
- Integration with system services

Network Readiness Assessment (5 Points)

Bandwidth Capacity (3 Points)

- **Gigabit+**: 3 points
- **100 Mbps+**: 2 points
- **10 Mbps+**: 1 point
- **Below 10 Mbps**: 0 points

Protocol Support (2 Points)

- **TLS 1.3**: Modern protocol support
- **HTTP/2, HTTP/3**: Advanced protocols
- **IPv6**: Next-generation networking
- **QoS Support**: Traffic prioritization

System Classification & Thresholds

Workstation Systems

Classification Criteria:

- Desktop or laptop computers
- Single-user or personal systems
- Platform family contains "workstation" or "standalone"
- RAM < 32GB (typical threshold)

Readiness Thresholds:

- Ready 88-100 points
- Partially Ready 68-87 points
- Update Required 45-67 points
- Not Ready 0-44 points

Server Systems

Classification Criteria:

- Enterprise or data center systems
- Multi-user or service systems
- High-performance hardware specifications
- RAM ≥ 32GB (typical threshold)

Readiness Thresholds:

- Ready 92-100 points
- Partially Ready 75-91 points
- Update Required 55-74 points
- Not Ready 0-54 points

Criticality Levels

- Critical: Mission-critical systems requiring immediate PQC readiness
- Important: Business-important systems with higher security requirements

- Standard: Standard business systems with normal security needs

Critical Failure Conditions

Certain system characteristics result in automatic score reductions or caps, reflecting fundamental incompatibilities with post-quantum cryptography requirements.

Blocking Conditions (0 Points)

- 32-bit Architecture: Cannot support PQC key sizes and operations
- Extremely Low Memory (<4GB): Insufficient for PQC algorithm execution

Score Limitations

- Legacy OS Versions: Score capped at 70% of maximum possible
- Outdated Crypto Libraries: Heavy penalties applied to crypto scoring

Recommendations & Upgrade Pathways

Immediate Actions (High Impact)

- **Operating System Update**: Upgrade to latest OS version with PQC framework support
- **OpenSSL Upgrade**: Update to OpenSSL 3.4.0+ for full post-quantum algorithm support
- **Memory Upgrade**: Increase RAM to recommended levels (16GB+ workstation, 32GB+ server)
- **Architecture Migration**: Replace 32-bit systems with 64-bit alternatives

Medium-Term Improvements

- **Hardware Security**: Enable TPM, Secure Boot, hardware RNG capabilities
- **Network Infrastructure**: Upgrade to gigabit networking, implement TLS 1.3
- **Crypto Framework**: Integrate modern cryptographic APIs and libraries
- **CPU Upgrade**: Modernize processors with AES-NI, AVX2 instruction sets

Long-Term Strategy

- **PQC Algorithm Testing**: Implement and test NIST-approved algorithms
- **Performance Optimization**: Tune systems for PQC cryptographic workloads
- **Security Policy Updates**: Develop quantum-safe cryptographic policies
- **Training & Documentation**: Prepare teams for post-quantum transition

Timeline Estimates

- Ready (88-92+): Immediate deployment
- Partially Ready (68-75+): 2-6 months
- Update Required (45-55+): 6-12 months
- Not Ready (<45-55): 12+ months

Technical Integration

Command Line Usage

```
# Enable quantum readiness assessment (default in local mode)
./certscanner -mode local -output report.json

# Disable quantum readiness assessment
./certscanner -mode local -disable-quantum-readiness -output report.json
```

```
# View quantum assessment in different formats
./certscanner -mode local -outputformat json -output quantum.json
./certscanner -mode local -outputformat tychon -output quantum.ndjson
./certscanner -mode local -outputformat html -output quantum.html
```

Output Integration

- **JSON**: quantum_readiness field in scan report
- **NDJSON**: Flattened quantum.* fields in all events
- **Tychon**: ECS-compliant quantum assessment events
- **HTML**: Interactive scoring dashboard
- **EventLog**: Event ID 1005 for quantum assessments

API Data Structure


```
{
  "quantum_readiness": {
    "assessment_id": "qr_20250915_101539_abc123",
    "timestamp": "2025-09-15T10:15:39.123456-07:00",
    "system_type": "workstation",
    "overall_score": 64,
    "readiness_status": "Update Required",
    "status_color": "orange",
    "hardware_score": {
      "total_score": 32,
      "cpu_score": 18,
      "memory_score": 14,
      "security_hardware_score": 0
    },
    "operating_system_score": {
      "total_score": 15,
      "version_score": 12,
      "crypto_framework_score": 3
    },
    "crypto_library_score": {
      "total_score": 12,
      "openssl_score": 8,
      "system_crypto_score": 4
    },
    "network_score": {
      "total_score": 5,
      "bandwidth_score": 3,
      "protocol_capability_score": 2
    },
    "recommendations": [
      "Upgrade to macOS 15.0+",
      "Update OpenSSL to 3.4.0+"
    ]
  }
}
```

Platform Support for Scans

Platform	TLS	SSH	Local Discovery	Connected Scan	Memory Scan	Filesystem	ARP Scan	VPN Detection	IPSec Detection	Quantum Readiness
Windows 11 (x64)	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
Windows 10 (x64)	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
Windows Server 2016+ (x64)	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
macOS Monterey+ (Intel x64)	✅	✅	✅	✅	❌	✅	✅	✅	✅	✅
macOS Monterey+ (Apple Silicon ARM64)	✅	✅	✅	✅	❌	✅	✅	✅	✅	✅
Ubuntu 20.04+ (x64)	✅	✅	✅	✅	⚠️	✅	✅	✅	✅	✅
RHEL/CentOS 7+ (x64)	✅	✅	✅	✅	⚠️	✅	✅	✅	✅	✅
Debian 11+ (x64)	✅	✅	✅	✅	⚠️	✅	✅	✅	✅	✅
SUSE Enterprise 15+ (x64)	✅	✅	✅	✅	⚠️	✅	✅	✅	✅	✅
Alpine Linux 3.15+ (x64)	✅	✅	✅	✅	⚠️	✅	✅	✅	✅	✅
Amazon Linux 2 (x64)	✅	✅	✅	✅	⚠️	✅	✅	✅	✅	✅
Fedora 35+ (x64)	✅	✅	✅	✅	⚠️	✅	✅	✅	✅	✅

Windows Family

- Windows 10 (build 1909+)
- Windows 11 (all builds)
- Windows Server 2016
- Windows Server 2019
- Windows Server 2022
- Windows Server 2025
- Windows Server Core

Full feature support including memory scanning

macOS Family

- macOS Monterey (12.0+)

- macOS Ventura (13.0+)
- macOS Sonoma (14.0+)
- macOS Sequoia (15.0+)

Intel x64 and Apple Silicon ARM64

Linux Distributions

- Ubuntu 20.04 LTS+
- RHEL/CentOS 7+
- Debian 11 (Bullseye)+
- SUSE Enterprise 15+
- Alpine Linux 3.15+
- Amazon Linux 2
- Fedora 35+

Notes:

- Memory scanning requires elevated privileges
- VPN Detection: Identifies installed VPN clients
- IPSec Detection: Discovers IPSec tunnel configurations
- Quantum Readiness: 100-point scoring system for PQC preparedness (local mode only)

Basic Performance Optimization

CPU Throttling for Resource Management

The scanner includes intelligent CPU throttling controls to manage system resource usage during intensive cipher enumeration scans.

Throttling Levels

- none- Original behavior (up to 25 concurrent workers)
- low- 50% of CPU cores, conservative resource usage
- medium- 75% of CPU cores, balanced performance (default)
- high- 100% of CPU cores, maximum performance

Adaptive Features

- Dynamic concurrency adjustment based on system load
- Performance monitoring and logging when enabled
- Intelligent throttle delays to reduce CPU pressure
- Automatic scaling based on available CPU cores

CPU Throttling Examples

```
# Conservative resource usage for production environments
./certscanner -host example.com -cipherscan -cputhrottle low

# Balanced performance (recommended default)
./certscanner -host example.com -cipherscan -cputhrottle medium

# Maximum performance with monitoring
./certscanner -host example.com -cipherscan -cputhrottle high -logfile
performance.log

# Performance monitoring comparison
./certscanner -host example.com -cipherscan -cputhrottle none -logfile
baseline.log
```

Performance Impact

Testing shows that-cputhrottle low reduces CPU usage from 100% to approximately 68% while maintaining scan effectiveness. This makes it ideal for production environments where system stability is prioritized over scan speed.

Performance Considerations

Connection Count

Cipher scanning tests multiple combinations:

- 87+ cipher suite tests
- Multiple protocol versions (TLS 1.0-1.3)
- Multiple key exchange groups
- Typical: 200-400 connections per target

Resource Management

CPU throttling controls concurrency:

- Low: 2 concurrent connections
- Medium: 5 concurrent connections
- High: 10 concurrent connections
- None: 20 concurrent connections

Tip: Use-cputhrottle low for production scans to avoid overwhelming targets and minimize resource usage. Consider scanning during maintenance windows for comprehensive enterprise deployments.

Advanced System Use

Advanced Examples

Note: All commands below assume a valid license key is configured.

Enterprise Security Assessment

Large-scale cryptographic asset discovery across enterprise networks. Scan entire subnets, generate compliance reports (CBOM), and tag scans for tracking and audit trails. Ideal for quarterly security audits, risk assessments, and post-quantum readiness evaluations.

POWERSHELL:

```
# Complete enterprise scan with tracking
.\certscanner-windows-amd64.exe -host 10.0.0.0/16 -ports 443,22,993,995,636,8443
`
  -cipherscan -tags "enterprise,quarterly-audit" `
  -outputformat tychon -output Q4-crypto-audit.tychon.ndjson

# Generate compliance report
.\certscanner-windows-amd64.exe -host critical-servers.txt -cipherscan `
  -outputformat cbom -output compliance-cbom.json

# Stream to SIEM
.\certscanner-windows-amd64.exe -mode local -scanfilesystem -scanmemory `
  -outputformat flatndjson | your-siem-ingester
```

BASH:

```
# Complete enterprise scan with tracking
./certscanner-linux-x64 -host 10.0.0.0/16 -ports 443,22,993,995,636,8443 \
  -cipherscan -tags "enterprise,quarterly-audit" \
  -outputformat tychon -output Q4-crypto-audit.tychon.ndjson

# Generate compliance report
./certscanner-linux-x64 -host critical-servers.txt -cipherscan \
  -outputformat cbom -output compliance-cbom.json

# Stream to SIEM
./certscanner-linux-x64 -mode local -scanfilesystem -scanmemory \
  -outputformat flatndjson | your-siem-ingester
```

BASH:

```
# Complete enterprise scan with tracking
./certscanner-darwin-amd64 -host 10.0.0.0/16 -ports 443,22,993,995,636,8443 \
  -cipherscan -tags "enterprise,quarterly-audit" \
  -outputformat tychon -output Q4-crypto-audit.tychon.ndjson

# Generate compliance report
./certscanner-darwin-amd64 -host critical-servers.txt -cipherscan \
  -outputformat cbom -output compliance-cbom.json

# Stream to SIEM (no memory scanning on macOS)
./certscanner-darwin-amd64 -mode local -scanfilesystem \
  -outputformat flatndjson | your-siem-ingester
```

Incident Response & Forensics

Rapid cryptographic asset discovery during security incidents and breach investigations. Capture complete system state including active connections, in-memory cryptographic libraries, filesystem certificates, and email archives. Generate forensic reports for incident response teams and compliance documentation.

POWERSHELL:

```
# Complete system crypto inventory
.\certscanner-windows-amd64.exe -mode local -scanfilesystem -scanmemory -
scanconnected `
  -scanoutlookarchives -outputformat html -output system-crypto-report.html

# Quick compromise assessment
.\certscanner-windows-amd64.exe -mode local -scanconnected -quickscan `
  -outputformat json -output active-connections.json

# Memory forensics for crypto libraries
.\certscanner-windows-amd64.exe -mode local -scanmemory `
  -outputformat flatndjson -output crypto-libs-memory.ndjson
```

BASH:

```
# Complete system crypto inventory
./certscanner-linux-x64 -mode local -scanfilesystem -scanmemory -scanconnected \
  -scanoutlookarchives -outputformat html -output system-crypto-report.html

# Quick compromise assessment
./certscanner-linux-x64 -mode local -scanconnected -quickscan \
  -outputformat json -output active-connections.json

# Memory forensics for crypto libraries
./certscanner-linux-x64 -mode local -scanmemory \
  -outputformat flatndjson -output crypto-libs-memory.ndjson
```

BASH:

```
# Complete system crypto inventory (no memory scanning)
./certscanner-darwin-amd64 -mode local -scanfilesystem -scanconnected \
  -scanoutlookarchives -outputformat html -output system-crypto-report.html

# Quick compromise assessment
./certscanner-darwin-amd64 -mode local -scanconnected -quickscan \
  -outputformat json -output active-connections.json

# Note: Memory scanning not available on macOS
```

Continuous Monitoring

Automated scanning for continuous security monitoring and compliance tracking. Stream results to SIEM platforms (Elasticsearch, Kafka, Splunk), integrate with Windows EventLog, enable ARP-based network discovery, and upload reports to S3 for centralized storage. Perfect for scheduled jobs, real-time alerting, and DevSecOps pipelines.

POWERSHELL:

```
# Automated daily scans with tracking
.\certscanner-windows-amd64.exe -host production-hosts.txt -cipherscan `
```

```
  -tags "automated,daily-scan" -outputformat tychon `
  -posttoelastic -elasticnode "https://elastic.company.com:9200" `
  -elasticapikey "$env:ELASTIC_KEY"

# Real-time Kafka streaming
.\certscanner-windows-amd64.exe -host production-hosts.txt -cipherscan `
  -tags "automated,realtime-stream" -posttokafka `
  -kafkabrokers "kafka1:9092,kafka2:9092,kafka3:9092" `
  -kafkatopic "tychon-crypto-assets" -kafkausername "$env:KAFKA_USER" `
  -kafkapassword "$env:KAFKA_PASSWORD" -kafkasecurityprotocol "SASL_SSL"

# Windows EventLog integration
.\certscanner-windows-amd64.exe -mode local -outputformat eventlog

# ARP-based network discovery
.\certscanner-windows-amd64.exe -mode local -arpscan -quickscan -outputformat
flatndjson `
  -output network-discovery.ndjson

# Upload reports to S3 for centralized storage
.\certscanner-windows-amd64.exe -host example.com -cipherscan -outputformat
tychon `
  -upload-s3 -s3bucket "company-security-reports" `
  -s3prefix "certscanner/production" -s3region "us-west-2"
```

BASH:

```
# Automated daily scans with tracking
./certscanner-linux-x64 -host production-hosts.txt -cipherscan \
  -tags "automated,daily-scan" -outputformat tychon \
  -posttoelastic -elasticnode "https://elastic.company.com:9200" \
  -elasticapikey "$ELASTIC_KEY"

# Real-time Kafka streaming
./certscanner-linux-x64 -host production-hosts.txt -cipherscan \
  -tags "automated,realtime-stream" -posttokafka \
  -kafkabrokers "kafka1:9092,kafka2:9092,kafka3:9092" \
  -kafkatopic "tychon-crypto-assets" -kafkausername "$KAFKA_USER" \
  -kafkapassword "$KAFKA_PASSWORD" -kafkasecurityprotocol "SASL_SSL"

# ARP-based network discovery
./certscanner-linux-x64 -mode local -arpscan -quickscan -outputformat flatndjson
\
  -output network-discovery.ndjson

# Upload reports to S3 for centralized storage
./certscanner-linux-x64 -host example.com -cipherscan -outputformat tychon \
  -upload-s3 -s3bucket "company-security-reports" \
  -s3prefix "certscanner/production" -s3region "us-west-2"
```

BASH:

```
# Automated daily scans with tracking
./certscanner-darwin-amd64 -host production-hosts.txt -cipherscan \
  -tags "automated,daily-scan" -outputformat tychon \
```

```
  -posttoelastic -elasticnode "https://elastic.company.com:9200" \
  -elasticapikey "$ELASTIC_KEY"

# Real-time Kafka streaming
./certscanner-darwin-amd64 -host production-hosts.txt -cipherscan \
  -tags "automated,realtime-stream" -posttokafka \
  -kafkabrokers "kafka1:9092,kafka2:9092,kafka3:9092" \
  -kafkatopic "tychon-crypto-assets" -kafkausername "$KAFKA_USER" \
  -kafkapassword "$KAFKA_PASSWORD" -kafkasecurityprotocol "SASL_SSL"

# ARP-based network discovery
./certscanner-darwin-amd64 -mode local -arpscan -quickscan -outputformat
flatndjson \
  -output network-discovery.ndjson

# Upload reports to S3 for centralized storage
./certscanner-darwin-amd64 -host example.com -cipherscan -outputformat tychon \
  -upload-s3 -s3bucket "company-security-reports" \
  -s3prefix "certscanner/production" -s3region "us-west-2"
```

Filesystem Scanning Details

The scanner provides comprehensive filesystem scanning capabilities to discover cryptographic assets across your systems:

Certificates

PEM, DER, CRT, CER, PKCS#12, CSRs, and private keys

Keystores

Windows Cert Store, macOS Keychain, Java JKS, PKCS#12

Outlook Archives

Encrypted PST, OST, PAB files with S/MIME certificates

Certificate Scanning

Certificate scanning discovers and analyzes cryptographic certificate files stored on the filesystem.

Enable Filesystem Scanning

```
# Enable filesystem certificate scanning ./certscanner -mode local -
scanfilesystem -output certificates.json # Use with full scan (includes memory,
keystores, etc.) ./certscanner -mode local -fullscan -output full-scan.json
```

Supported File Formats

Format	Extensions	Description
PEM	.pem, .crt, .cert, .cer	Base64-encoded certificates and keys
DER	.der	Binary-encoded certificates

PKCS#12	.p12, .pfx	Password-protected certificate bundles
CSR	.csr	Certificate signing requests
Private Keys	.key, .pem	RSA, ECDSA, Ed25519, Ed448, DSA keys

Post-Quantum Cryptography Detection: The scanner automatically detects ML-DSA (Dilithium), ML-KEM (Kyber), SLH-DSA (SPHINCS+), and other PQC algorithms in certificates and private keys.

Keystore Detection

Keystore scanning automatically discovers OS-native keystores and file-based keystores (Java JKS, PKCS#12).

OS-Native Keystores

Windows

- Current User Certificate Store
- Local Machine Certificate Store
- All certificate contexts (MY, ROOT, CA, etc.)

macOS

- Login Keychain
- System Keychain
- User-specific keychains

Linux

- System CA bundle
- User certificate directories
- NSS databases (Firefox, Chrome)

File-Based Keystores

Type	Extensions	Common Uses
Java JKS	.jks, .keystore	Java applications, Tomcat, WebLogic
PKCS#12	.p12, .pfx	Cross-platform keystores, IIS, browsers
BKS	.bks	Android applications, BouncyCastle

Automatic Discovery: Keystore detection is automatically enabled with-scanfilesystemflag. No additional configuration required.

Outlook Archive Scanning

Detect encrypted Outlook archive files that may contain S/MIME certificates and encrypted emails.

Enable Outlook Scanning

```
# Scan for Outlook archives ./certscanner -mode local -scanoutlookarchives -
output outlook-archives.json # Combined with filesystem scan ./certscanner -mode
local -scanfilesystem -scanoutlookarchives -output combined-scan.json
```

Detected File Types

Extension	Type	Description
.pst	Personal Storage Table	Outlook data file containing emails, contacts, calendar
.ost	Offline Storage Table	Cached Exchange mailbox data
.pab	Personal Address Book	Outlook address book (legacy)

Encryption Detection Only: The scanner identifies encrypted Outlook files but does not decrypt or extract contents. This helps identify potential repositories of S/MIME certificates and encrypted communications.

Default Search Paths

When-filesystem-pathsis not specified, the scanner uses OS-specific default paths optimized for certificate discovery.

Windows Default Paths

User Directories

- C:\Users (all user profiles)

System Directories

- C:\Program Files
- C:\Program Files (x86)
- C:\ProgramData\Microsoft\Crypto
- C:\Windows\System32\certsrv
- C:\Windows\System32\CertLog

Application-Specific (if exist)

- C:\inetpub\certs (IIS)
- C:\Apache\conf\ssl, C:\Apache24\conf\ssl
- C:\nginx\conf\ssl
- C:\OpenSSL\certs, C:\OpenSSL-Win64\certs

macOS Default Paths

User Directories

- /Users (all user home directories)

System Directories

- /System/Library/Keychains
- /Library/Keychains
- /private/etc/certificates
- /etc/ssl/certs
- /usr/local/etc/openssl
- /opt/homebrew/etc/openssl (Apple Silicon)

Linux Default Paths

User Directories

- /home (all user home directories)
- /root (root user home)

System Certificate Directories

- /etc/ssl/certs
- /etc/pki/tls/certs

- /etc/pki/ca-trust
- /usr/share/ca-certificates
- /usr/local/share/ca-certificates
- /etc/letsencrypt/live
- /etc/letsencrypt/archive

Web Server Locations (if exist)

- /etc/apache2/ssl
- /etc/nginx/ssl
- /etc/httpd/ssl
- /var/lib/docker/volumes
- /opt/docker/certs

Custom Search Paths

Override default paths with custom directories for targeted scanning. Use the-filesystem-pathsflag with comma-separated paths.

Docker/Container Scanning

Tip: When specifying custom paths, the scanner ONLY scans those directories. Default paths are not included. To combine custom and default paths, list all directories explicitly.

Performance Considerations

Optimization Features

- Multi-threaded filesystem traversal
- Intelligent directory skipping (node_modules, .git, etc.)
- CPU throttling support (-cputhrottle)
- Concurrent file parsing

Performance Tips

- Use targeted paths for faster scanning
- Combine with-cputhrottle high for low-impact
- Run during maintenance windows for full scans
- Use-quick scan for initial discovery

Skipped Directories

The scanner automatically skips these directories to improve performance:

```
node_modules, .git, .svn, .hg, __pycache__, .cache, .tmp, temp, Trash
```

Output & Reporting

Filesystem scan results are included in the main JSON output under dedicated sections:


```
{ "filesystem_scan_results": [ { "source_file_path":
"/etc/ssl/certs/server.crt", "version": 3, "serial_number": "1234567890",
"signature_algorithm": "MLDSA65", "subject": { "common_name": "example.com",
"organization": ["Example Corp"] }, "validity": { "not_before": "2025-01-
01T00:00:00Z", "not_after": "2026-01-01T00:00:00Z" }, "subject_public_key_info":
{ "algorithm": "ML-DSA-65", "bit_size": 4096 }, "pqc_vulnerable": false,
"quantum_risk": "low", "file_details": { "path": "/etc/ssl/certs/server.crt",
"size": 2048, "owner": "root", "group": "root" } } ], "keystore_results": [...],
"outlook_archive_results": [...] }
```

Use different output formats for various use cases: JSON for processing, HTML for reporting, CBOM for compliance, FlatNDJSON for SIEM integration.

Frequently Asked Questions

This FAQ section provides comprehensive answers to common questions about Quantum Readiness Scanner, its compliance with federal mandates, and how it helps organizations prepare for the post-quantum cryptography era.

Compliance & Requirements

How does Quantum Readiness Scanner meet the OMB inventory requirements?

OMB Memorandum M-23-02 "Migrating to Post-Quantum Cryptography" directs federal agencies to 1) maintain annual inventories of quantum vulnerable cryptographic algorithms and 2) engage with NIST to coordinate efforts. In response to this memorandum, NIST issued the NIST 1800-38 series on Cryptographic Discovery and Interoperability Testing with practical guidance and specific reporting requirements.

NIST 1800-38B: Preparation for Considering the Implementation and Adoption of Quantum Safe Cryptography

- Quantum Readiness Scanner Module discovers keystore and cryptographic cipher information associated with all tools listed by NIST, plus others
- Coverage includes:Executables, Application binaries, Cryptographic libraries, Java archives, Non-executables, Keystores: PKCS#12, Java Keystores, OpenPGP Keys, X.509 Certificates, OpenSSH Keys, PKCS#1 and PKCS#8

What laws and policies are related to the mandate to establish a cryptographic inventory?

- 05/12/2021: Executive Order 14028Improving the Nation's Cybersecurity: Established a timeline for supporting TLS 1.3 and identifying product categories for PQC capable tools.
- 04/18/2022: H.R. 7535Quantum Computing Cybersecurity Preparedness Act: Requires federal agencies to prepare for the post-quantum era by evaluating and transitioning to quantum-resistant cryptography.
- 05/04/2022: NSM 10Promoting United States Leadership in Quantum Computing: Directs specific actions for agencies to begin to migrate vulnerable computer systems to quantum resistant information systems. Each agency is responsible for discovering, documenting, and maintaining a comprehensive inventory of devices and applications vulnerable to decryption by quantum computers.
- 11/18/2022: M-23-02Migrating to Post-Quantum Cryptography: Directed agencies to inventory cryptographic systems annually and laid out concrete steps toward being quantum ready.

Is Quantum Readiness only a government requirement or should commercial enterprises establish cryptographic inventory and risk assessment?

While government mandates and requirements target federal agencies, all enterprises with sensitive information are vulnerable. Organizations that are serious about protecting their data must start planning for post-quantum cryptography, and the first step is to understand what standards are currently in place. Quantum Readiness can help all organizations—federal and commercial—to easily generate a comprehensive inventory of cryptographic systems and identify weak ciphers and vulnerabilities.

- Government agencies
- Defense contractors
- Commercial enterprises
- Critical infrastructure providers
- Defense Industrial Base (DIB)

Does Quantum Readiness Scanner meet all the requirements established by OMB, NIST, CISA, and others?
Quantum Readiness Scanner is designed to comprehensively address federal cryptographic inventory and assessment requirements. We've done the homework to make it easy for federal buyers to justify the purchase of Quantum Readiness Scanner.
✓ OMB M-23-02 Compliance
✓ NIST 1800-38B Alignment
✓ NSM 10 Requirements
✓ CISA Cybersecurity Guidance

Technical Capabilities

How does Quantum Readiness Scanner work?
Quantum Readiness Scanner is a discovery and risk classification solution based on the fast-moving and daunting cryptographic space. We parse the complexity of algorithms and their implementations into discrete components, analyzing each step in the process, pinpointing the most urgent risks.

Why is it important to use an endpoint-centric cryptographic inventory toolset vs a network-only scanner?
Network-only monitoring solutions cannot effectively track enterprise-wide operations because they do not detect cryptographic operations that occur locally on endpoints. Without endpoint visibility, organizations only see encrypted data traversing the network, missing crucial information about the encryption process and implementation.
Critical Insight: On-device discovery captures both cryptography in use, and just as important: cryptography available for use. Bad ciphers must be found and removed from the system. Once they are used on the network, it is often too late to stop the negative impact.
Does Quantum Readiness Scanner provide both endpoint and network device cryptographic inventory?
Yes. the scanner's lightweight endpoint scripts deliver network-based information, but also continuous visibility regardless of location, ensuring that organizations maintain oversight of their cryptographic assets and operations.
What datasets does Quantum Readiness Scanner deliver?

Category	Dataset	Description
General	Host OS Info	Device Guard, Trusted Platform Module (TPM), UEFI Settings
Quantum Readiness	PQC Client TLS Protocols Data	Protocols Used to Originate a Client Session
PQC Discovered Cipher	Method to Transform Plaintext to Cipher Text	
PQC Listening Port Certificate	Attributes of Application Certificates	
PQC System Certificates	Certificates Used to Verify Secure Connections	
Network Data	Network PQC Discovered Ciphers	TLS Cipher Data in Network Packets
Non-TLS Protocol Discovery	Protocols such as SSH, S/MIME and VPNs	

Network PQC Listening Port Certificate	Certificate Data in Network Packets	

Integration & Deployment

How does the scanner provide visualization of the Quantum Readiness inventory?
Quantum Readiness Scanner includes pre-built dashboards for inventory and risk assessment for Splunk and Elastic integrations.
Can Quantum Readiness Scanner also perform remediation and migration to new quantum-resistant algorithms?
Quantum Readiness Scanner can be paired with the scanner Enterprise or other systems management tools like Microsoft Intune and BigFix to perform remediation and migration to quantum-resistant algorithms.

Does Quantum Readiness Scanner work with Splunk and Elastic?
Yes. Quantum Readiness Scanner integrates seamlessly with both Splunk and Elastic platforms. Pre-built dashboard content packs are available for both platforms. Note that you need to bring your own license (BYOL) for these platforms.
Does the scanner support other big data platforms, business intelligence tools, or SIEMS?
Yes. the scanner supports a wide range of enterprise platforms including:
SIEM Platforms

- IBM QRadar
- LogRhythm
- Splunk Enterprise

Data Platforms

- Elasticsearch
- Amazon OpenSearch
- Datadog
- Snowflake

Does the scanner work with any providers of quantum-resistant algorithms and solutions?
Yes. the scanner offers integrations with companies like SafeLogic and Qrypt to provide end-to-end cryptographic solutions.

Procurement & Support

Does the scanner offer pilots or proof of concept for Quantum Readiness?
Yes. the scanner offers pilot programs and proof of concept deployments for organizations interested in evaluating Quantum Readiness capabilities.
Is Quantum Readiness Scanner available for purchase on government contract vehicles?
Yes. Quantum Readiness Scanner is available through Carahsoft, our government contracting partner.

Appendix A: Output Formats

JSON Format

Overview

The JSON output format provides a comprehensive hierarchical structure containing all scan results. This is the default format and is ideal for integration with custom tools and comprehensive analysis.

Usage

BASH:

```
.\certscanner-windows-amd64.exe -host example.com -outputformat json -output
report.json
```

Complete JSON Schema

Top-Level Root Fields

Field	Type	Required	Description
scanning_system_info	Object	Yes	Information about the scanning system
scan_type	String	Yes	Scan mode: "local" or "remote"
target	String	Yes	Target specification provided by user
timestamp	String	Yes	ISO 8601 timestamp of scan start
tags	Array<String>	No	Custom tags applied to the scan
results	Array<NetworkResult>	No	Network scan results per host
filesystem_scan_results	Array<FilesystemCertificate>	No	Certificates found on filesystem
process_memory_scan_results	Array<MemoryLibrary>	No	Crypto libraries found in memory
outlook_archives	Array<OutlookArchive>	No	Outlook archive files discovered
ssh_scan_results	Array<SSHResult>	No	SSH host keys and protocol information
vpn_client_scan_results	Array<VPNClientInfo>	No	VPN client installations and PQC assessmentsNEW
ipsec_tunnel_scan_results	Array<IPSecTunnelInfo>	No	IPSec tunnel configurations and security analysisNEW

quantum_readiness	QuantumReadinessAssessment	No	System quantum readiness assessment with 100-point scoring (local mode only)NEW
keystore_scan_results	Array<KeystoreInfo>	No	Keystore discovery and certificate analysis (PKCS12, JKS, System Stores)NEW

scanning_system_info Object

Field	Type	Description	Example
hostname	String	Hostname of scanning system	"scanner-host"
os	String	Operating system	"windows", "linux", "darwin"
platform	String	Platform architecture	"windows/amd64", "linux/amd64"
platform_version	String	OS version	"10.0.19045", "20.04.6"
software_version	String	the scanner ACDI version	"1.0.42"
ip_addresses	Array<String>	Scanner system IP addresses	["192.168.1.100", "10.0.0.1"]
fips_mode_enabled	Boolean	FIPS 140-2 mode status	true, false
organization	String	Organization name (optional)	"Security Team"
username	String	User running the scan	"security.analyst"
bigfix_client_installed	Boolean	Indicates if BigFix client is installed	true, false
bigfix_client_id	String	BigFix client ID for asset correlation (optional)	"12345678"

NetworkResult Object (results array)

Field	Type	Description
scanned_host	String	Target hostname or IP address
domain	String	Resolved domain name
ip_address	String	Resolved IP address
scan_status	String	"success", "failed", "timeout"
error_message	String	Error details if scan failed
ports	Array<PortResult>	Results for each scanned port

PortResult Object

Field	Type	Description
port	Integer	Port number (443, 22, 993, etc.)
status	String	"open", "closed", "filtered"
protocol_detected	String	"TLS", "SSH", "SMTP-STARTTLS", etc.
tls_version	String	Negotiated TLS version

leaf_certificate	Certificate	Server's leaf certificate
certificate_chain	Array<Certificate>	Complete certificate chain
supported_cipher_suites	Array<CipherSuite>	All supported cipher suites
preferred_cipher_suite	CipherSuite	Server's preferred cipher suite
ssh_host_keys	Array<SSHHostKey>	SSH host keys (for SSH ports)
ssh_banner	String	SSH server banner
connection_time_ms	Integer	Connection establishment time
quantum_ready_kx	Boolean	Key exchange algorithm is quantum-safeNEW
quantum_ready_cipher	Boolean	Encryption cipher is quantum-safeNEW
quantum_ready_cert	Boolean	Certificate public key is quantum-safeNEW
quantum_ready	Boolean	Overall port quantum readiness (all components quantum-safe)NEW

Certificate Object (Complete Structure)

Field	Type	Description	Example
subject	SubjectObject	Certificate subject DN components	See SubjectObject
issuer	IssuerObject	Certificate issuer DN components	See IssuerObject
validity	ValidityObject	Certificate validity period	See ValidityObject
serial_number	String	Certificate serial number	"123456789012345678901234567890"
signature_algorithm	String	Signature algorithm	"SHA256-RSA", "SHA384-ECDSA"
subject_public_key_info	PublicKeyObject	Public key details	See PublicKeyObject
sha256_fingerprint	String	SHA-256 fingerprint	"ab:cd:ef:12:34:..."
sha1_fingerprint	String	SHA-1 fingerprint (legacy)	"12:34:56:78:..."
is_self_signed	Boolean	Whether certificate is self-signed	false
is_ca_certificate	Boolean	Whether this is a CA certificate	true
basic_constraints	Object	Basic constraints extension	{"ca": true, "path_len": 0}
key_usage	Array<String>	Key usage extensions	["digitalSignature", "keyEncipherment"]

extended_key_usage	Array<String>	Extended key usage	["serverAuth", "clientAuth"]
subject_alt_names	Array<String>	Subject alternative names	["*.example.com", "example.com"]
crl_distribution_points	Array<String>	CRL distribution URLs	["http://crl.example.com/ca.crl"]
authority_info_access	Object	OCSP and CA issuer URLs	{"ocsp": ["http://ocsp.example.com"]}
source_file_path	String	File path (filesystem scans)	"/etc/ssl/certs/ca.pem"

SubjectObject / IssuerObject

Field	Type	Description	Example
common_name	String	Common Name (CN)	"example.com"
organization	String	Organization (O)	"Example Corp"
organizational_unit	String	Organizational Unit (OU)	"IT Department"
country	String	Country (C)	"US"
state_or_province	String	State/Province (ST)	"California"
locality	String	City/Locality (L)	"San Francisco"
email_address	String	Email Address	"admin@example.com"
raw	String	Complete DN string	"CN=example.com,O=Example Corp,C=US"

CipherSuite Object

Field	Type	Description	Example
protocol	String	TLS protocol version	"TLSv1.3", "TLSv1.2"
cipher_suite	String	IANA cipher suite name	"TLS_AES_256_GCM_SHA384"
cipher_suite_hex	String	Hexadecimal identifier	"0x13,0x02"
key_length_bits	Integer	Symmetric key length in bits	256
negotiated_group	String	Key exchange group/curve	"X25519", "secp384r1"
is_preferred	Boolean	Server's preferred choice	true
source	String	OpenSSL cipher name	"ECDHE-RSA-AES256-GCM-SHA384"
intel	IntelObject	Security intelligence data	See IntelObject

IntelObject (Security Intelligence)

Field	Type	Description	Values
security_level	String	Overall security assessment	"high", "medium", "low", "insecure"

recommendation	String	Security recommendation	"recommended", "acceptable", "legacy", "avoid"
pqc_ready	Boolean	Post-quantum cryptography ready	false (most current ciphers)
vulnerabilities	Array<String>	Known security vulnerabilities	["BEAST", "CRIME", "POODLE"]
nist_security_category	String	NIST classification	"Recommended", "Legacy-Use", "Deprecated"
friendly_name	String	Human-readable cipher name	"AES-256 with GCM and SHA-384"
description	String	Detailed cipher description	"Advanced Encryption Standard..."

MemoryLibrary Object

Field	Type	Description
process_id	Integer	Process ID
process_name	String	Process executable name
process_path	String	Full path to process executable
command_line	String	Complete command line
username	String	User running the process
library_name	String	Crypto library name
library_version	String	Library version
library_path	String	Library file path
crypto_type	String	"openssl", "bcrypt", "java_crypto"
product_name	String	Product name from file metadata
company_name	String	Company from file metadata
file_description	String	File description
sha256_hash	String	SHA-256 hash of library file

SSHHostKey Object

Field	Type	Description	Example
algorithm	String	Host key algorithm	"ssh-rsa", "ecdsa-sha2-nistp256"
key_size_bits	Integer	Key size in bits	2048, 256
fingerprint_md5	String	MD5 fingerprint (legacy)	"12:34:56:78:..."
fingerprint_sha256	String	SHA-256 fingerprint	"SHA256:abcd..."
curve_name	String	Elliptic curve name (ECDSA keys)	"nistp256", "nistp384"
public_key_data	String	Base64-encoded public key	"AAAAB3NzaC1yc2E..."
is_weak	Boolean	Whether key is cryptographically weak	false

OutlookArchive Object

Field	Type	Description	Example

file_path	String	Full path to PST/OST file	"C:\\Users\\user\\archive.pst"
file_size_bytes	Integer	File size in bytes	1048576000
is_encrypted	Boolean	Whether archive is encrypted	true
encryption_type	String	Encryption method	"Compressible", "High"
created_date	String	Archive creation date	"2024-01-15T10:30:00Z"
last_modified	String	Last modification date	"2024-12-01T15:45:00Z"
owner	String	File owner	"DOMAIN\\username"

VPNClientInfo ObjectNEW

Field	Type	Description	Example
source_id	String	Unique identifier for tracking	"d87c1d880886fd83..."
client_name	String	VPN client application name	"Palo Alto GlobalProtect"
vendor	String	Software vendor/manufacturer	"Palo Alto Networks"
version	String	Client software version	"6.3.2-525"
install_path	String	Installation directory path	"/Applications/GlobalProtect.app"
config_path	String	Configuration file location	"~/Library/Application Support/..."
executable_path	String	Main executable file path	"/Applications/.../GlobalProtect"
service_name	String	System service identifier	"com.paloaltonetworks.globalprotect"
process_id	Integer	Current process ID (if running)	4473
status	String	Current operational status	"active", "inactive", "unknown"
detection_method	String	How client was discovered	"filesystem", "registry", "process"
detection_confidence	String	Detection accuracy level	"high", "medium", "low"
pqc_assessment	PQCAssessment	Post-quantum cryptography analysis	See PQCAssessment Object
security_assessment	VPNSecurityAssessment	Comprehensive security analysis and scoring	See VPNSecurityAssessment Object
configuration_security	VPNConfigSecurity	VPN configuration security settings	See VPNConfigSecurity Object
active	Boolean	Whether client is currently active	true

last_seen	String	ISO 8601 timestamp of last detection	"2025-09-12T12:54:43.593113-04:00"
first_detected	String	ISO 8601 timestamp of first detection	"2025-09-12T12:54:43.583862-04:00"

IPSecTunnelInfo ObjectNEW

Field	Type	Description	Example
source_id	String	Unique identifier for tracking	"90e2352de5c7c9d856327..."
tunnel_name	String	IPSec tunnel or connection name	"macOS Built-in IPSec"
implementation	String	IPSec implementation type	"strongswan", "libreswan", "macOS"
config_path	String	Configuration file location	"/etc/ipsec.conf"
status	String	Current tunnel status	"active", "inactive", "unknown"
detection_method	String	How tunnel was discovered	"config_file", "process", "kernel"
detection_confidence	String	Detection accuracy level	"high", "medium", "low"
tunnel_details	IPSecTunnelDetails	Detailed tunnel configuration	See IPSecTunnelDetails Object
pqc_assessment	PQCAssessment	Post-quantum cryptography analysis	See PQCAssessment Object
security_assessment	IPSecSecurityAssessment	Comprehensive security analysis and scoring	See IPSecSecurityAssessment Object
active	Boolean	Whether tunnel is currently active	false
last_seen	String	ISO 8601 timestamp of last detection	"2025-09-12T12:49:37.307164-04:00"
first_detected	String	ISO 8601 timestamp of first detection	"2025-09-12T12:49:37.296218-04:00"

PQCAssessment ObjectNEW

Field	Type	Description	Example
is_pqc_ready	Boolean	Whether implementation supports PQC algorithms	true
quantum_resistance	String	Level of quantum resistance	"high", "medium", "low", "none"
pqc_migration_status	String	Migration readiness status	"ready", "partial", "not_ready"

supported_pqc_algorithms	Array<String>	List of supported PQC algorithms	["ML-KEM-512", "ML-DSA-44"]
pqc_version_available	String	Version with PQC support (if any)	"6.4.0"
last_assessed	String	ISO 8601 timestamp of assessment	"2025-09-12T12:54:43.476222-04:00"

VPNSecurityAssessment ObjectNEW

Field	Type	Description	Example
security_score	Integer	Overall security score (0-100)	87
risk_level	String	Security risk assessment level	"low", "medium", "high", "critical"
pqc_vulnerable	Boolean	Vulnerable to quantum attacks	false
pqc_support	Boolean	Supports post-quantum cryptography	true
vulnerable	Boolean	Has known security vulnerabilities	false
weak_crypto	Boolean	Uses weak cryptographic algorithms	false
known_vulnerabilities	Array<String>	List of CVE identifiers	["CVE-2023-1234", "CVE-2023-5678"]
last_assessed	String	ISO 8601 timestamp of assessment	"2025-09-19T10:30:00Z"

VPNConfigSecurity ObjectNEW

Field	Type	Description	Example
authentication_method	String	Authentication method used	"certificate", "psk", "username_password"
dns_leak_protection	Boolean	DNS leak protection enabled	true
kill_switch	Boolean	Kill switch/network lock enabled	true
split_tunneling	Boolean	Split tunneling enabled (security risk)	false
ipv6_leak_protection	Boolean	IPv6 leak protection enabled	true
auto_reconnect	Boolean	Auto-reconnect enabled	true
logging_enabled	Boolean	VPN logging enabled	false
config_encrypted	Boolean	Configuration file encrypted	true

IPSecSecurityAssessment ObjectNEW

Field	Type	Description	Example
security_score	Integer	Overall security score (0-100)	92

risk_level	String	Security risk assessment level	"low", "medium", "high", "critical"
pqc_vulnerable	Boolean	Vulnerable to quantum attacks	false
pqc_support	Boolean	Supports post-quantum cryptography	true
vulnerable	Boolean	Has known security vulnerabilities	false
weak_crypto	Boolean	Uses weak cryptographic algorithms	false
known_vulnerabilities	Array<String>	List of CVE identifiers	[]
last_assessed	String	ISO 8601 timestamp of assessment	"2025-09-19T10:30:00Z"

KeystoreInfo ObjectNEW

Field	Type	Description	Example
source_id	String	Unique identifier for tracking	"ks_12345abcd..."
path	String	Full path to keystore file or identifier	"/home/user/keystore.p12"
type	String	Keystore format type	"PKCS12", "JKS", "Windows", "macOS"
accessible	Boolean	Whether keystore is accessible	true
requires_auth	Boolean	Whether authentication is required	false
cert_count	Integer	Number of certificates found	15
owner	String	File owner (if available)	"domain\\username"
permissions	String	File permissions	"rw-r--r--"
size	Integer	File size in bytes	2048576
last_modified	String	ISO 8601 timestamp of last modification	"2024-12-01T10:30:00Z"
error_message	String	Error details if access failed	"Password required"
certificates	Array<KeystoreCertificate>	Certificates contained in keystore	See KeystoreCertificate Object
active	Boolean	Whether keystore is currently active	true
last_seen	String	ISO 8601 timestamp of last detection	"2025-09-17T14:30:00Z"
first_detected	String	ISO 8601 timestamp of first detection	"2025-09-17T14:30:00Z"

KeystoreCertificate ObjectNEW

Field	Type	Description	Example
alias	String	Certificate alias in keystore	"my-server-cert"
subject	String	Certificate subject DN	"CN=example.com,O=Example Corp"
issuer	String	Certificate issuer DN	"CN=CA Root,O=Trust CA"
serial_number	String	Certificate serial number	"123456789"
thumbprint	String	SHA-1 thumbprint	"12:34:56:78:ab:cd"
not_before	String	Certificate validity start date	"2024-01-01T00:00:00Z"
not_after	String	Certificate validity end date	"2025-01-01T00:00:00Z"
key_algorithm	String	Public key algorithm	"RSA", "ECDSA", "Ed25519"
key_size	Integer	Key size in bits	2048
signature_algo	String	Signature algorithm	"SHA256-RSA"
version	Integer	X.509 version	3
is_ca	Boolean	Whether certificate is a CA	false
is_self_signed	Boolean	Whether certificate is self-signed	false
has_private_key	Boolean	Whether private key is available	true
key_usage	Array<String>	Key usage extensions	["digitalSignature", "keyEncipherment"]
ext_key_usage	Array<String>	Extended key usage	["serverAuth", "clientAuth"]
chain_length	Integer	Certificate chain length	3
chain_complete	Boolean	Whether certificate chain is complete	true
vulnerable	Boolean	Whether certificate has vulnerabilities	false
risk_level	String	Risk assessment level	"low", "medium", "high", "critical"
risk_reason	String	Reason for risk assessment	"Weak key size"
cve_list	Array<String>	Associated CVE identifiers	["CVE-2024-1234"]
fixed_in_version	String	Version where vulnerability is fixed	"1.2.3"
pqc_vulnerable	Boolean	Whether vulnerable to quantum attacks	true
pqc_reason	String	Reason for PQC vulnerability	"RSA algorithm vulnerable to quantum"
active	Boolean	Whether certificate is currently active	true
last_seen	String	ISO 8601 timestamp of last detection	"2025-09-17T14:30:00Z"

QuantumReadinessAssessment ObjectNEW

Field	Type	Description	Example
assessment_id	String	Unique identifier for this assessment	"qr_20250915_101539_abc123"
timestamp	String	ISO 8601 timestamp of assessment	"2025-09-15T10:15:39.123456-07:00"
assessment_type	String	Type of assessment performed	"comprehensive"
system_type	String	Classification of system type	"workstation", "server"
system_role	String	Primary role of the system	"workstation", "server", "unknown"
criticality_level	String	System criticality classification	"critical", "important", "standard"
fips_mode_enabled	Boolean	FIPS 140-2 mode status at time of assessment	true, false
hardware_score	HardwareAssessment	Hardware readiness scoring (40 points max)	See HardwareAssessment Object
operating_system_score	OSAssessment	OS readiness scoring (30 points max)	See OSAssessment Object
crypto_library_score	CryptoAssessment	Crypto library scoring (25 points max)	See CryptoAssessment Object
network_score	NetworkAssessment	Network readiness scoring (5 points max)	See NetworkAssessment Object
overall_score	Integer	Total quantum readiness score out of 100	64
readiness_status	String	Overall readiness classification	"Ready", "Partially Ready", "Update Required", "Not Ready"
status_color	String	Color code for status visualization	"green", "yellow", "orange", "red"
ready_timeline	String	Estimated timeline to quantum readiness	"2-6 months"
recommendations	Array<String>	Actionable recommendations	["Upgrade to macOS 15.0+", "Update OpenSSL to 3.4.0+"]
critical_issues	Array<Issue>	Critical blocking issues	[]

upgrade_pathway	Array<UpgradeStep>	Step-by-step upgrade plan	See UpgradeStep Object
compliance_status	ComplianceAssessment	Compliance framework assessment	See ComplianceAssessment Object
detailed_report	String	Comprehensive assessment summary	"System shows moderate quantum readiness..."

Sample Output

Remote Scan Example

JSON:


```
{
  "scanning_system_info": {
    "hostname": "scanner-host",
    "os": "darwin",
    "platform": "darwin",
    "platform_version": "15.5",
    "software_version": "1.0.42",
    "ip_addresses": ["192.168.1.100"],
    "organization": "Security Team"
  },
  "scan_type": "remote",
  "target": "example.com:443",
  "timestamp": "2025-09-02T09:00:17-04:00",
  "tags": ["production", "weekly-scan"],
  "results": [
    {
      "scanned_host": "example.com",
      "domain": "example.com",
      "ports": [
        {
          "port": 443,
          "status": "open",
          "protocol_detected": "TLS",
          "leaf_certificate": {
            "subject": {
              "common_name": "example.com",
              "organization": "Example Corp",
              "country": "US",
              "raw": "CN=example.com,O=Example Corp,C=US"
            },
            "issuer": {
              "common_name": "DigiCert TLS RSA SHA256 2020 CA1",
              "organization": "DigiCert Inc",
              "country": "US",
              "raw": "CN=DigiCert TLS RSA SHA256 2020 CA1,O=DigiCert Inc,C=US"
            },
            "validity": {
              "not_before": "2024-03-01T00:00:00Z",
              "not_after": "2025-03-01T23:59:59Z"
```

```
            },
            "serial_number": "1234567890123456789012345678 90",
            "signature_algorithm": "SHA256-RSA",
            "subject_public_key_info": {
              "algorithm": "RSA",
              "bit_size": 2048
            },
            "sha256_fingerprint": "ab:cd:ef:12:34:56:78:90:...",
            "is_self_signed": false
          },
          "supported_cipher_suites": [
            {
              "protocol": "TLSv1.3",
              "cipher_suite": "TLS_AES_256_GCM_SHA384",
              "key_length_bits": 256,
              "negotiated_group": "X25519",
              "is_preferred": true,
              "source": "ECDHE-RSA-AES256-GCM-SHA384",
              "intel": {
                "security_level": "high",
                "recommendation": "recommended",
                "pqc_ready": false
              }
            }
          ]
        }
      ]
    }
  ],
  "filesystem_scan_results": [
    {
      "source_file_path": "/etc/ssl/certs/example.pem",
      "subject": {
        "common_name": "Internal CA",
        "organization": "Example Corp"
      },
      "validity": {
        "not_before": "2023-01-01T00:00:00Z",
        "not_after": "2033-01-01T00:00:00Z"
      },
      "serial_number": "98765432109876543210987654321 0",
      "is_self_signed": true
    }
  ],
  "vpn_client_scan_results": [
    {
      "source_id": "d87c1d880886fd83db018456d742cb83efa0758e",
      "client_name": "Palo Alto GlobalProtect",
      "vendor": "Palo Alto Networks",
      "version": "6.3.2-525",
      "install_path": "/Applications/GlobalProtect.app",
      "config_path": "~/Library/Application
Support/com.paloaltonetworks.globalprotect",
```

```
      "executable_path":
"/Applications/GlobalProtect.app/Contents/MacOS/GlobalProtect",
      "service_name": "com.paloaltonetworks.globalprotect",
      "process_id": 4473,
      "status": "active",
      "detection_method": "filesystem",
      "detection_confidence": "high",
      "pqc_assessment": {
        "is_pqc_ready": true,
        "quantum_resistance": "medium",
        "pqc_migration_status": "partial",
        "supported_pqc_algorithms": ["ML-KEM-512"],
        "pqc_version_available": "6.4.0",
        "last_assessed": "2025-09-12T12:54:43.476222-04:00"
      },
      "active": true,
      "last_seen": "2025-09-12T12:54:43.593113-04:00",
      "first_detected": "2025-09-12T12:54:43.583862-04:00"
    }
  ],
  "ipsec_tunnel_scan_results": [
    {
      "source_id": "90e2352de5c7c9d856327dcfef4ffbd89c2634a1",
      "tunnel_name": "strongSwan Site-to-Site",
      "implementation": "strongswan",
      "config_path": "/etc/ipsec.conf",
      "status": "inactive",
      "detection_method": "config_file",
      "detection_confidence": "high",
      "tunnel_details": {
        "local_subnet": "192.168.1.0/24",
        "remote_subnet": "10.0.0.0/24",
        "gateway": "203.0.113.1",
        "encryption_algorithms": ["aes256"],
        "integrity_algorithms": ["sha256"],
        "key_exchange_groups": ["modp2048"]
      },
      "pqc_assessment": {
        "is_pqc_ready": false,
        "quantum_resistance": "low",
        "pqc_migration_status": "not_ready",
        "supported_pqc_algorithms": [],
        "pqc_version_available": "5.9.12",
        "last_assessed": "2025-09-12T12:49:37.296203-04:00"
      },
      "active": false,
      "last_seen": "2025-09-12T12:49:37.307164-04:00",
      "first_detected": "2025-09-12T12:49:37.296218-04:00"
    }
  ],
  "keystore_scan_results": [
    {
      "source_id": "ks_a1b2c3d4e5f67890123456789012345678 90abcd",
```

```
      "path": "/Users/admin/Documents/certificates/server.p12",
      "type": "PKCS12",
      "accessible": true,
      "requires_auth": false,
      "cert_count": 3,
      "owner": "admin",
      "permissions": "rw-r--r--",
      "size": 4096,
      "last_modified": "2024-12-01T10:30:00Z",
      "certificates": [
        {
          "alias": "server-cert",
          "subject": "CN=api.example.com,O=Example Corp,C=US",
          "issuer": "CN=Example Internal CA,O=Example Corp,C=US",
          "serial_number": "0x1a2b3c4d5e6f7890",
          "thumbprint":
"a1:b2:c3:d4:e5:f6:78:90:12:34:56:78:90:ab:cd:ef:12:34:56:78",
          "not_before": "2024-01-01T00:00:00Z",
          "not_after": "2025-12-31T23:59:59Z",
          "key_algorithm": "RSA",
          "key_size": 2048,
          "signature_algo": "SHA256-RSA",
          "version": 3,
          "is_ca": false,
          "is_self_signed": false,
          "has_private_key": true,
          "key_usage": ["digitalSignature", "keyEncipherment"],
          "ext_key_usage": ["serverAuth"],
          "chain_length": 2,
          "chain_complete": true,
          "vulnerable": false,
          "risk_level": "medium",
          "risk_reason": "RSA-2048 approaching deprecation timeline",
          "pqc_vulnerable": true,
          "pqc_reason": "RSA algorithm vulnerable to quantum cryptanalysis",
          "active": true,
          "last_seen": "2025-09-17T14:30:00Z"
        }
      ],
      "active": true,
      "last_seen": "2025-09-17T14:30:00Z",
      "first_detected": "2025-09-17T14:30:00Z"
    }
  ]
}
```

CBOM Format

Overview

The CBOM (Cryptographic Bill of Materials) format follows the CycloneDX specification and IBM CBOM 1.0 standards. It provides a comprehensive inventory of all cryptographic assets discovered during scanning.

Standards Compliance

- CycloneDX BOM Format
- IBM CBOM 1.0 Schema
- JSON Schema Draft-07
- UUID Serial Numbers

Usage

BASH:

```
.\certscanner-windows-amd64.exe -host example.com `
  -outputformat cbom `
  -output compliance.cbom.json
```

CBOM Schema Structure

Required Top-Level Fields

Field	Type	Description	Example
bomFormat	String	Must be "CycloneDX"	"CycloneDX"
specVersion	String	CBOM specification version	"1.4-cbom-1.0"
serialNumber	String	Unique UUID for this BOM	"urn:uuid:..."
version	Integer	BOM version number	1
metadata	Object	Scan metadata and tool info	See below
components	Array	Cryptographic components	See below
services	Array	Network services discovered	Optional

Component Types

Component Type	Asset Type	Description
cryptographic-asset	algorithm	TLS cipher suites and crypto algorithms
cryptographic-asset	certificate	X.509 certificates (network & filesystem)
cryptographic-asset	relatedCryptoMaterial	SSH host keys and crypto keys
library	-	Cryptographic libraries in memory

file	-	Outlook archives and crypto files
application	-	VPN client applications with PQC assessmentsNEW
cryptographic-asset	protocol	IPSec tunnel configurations and protocolsNEW
cryptographic-asset	certificate	Keystore certificates (PKCS12, JKS, System Stores)NEW
file	-	Keystore files and certificate containersNEW

Crypto Properties Schema

Property Type	Key Fields	Use Case
algorithmProperties	primitive, parameterSetIdentifier, curve, executionEnvironment, mode, padding, classicalSecurityLevel, nistQuantumSecurityLevel	Cipher suites, crypto algorithms
certificateProperties	subjectName, issuerName, notValidBefore, notValidAfter, certificateFormat, certificateExtension	X.509 certificates
protocolProperties	type, version, cipherSuites, supportedDHGroups, supportedEncryptions, supportedHashes, supportedAuthentications	TLS/SSH/IPSec protocols
relatedCryptoMaterialProperties	type, algorithm, size, format, state, creationDate, activationDate, expirationDate	SSH keys, crypto material

Sample CBOM Output

JSON:


```
{
  "bomFormat": "CycloneDX",
  "specVersion": "1.4-cbom-1.0",
  "serialNumber": "urn:uuid:68b6ea62-252b-252b-252b-78adf023bc59",
  "version": 1,
  "metadata": {
    "timestamp": "2025-09-02T09:00:17-04:00",
    "tools": [
      {
        "vendor": "Tychon LLC",
```

```
        "name": "TYCHON Quantum Readiness",
        "version": "1.0.82"
      }
    ],
    "properties": [
      {
        "name": "scan:type",
        "value": "remote"
      },
      {
        "name": "scan:target",
        "value": "example.com:443"
      },
      {
        "name": "observer:hostname",
        "value": "scanner-host"
      }
    ]
  },
  "components": [
    {
      "type": "cryptographic-asset",
      "bom-ref": "cipher:example.com:443:TLS_AES_256_GCM_SHA384",
      "name": "TLS_AES_256_GCM_SHA384",
      "version": "TLSv1.3",
      "description": "TLS cipher suite TLS_AES_256_GCM_SHA384 on
example.com:443",
      "cryptoProperties": {
        "assetType": "algorithm",
        "algorithmProperties": {
          "primitive": "cipher-suite",
          "parameterSetIdentifier": "TLS_AES_256_GCM_SHA384",
          "executionEnvironment": "tls-connection"
        }
      },
      "properties": [
        {
          "name": "cipher:openssl-name",
          "value": "ECDHE-RSA-AES256-GCM-SHA384"
        },
        {
          "name": "cipher:key-length",
          "value": "256"
        },
        {
          "name": "cipher:is-preferred",
          "value": "true"
        },
        {
          "name": "cipher:intel:security_level",
          "value": "high"
        }
      ]
```

```
    },
    {
      "type": "cryptographic-asset",
      "bom-ref": "cert:example.com:443:123456789012345678901234567890",
      "name": "example.com",
      "description": "X.509 certificate (network-certificate) for
example.com:443",
      "hashes": [
        {
          "alg": "SHA-256",
          "content": "ab:cd:ef:12:34:56:78:90:..."
        }
      ],
      "cryptoProperties": {
        "assetType": "certificate",
        "certificateProperties": {
          "subjectName": "CN=example.com,O=Example Corp,C=US",
          "issuerName": "CN=DigiCert TLS RSA SHA256 2020 CA1,O=DigiCert
Inc,C=US",
          "notValidBefore": "2024-03-01T00:00:00Z",
          "notValidAfter": "2025-03-01T23:59:59Z",
          "certificateFormat": "X.509",
          "certificateExtension": "DER/PEM"
        }
      },
      "properties": [
        {
          "name": "cert:serial-number",
          "value": "123456789012345678901234567890"
        },
        {
          "name": "cert:signature-algorithm",
          "value": "SHA256-RSA"
        },
        {
          "name": "cert:public-key-algorithm",
          "value": "RSA"
        },
        {
          "name": "cert:public-key-size",
          "value": "2048"
        }
      ]
    }
  ],
  "services": [
    {
      "bom-ref": "service:example.com:443",
      "name": "example.com:443",
      "description": "Network service on example.com port 443",
      "endpoints": ["example.com:443"],
      "properties": [
        {
```

```
        "name": "port",
        "value": "443"
      },
      {
        "name": "status",
        "value": "open"
      },
      {
        "name": "protocol",
        "value": "TLS"
      }
    ]
  }
],
"additionalComponents": [
  {
    "type": "application",
    "bom-ref": "vpn-client:Cloudflare WARP:2024.6.415",
    "name": "Cloudflare WARP",
    "version": "2024.6.415",
    "description": "VPN client application: Cloudflare WARP",
    "properties": [
      {
        "name": "application:vendor",
        "value": "Cloudflare Inc."
      },
      {
        "name": "application:active",
        "value": "true"
      },
      {
        "name": "pqc:is-ready",
        "value": "true"
      },
      {
        "name": "pqc:quantum-resistance",
        "value": "high"
      },
      {
        "name": "pqc:supported-algorithms",
        "value": "Kyber768,X25519Kyber768Draft00"
      }
    ]
  },
  {
    "type": "cryptographic-asset",
    "bom-ref": "ipsec-tunnel:VPN-HQ",
    "name": "VPN-HQ",
    "version": "IKEv2",
    "description": "IPSec tunnel configuration: VPN-HQ",
    "cryptoProperties": {
      "assetType": "protocol",
      "protocolProperties": {
```

```
            "type": "IPSec",
            "version": "IKEv2",
            "supportedEncryptions": ["AES-256-GCM"],
            "supportedHashes": ["SHA256"],
            "supportedDHGroups": ["group19"]
          }
        },
        "properties": [
          {
            "name": "ipsec:implementation",
            "value": "strongSwan"
          },
          {
            "name": "ipsec:status",
            "value": "established"
          },
          {
            "name": "pqc:is-ready",
            "value": "false"
          }
        ]
      },
      {
        "type": "file",
        "bom-ref": "keystore-file:C:\\Users\\Admin\\certificates.pfx",
        "name": "Keystore (PKCS12)",
        "description": "Keystore file: C:\\Users\\Admin\\certificates.pfx",
        "properties": [
          {
            "name": "keystore:type",
            "value": "PKCS12"
          },
          {
            "name": "keystore:cert-count",
            "value": "3"
          },
          {
            "name": "keystore:accessible",
            "value": "true"
          }
        ]
      },
      {
        "type": "cryptographic-asset",
        "bom-ref": "keystore-cert:C:\\Users\\Admin\\certificates.pfx:1234567890",
        "name": "CN=MyApp Code Signing",
        "description": "Certificate from keystore:
C:\\Users\\Admin\\certificates.pfx",
        "cryptoProperties": {
          "assetType": "certificate",
          "certificateProperties": {
            "subjectName": "CN=MyApp Code Signing,O=MyCompany",
            "issuerName": "CN=MyCompany Root CA",
```

```
            "notValidBefore": "2024-01-01T00:00:00Z",
            "notValidAfter": "2027-01-01T00:00:00Z",
            "certificateFormat": "X.509",
            "certificateExtension": "DER/PEM"
          }
        },
        "properties": [
          {
            "name": "cert:keystore-type",
            "value": "PKCS12"
          },
          {
            "name": "cert:has-private-key",
            "value": "true"
          },
          {
            "name": "cert:key-usage",
            "value": "digitalSignature,keyEncipherment"
          }
        ]
      }
    ]
  }
```

Note: The "additionalComponents" section above shows example VPN client, IPSec tunnel, keystore file, and keystore certificate components. These would normally be in the main "components" array.

Cryptographic Asset Types

Cipher Suite Components

Each negotiated TLS cipher suite becomes a cryptographic-asset component

Certificate Components

X.509 certificates from network connections and filesystem

SSH Host Key Components

SSH host keys discovered during network scanning

Cryptographic Library Components

Crypto libraries discovered in process memory

VPN Client ComponentsNEW

VPN client applications with PQC readiness assessments

IPSec Tunnel ComponentsNEW

IPSec tunnel configurations and cryptographic protocols

Keystore Certificate ComponentsNEW

Certificates from PKCS12, JKS, Windows Certificate Store, and macOS Keychain

Keystore File ComponentsNEW

Keystore files and certificate containers discovered on filesystem

Use Cases & Integration

Compliance & Governance

- Regulatory Compliance: NIST, FIPS, Common Criteria
- Audit Trails: Complete cryptographic asset inventory
- Risk Assessment: Identify weak or deprecated crypto
- Supply Chain Security: Track crypto dependencies

Tool Integration

- Vulnerability Scanners: Import crypto asset data
- Asset Management: Track crypto inventory changes
- Policy Engines: Validate crypto policy compliance
- Reporting Tools: Generate compliance reports

Example Workflows

BASH:

```
# Generate quarterly compliance report
.\certscanner-windows-amd64.exe -host production-systems.txt -cipherscan `
  -tags "Q4-2025,compliance-audit" `
  -outputformat cbom -output Q4-crypto-inventory.cbom.json

# Continuous compliance monitoring
.\certscanner-windows-amd64.exe -mode local -scanfilesystem -scanmemory `
  -outputformat cbom -output daily-crypto-inventory.cbom.json

# Integration with vulnerability management
.\certscanner-windows-amd64.exe -host critical-infrastructure.txt `
  -outputformat cbom | vulnerability-analyzer --cbom-input
```

Metadata Properties Reference

Quantum Readiness Scanner includes comprehensive metadata in the CBOM metadata.properties array:

Scan Metadata Properties

Property Name	Description	Example Value
scan:type	Type of scan performed	"remote", "local"
scan:target	Target hostname or IP	"example.com:443"
scan:timestamp	When scan was performed	"2025-10-01T14:30:00Z"
scan:tags	User-defined tags	"production,compliance"

Observer System Properties

Property Name	Description	Example Value
observer:hostname	Scanner hostname	"scanner-01.example.com"
observer:os	Operating system	"windows"
observer:platform	Platform architecture	"amd64"
observer:version	OS version	"10.0.19045"
observer:fips_mode_enabled	FIPS mode status	"true", "false"
observer:organization	Organization name	"ACME Corporation"

Quantum Readiness Properties

Property Name	Description	Example Value
quantum:assessment_id	Unique assessment ID	"qa_abc123..."
quantum:fips_mode_enabled	System FIPS mode	"true", "false"
quantum:overall_score	Total readiness score	"75"
quantum:max_score	Maximum possible score	"100"
quantum:readiness_status	Overall readiness level	"ready", "partial", "not_ready"
quantum:ready_timeline	Expected timeline	"2025", "2026-2027", "2028+"

Schema Validation

Validation Against IBM CBOM Schema

Quantum Readiness Scanner's CBOM output is designed to validate against the official IBM CBOM 1.0 schema:

BASH:

```
# Validate with JSON schema tools
jsonschema -i compliance.cbom.json \
  https://github.com/IBM/CBOM/blob/main/bom-1.4-cbom-1.0.schema.json
```

Flat NDJSON Format

Overview

The Flat NDJSON format outputs one JSON record per line, with each line representing a single cipher suite, certificate, crypto library, or other cryptographic asset. All nested structures are flattened using dot notation.

Best For

- • ELK Stack ingestion
- • Streaming log analysis
- • Time-series databases
- • Log aggregation systems

Usage

POWERSHELL:

```
`.\certscanner-windows-amd64.exe -host example.com `
  -outputformat flatndjson `
  -output stream.ndjson
```

BASH:

```
./certscanner-linux-x64 -host example.com \
  -outputformat flatndjson \
  -output stream.ndjson
```

BASH:

```
# Intel Macs
./certscanner-darwin-amd64 -host example.com \
  -outputformat flatndjson \
  -output stream.ndjson

# Apple Silicon Macs
./certscanner-darwin-arm64 -host example.com \
  -outputformat flatndjson \
  -output stream.ndjson
```

Complete Flat NDJSON Schema

Schema Overview

Each line in the NDJSON output represents a single cryptographic asset event. All nested JSON structures are flattened using dot notation (e.g.,certificate.subject.common_name). Different event types share common base fields but include type-specific fields.

Base Event Fields (All Events)

Field	Type	Required	Description	Example
@timestamp	String	Yes	Event timestamp (ISO 8601)	"2025-09-02T13:45:30.123Z"
event.action	String	Yes	Event type identifier	"cipher_suite_discovered"

event.category	String	Yes	ECS event category	"host", "network"
event.type	String	Yes	ECS event type	"info", "connection"
observer.hostname	String	Yes	Scanning system hostname	"scanner-host.company.com"
observer.ip	Array<String>	No	Scanner IP addresses	["192.168.1.100"]
observer.os.name	String	Yes	Scanner OS name	"Windows", "Linux", "macOS"
observer.os.version	String	No	Scanner OS version	"10.0.19045"
observer.fips_mode_enabled	Boolean	Yes	FIPS 140-2 mode status	true, false
observer.bigfix_client_installed	Boolean	No	Indicates if BigFix client is installed	true, false
observer.bigfix_client_id	String	No	BigFix client ID for asset correlation	"12345678"
observer.software.name	String	Yes	Tool name	"Quantum Readiness Scanner"
observer.software.version	String	Yes	Tool version	"1.0.42"
scan.type	String	Yes	Scan mode	"local", "remote"
scan.target	String	Yes	Original target specification	"example.com:443"
tags	Array<String>	No	Custom scan tags	["prod", "compliance"]

Quantum Readiness Fields (Local Mode Only)NEW

These fields are added to all events when quantum readiness assessment is enabled in local mode (default). Can be disabled with-disable-quantum-readinessflag.

Field	Type	Description	Example
quantum.assessment_id	String	Unique assessment identifier	"qr_20250915_101539_abc123"
quantum.timestamp	String	Assessment timestamp (ISO 8601)	"2025-09-15T10:15:39.123456-07:00"

quantum.assessment_type	String	Type of assessment performed	"comprehensive"
quantum.system_type	String	Classification of system type	"workstation"
quantum.system_role	String	Primary role of the system	"workstation"
quantum.criticality_level	String	System criticality classification	"standard"
quantum.fips_mode_enabled	Boolean	FIPS 140-2 mode status at assessment time	true, false
quantum.overall_score	Integer	Total quantum readiness score (0-100)	64
quantum.readiness_status	String	Overall readiness classification	"Update Required"
quantum.status_color	String	Color code for status visualization	"orange"
quantum.ready_timeline	String	Estimated timeline to quantum readiness	"2-6 months"
quantum.hardware_score.total_score	Integer	Hardware assessment score (0-40)	32
quantum.hardware_score.cpu_score	Integer	CPU assessment score (0-20)	18
quantum.hardware_score.memory_score	Integer	Memory assessment score (0-15)	14
quantum.hardware_score.security_hardware_score	Integer	Security hardware score (0-5)	0

quantum.operating_system_score.total_score	Integer	OS assessment score (0-30)	15
quantum.operating_system_score.version_score	Integer	OS version score (0-20)	12
quantum.operating_system_score.crypto_framework_score	Integer	Crypto framework score (0-10)	3
quantum.crypto_library_score.total_score	Integer	Crypto library score (0-25)	12
quantum.crypto_library_score.openssl_score	Integer	OpenSSL assessment score (0-15)	8
quantum.crypto_library_score.system_crypto_score	Integer	System crypto score (0-10)	4
quantum.network_score.total_score	Integer	Network assessment score (0-5)	5
quantum.network_score.bandwidth_score	Integer	Bandwidth score (0-3)	3
quantum.network_score.protocol_capability_score	Integer	Protocol capability score (0-2)	2
quantum.recommendations	Array<String>	Actionable recommendations	["Upgrade to macOS 15.0+"]
quantum.detailed_report	String	Comprehensive assessment summary	"System shows moderate quantum readiness..."

Network Target Fields (Remote Scans)

Field	Type	Description	Example
destination.address	String	Target hostname or IP	"example.com"
destination.ip	String	Resolved IP address	"203.0.113.1"
destination.domain	String	Domain name	"example.com"
destination.port	Integer	Target port number	443
network.protocol	String	Network protocol	"tcp"
network.transport	String	Transport protocol	"tcp"
service.name	String	Detected service	"https", "ssh"
service.version	String	Service version (if detected)	"TLSv1.3"

Cipher Suite Event Fields

Event Action:cipher_suite_discovered

Field	Type	Description	Example
tls.version	String	TLS protocol version	"TLSv1.3"
tls.cipher	String	IANA cipher suite name	"TLS_AES_256_GCM_SHA384"
tls.cipher_hex	String	Hex identifier	"0x13,0x02"
tls.key_length	Integer	Key length in bits	256
tls.negotiated_group	String	Key exchange group	"X25519"
tls.is_preferred	Boolean	Server's preferred choice	true
tls.openssl_name	String	OpenSSL cipher name	"ECDHE-RSA-AES256-GCM-SHA384"
security.level	String	Security assessment	"high", "medium", "low"
security.recommendation	String	Security recommendation	"recommended", "legacy"
security.pqc_ready	Boolean	Post-quantum ready	false
security.vulnerabilities	Array<String>	Known vulnerabilities	["BEAST", "CRIME"]
security.nist_category	String	NIST classification	"Recommended"

Certificate Event Fields

Event Actions:network_certificate_discovered,filesystem_certificate_discovered

Field	Type	Description	Example
x509.subject.common_name	String	Certificate subject CN	"example.com"
x509.subject.organization	String	Subject organization	"Example Corp"
x509.subject.organizational_unit	String	Subject OU	"IT Department"
x509.subject.country	String	Subject country	"US"
x509.subject.state_or_province	String	Subject state/province	"California"
x509.subject.locality	String	Subject locality	"San Francisco"
x509.subject.email_address	String	Subject email	"admin@example.com"

x509.subject.raw	String	Complete subject DN	"CN=example.com,O=Example Corp,C=US"
x509.issuer.common_name	String	Issuer CN	"DigiCert TLS RSA SHA256 2020 CA1"
x509.issuer.organization	String	Issuer organization	"DigiCert Inc"
x509.issuer.country	String	Issuer country	"US"
x509.issuer.raw	String	Complete issuer DN	"CN=DigiCert TLS RSA SHA256 2020 CA1,O=DigiCert Inc,C=US"
x509.serial_number	String	Certificate serial number	"12345678901234567890123456 7890"
x509.signature_algorithm	String	Signature algorithm	"SHA256-RSA"
x509.public_key.algorithm	String	Public key algorithm	"RSA", "ECDSA"
x509.public_key.size	Integer	Public key size in bits	2048
x509.public_key.curve	String	Curve name (ECDSA)	"secp384r1"
x509.not_before	String	Valid from date (ISO 8601)	"2024-03-01T00:00:00Z"
x509.not_after	String	Valid until date (ISO 8601)	"2025-03-01T23:59:59Z"
x509.is_expired	Boolean	Whether cert is expired	false
x509.days_until_expiry	Integer	Days until expiration	182
x509.is_self_signed	Boolean	Self-signed certificate	false
x509.is_ca	Boolean	CA certificate	false
x509.fingerprint.sha256	String	SHA-256 fingerprint	"ab:cd:ef:12:34:..."
x509.fingerprint.sha1	String	SHA-1 fingerprint	"12:34:56:78:..."
x509.san	Array<Strin g>	Subject alt names	["*.example.com"]
x509.key_usage	Array<Strin g>	Key usage extensions	["digitalSignature"]
x509.extended_key_usage	Array<Strin g>	Extended key usage	["serverAuth"]

Filesystem Event Fields

Event Action:filesystem_certificate_discovered

Field	Type	Description	Example
file.path	String	Full path to certificate file	"/etc/ssl/certs/ca.pem"
file.name	String	File name only	"ca.pem"
file.extension	String	File extension	".pem"
file.size	Integer	File size in bytes	2048
file.created	String	File creation time	"2024-01-15T10:30:00Z"
file.modified	String	Last modification time	"2024-01-15T10:30:00Z"
file.owner	String	File owner	"root", "DOMAIN\\user"
file.permissions	String	File permissions	"644", "rw-r--r--"
file.hash.sha256	String	SHA-256 hash of file	"a1b2c3d4..."
certificate.format	String	Certificate format	"PEM", "DER", "PKCS12"
certificate.type	String	Certificate type	"x509", "pkcs12"

Memory Library Event Fields

Event Action:crypto_library_discovered

Field	Type	Description	Example
process.pid	Integer	Process ID	1234
process.name	String	Process executable name	"nginx.exe"
process.executable	String	Full executable path	"C:\\nginx\\nginx.exe"
process.command_line	String	Complete command line	"nginx.exe -c nginx.conf"
process.username	String	Process owner	"SYSTEM"
library.name	String	Crypto library name	"OpenSSL"
library.version	String	Library version	"3.0.8"
library.path	String	Library file path	"C:\\openssl\\libssl.dll"
library.crypto_type	String	Crypto implementation type	"openssl", "bcrypt", "java_crypto"
library.product_name	String	Product name from metadata	"OpenSSL Toolkit"
library.company_name	String	Company from metadata	"The OpenSSL Project"
library.file_description	String	File description	"OpenSSL Shared Library"
library.hash.sha256	String	SHA-256 hash of library	"e1f2a3b4..."

Keystore Event FieldsNEW

Event Actions:keystore_discovered,keystore_certificate_discovered

Field	Type	Description	Example
keystore.path	String	Full path to keystore file	"/home/user/keystore.p12"
keystore.type	String	Keystore format type	"PKCS12", "JKS", "Windows", "macOS"
keystore.accessible	Boolean	Whether keystore is accessible	true
keystore.requires_auth	Boolean	Whether authentication required	false
keystore.cert_count	Integer	Number of certificates found	15
keystore.owner	String	File owner (if available)	"domain\\username"
keystore.permissions	String	File permissions	"rw-r--r--"
keystore.size	Integer	File size in bytes	2048576
keystore.last_modified	String	Last modification timestamp	"2024-12-01T10:30:00Z"
keystore.error_message	String	Error details if access failed	"Password required"
keystore_certificate.alias	String	Certificate alias in keystore	"my-server-cert"
keystore_certificate.has_private_key	Boolean	Whether private key available	true
keystore_certificate.chain_length	Integer	Certificate chain length	3
keystore_certificate.chain_complete	Boolean	Whether chain is complete	true
keystore_certificate.vulnerable	Boolean	Whether has vulnerabilities	false
keystore_certificate.risk_level	String	Risk assessment level	"low", "medium", "high"
keystore_certificate.pqc_vulnerable	Boolean	Quantum vulnerability status	true

keystore_certificate.pqc_reason	String	Reason for PQC vulnerability	"RSA algorithm vulnerable"

SSH Host Key Event Fields

Event Action:ssh_host_key_discovered

Field	Type	Description	Example
ssh.host_key.algorithm	String	SSH key algorithm	"ssh-rsa", "ecdsa-sha2-nistp256"
ssh.host_key.size	Integer	Key size in bits	2048
ssh.host_key.curve	String	Elliptic curve (ECDSA)	"nistp256"
ssh.host_key.fingerprint.md5	String	MD5 fingerprint (legacy)	"12:34:56:78:..."
ssh.host_key.fingerprint.sha256	String	SHA-256 fingerprint	"SHA256:abcd..."
ssh.host_key.public_key	String	Base64 public key data	"AAAAB3NzaC1yc2E..."
ssh.host_key.is_weak	Boolean	Cryptographically weak key	false
ssh.banner	String	SSH server banner	"SSH-2.0-OpenSSH_8.9"
ssh.server_version	String	SSH server software	"OpenSSH_8.9"

Outlook Archive Event Fields

Event Action:outlook_archive_discovered

Field	Type	Description	Example
file.path	String	Path to PST/OST file	"C:\\Users\\user\\archive.pst"
file.size	Integer	Archive file size	1048576000
outlook.is_encrypted	Boolean	Archive encryption status	true
outlook.encryption_type	String	Encryption method	"Compressible", "High"
outlook.version	String	Outlook version	"2019", "365"
user.name	String	Archive owner username	"john.doe"

VPN Client Event FieldsNEW

Event Action:vpn_client_discovered

Field	Type	Description	Example
vpn.client_name	String	VPN client application name	"Palo Alto GlobalProtect"

vpn.vendor	String	Software vendor	"Palo Alto Networks"
vpn.version	String	Client version	"6.3.2-525"
vpn.install_path	String	Installation directory	"/Applications/GlobalProtect.app"
vpn.config_path	String	Configuration file location	"~/Library/Application Support/..."
vpn.executable_path	String	Main executable path	"/Applications/.../GlobalProtect"
vpn.service_name	String	System service identifier	"com.paloaltonetworks.globalprotect"
vpn.status	String	Current operational status	"active", "inactive", "unknown"
vpn.detection_method	String	How client was discovered	"filesystem", "registry", "process"
vpn.detection_confidence	String	Detection accuracy level	"high", "medium", "low"
vpn.pqc_ready	Boolean	Post-quantum cryptography support	true
vpn.quantum_resistance	String	Level of quantum resistance	"high", "medium", "low", "none"
vpn.pqc_migration_status	String	PQC migration readiness	"ready", "partial", "not_ready"
vpn.supported_pqc_algorithms	Array<String>	Supported PQC algorithms	["ML-KEM-512", "ML-DSA-44"]
process.pid	Integer	Process ID (if running)	4473

IPSec Tunnel Event FieldsNEW

Event Action:ipsec_tunnel_discovered

Field	Type	Description	Example
ipsec.tunnel_name	String	IPSec tunnel identifier	"strongSwan Site-to-Site"

ipsec.implementation	String	IPSec implementation type	"strongswan", "libreswan", "macOS"
ipsec.config_path	String	Configuration file location	"/etc/ipsec.conf"
ipsec.status	String	Current tunnel status	"active", "inactive", "unknown"
ipsec.detection_method	String	How tunnel was discovered	"config_file", "process", "kernel"
ipsec.detection_confidence	String	Detection accuracy level	"high", "medium", "low"
ipsec.local_subnet	String	Local network subnet	"192.168.1.0/24"
ipsec.remote_subnet	String	Remote network subnet	"10.0.0.0/24"
ipsec.gateway	String	Remote gateway IP address	"203.0.113.1"
ipsec.encryption_algorithms	Array<String>	Configured encryption algorithms	["aes256", "aes128"]
ipsec.integrity_algorithms	Array<String>	Configured hash algorithms	["sha256", "sha1"]
ipsec.key_exchange_groups	Array<String>	Configured DH groups	["modp2048", "ecp256"]
ipsec.pqc_ready	Boolean	Post-quantum cryptography support	false
ipsec.quantum_resistance	String	Level of quantum resistance	"high", "medium", "low", "none"
ipsec.pqc_migration_status	String	PQC migration readiness	"ready", "partial", "not_ready"

Security Intelligence Fields

Applied to cipher suites and cryptographic assets

Field	Type	Description	Possible Values
intel.security_level	String	Overall security assessment	"high", "medium", "low", "insecure"
intel.recommendation	String	Security recommendation	"recommended", "acceptable", "legacy", "avoid"
intel.pqc_ready	Boolean	Post-quantum ready	false
intel.pqc_vulnerable	Boolean	Quantum vulnerable	true
intel.vulnerabilities	Array<String>	Known vulnerabilities	["BEAST", "CRIME", "POODLE"]

intel.nist_category	String	NIST security category	"Recommended", "Legacy-Use", "Deprecated"
intel.friendly_name	String	Human-readable name	"AES-256 with GCM and SHA-384"
intel.description	String	Detailed description	"Advanced Encryption Standard with..."
intel.risk_score	Integer	Numeric risk score (0-100)	25

Event Action Types

Network Discovery Events

- cipher_suite_discovered
- network_certificate_discovered
- ssh_host_key_discovered
- tls_handshake_completed
- protocol_detected

Local Discovery Events

- filesystem_certificate_discovered
- crypto_library_discovered
- outlook_archive_discovered
- private_key_discovered
- java_keystore_discovered
- keystore_discoveredNEW
- keystore_certificate_discoveredNEW
- vpn_client_discoveredNEW
- ipsec_tunnel_discoveredNEW

Sample Records

Cipher Suite Record

JSON:


```
{"observer.hostname":"scanner-
host","observer.software_version":"1.0.42","scan.type":"remote","scan.timestamp"
:"2025-09-02T09:00:17-
04:00","target_host.address":"example.com","target_host.ip":"93.184.216.34","por
t.number":443,"port.status_overall":"open","port.protocol_detected":"TLS","ciphe
r.protocol":"TLSv1.3","cipher.cipher_suite":"TLS_AES_256_GCM_SHA384","cipher.key
_length_bits":256,"cipher.negotiated_group":"X25519","cipher.is_preferred":true,
"cipher.intel.security_level":"high","cipher.intel.recommendation":"recommended"
,"x509.subject.distinguished_name":"CN=example.com,O=Example
Corp,C=US","x509.serial_number":"123456789012345678901234567890","hash.sha256_ce
rtificate":"ab:cd:ef:12:34:56:78:90:..."}
```


Filesystem Certificate Record

JSON:

{"observer.hostname":"scanner-host","scan.type":"local","event.action":"filesystem_certificate_discovered","file.path":"/etc/ssl/certs/ca-cert.pem","certificate.subject.common_name":"Internal Root CA","certificate.issuer.common_name":"Internal Root CA","certificate.serial_number":"123456789","x509.is_valid":true,"hash.sha256_certificate":"12:34:56:78:90:ab:cd:ef:..."}

Crypto Library Record

JSON:

{"observer.hostname":"scanner-host","scan.type":"local","event.action":"crypto_library_in_memory","process.pid":1234,"process.name":"nginx","process.executable":"/usr/sbin/nginx","cryptolibrary.name":"libssl.so.3","cryptolibrary.path":"/usr/lib/x86_64-linux-gnu/libssl.so.3","cryptolibrary.crypto_type":"TLS Library","cryptolibrary.detected_apis":"SSL_connect,SSL_accept,TLS_method"}

Keystore Certificate RecordNEW

JSON:

{"@timestamp":"2025-09-17T14:30:00.000Z","observer.hostname":"scanner-host","scan.type":"local","event.action":"keystore_certificate_discovered","event.category":"file","event.dataset":"keystore_certificate","keystore.path":"/Users/admin/Documents/certificates/server.p12","keystore.type":"PKCS12","keystore.accessible":true,"keystore.cert_count":3,"keystore_certificate.alias":"server-cert","keystore_certificate.has_private_key":true,"keystore_certificate.chain_length":2,"x509.subject.distinguished_name":"CN=api.example.com,O=Example Corp,C=US","x509.issuer.distinguished_name":"CN=Example Internal CA,O=Example Corp,C=US","x509.serial_number":"0x1a2b3c4d5e6f7890","x509.public_key.algorithm":"RSA","x509.public_key.size":2048,"x509.signature_algorithm":"SHA256-RSA","x509.not_before":"2024-01-01T00:00:00Z","x509.not_after":"2025-12-31T23:59:59Z","keystore_certificate.vulnerable":false,"keystore_certificate.risk_level":"medium","keystore_certificate.pqc_vulnerable":true,"keystore_certificate.pqc_reason":"RSA algorithm vulnerable to quantum cryptanalysis"}

Key Features

Data Completeness

- Complete Coverage: All JSON data included
- No Field Filtering: Every certificate field exported
- Certificate Chains: Full chain data flattened
- Process Information: Complete process metadata

Flattening Approach

- Dot Notation: Nested objects flattened with dots
- Array Handling: String arrays joined with commas
- Time Formatting: ISO 8601 RFC3339Nano format
- Intel Maps: Recursively flattened key-value data

Event Types

Network Cipher Events
One record per cipher suite negotiated on network ports

Filesystem Certificate Events
One record per certificate file discovered

Memory Crypto Library Events
One record per crypto library found in process memory

Java Crypto Library Events
One record per Java crypto library discovered

Keystore Certificate EventsNEW
One record per certificate found in keystores (PKCS12, JKS, System Stores)

Integration Examples

Elasticsearch Ingestion
POWERSHELL:

```
# Direct streaming to Elasticsearch
`.\certscanner-windows-amd64.exe -host internal-network.txt -cipherscan `
  -outputformat flatndjson `
  -posttoelastic -elasticnode "https://elastic.company.com:9200" `
  -elasticindex "crypto-scans"

# File-based ingestion
`.\certscanner-windows-amd64.exe -mode local -scanfilesystem `
  -outputformat flatndjson -output certs.ndjson
curl -X POST "elastic.company.com:9200/certs/_bulk" `
  -H "Content-Type: application/x-ndjson" `
  --data-binary "@certs.ndjson"
```

BASH:

```
# Direct streaming to Elasticsearch
./certscanner-linux-x64 -host internal-network.txt -cipherscan \
  -outputformat flatndjson \
  -posttoelastic -elasticnode "https://elastic.company.com:9200" \
  -elasticindex "crypto-scans"

# File-based ingestion
./certscanner-linux-x64 -mode local -scanfilesystem \
  -outputformat flatndjson -output certs.ndjson
curl -X POST "elastic.company.com:9200/certs/_bulk" \
  -H "Content-Type: application/x-ndjson" \
  --data-binary "@certs.ndjson"
```

BASH:

```
# Direct streaming to Elasticsearch - Intel Macs
./certscanner-darwin-amd64 -host internal-network.txt -cipherscan \
  -outputformat flatndjson \
  -posttoelastic -elasticnode "https://elastic.company.com:9200" \
```

```
  -elasticindex "crypto-scans"

# File-based ingestion - Intel Macs
./certscanner-darwin-amd64 -mode local -scanfilesystem \
  -outputformat flatndjson -output certs.ndjson

# For Apple Silicon Macs, use:
# ./certscanner-darwin-arm64 [same arguments]

curl -X POST "elastic.company.com:9200/certs/_bulk" \
  -H "Content-Type: application/x-ndjson" \
  --data-binary "@certs.ndjson"
```

Log Analysis Pipeline

POWERSHELL:

```
# Stream processing with jq
`.\certscanner-windows-amd64.exe -host servers.txt -outputformat flatndjson | `
  jq 'select(.cipher.intel.security_level == "low")' | `
  jq '.target_host.address + ":" + (.port.number | tostring) + " - " +
.cipher.cipher_suite'

# Filter for expiring certificates
`.\certscanner-windows-amd64.exe -mode local -scanfilesystem -outputformat
flatndjson | `
  jq 'select(.event.action == "filesystem_certificate_discovered") |
select(.x509.is_valid == false)'
```

BASH:

```
# Stream processing with jq
./certscanner-linux-x64 -host servers.txt -outputformat flatndjson | \
  jq 'select(.cipher.intel.security_level == "low")' | \
  jq '.target_host.address + ":" + (.port.number | tostring) + " - " +
.cipher.cipher_suite'

# Filter for expiring certificates
./certscanner-linux-x64 -mode local -scanfilesystem -outputformat flatndjson | \
  jq 'select(.event.action == "filesystem_certificate_discovered") |
select(.x509.is_valid == false)'
```

BASH:

```
# Stream processing with jq - Intel Macs
./certscanner-darwin-amd64 -host servers.txt -outputformat flatndjson | \
  jq 'select(.cipher.intel.security_level == "low")' | \
  jq '.target_host.address + ":" + (.port.number | tostring) + " - " +
.cipher.cipher_suite'

# Filter for expiring certificates - Intel Macs (no memory scanning)
./certscanner-darwin-amd64 -mode local -scanfilesystem -outputformat flatndjson
| \
  jq 'select(.event.action == "filesystem_certificate_discovered") |
select(.x509.is_valid == false)'

# For Apple Silicon Macs, replace -darwin-amd64 with -darwin-arm64
```

HTML Format

Overview

The HTML output format generates interactive web-based reports ideal for human review, presentations, and executive reporting. Reports include certificate detail popups, search functionality, and responsive design.

Interactive Features

- • Certificate detail popups with full X.509 data
- • Real-time search and filtering
- • Cipher suite intelligence tooltips
- • Responsive design for all devices
- • VPN client discovery with PQC assessmentsNEW
- • IPSec tunnel configuration analysisNEW
- • System quantum readiness assessment with scoring dashboardNEW

Usage

BASH:

```
.\certscanner-windows-amd64.exe -host example.com `
  -outputformat html `
  -output security-report.html
```

Report Features

Search & Filter

Real-time search across all scan results with instant filtering

Certificate Popups

Click any certificate to view complete X.509 details in modal

Intelligence Integration

Cipher suite recommendations and security analysis

Network Results Display

- Host Summary: IP address, domain, open ports
- Certificate Table: Clickable certificates with validity status
- Cipher Suites: Security level color coding
- SSH Information: Host keys and algorithm details

Local Results Display

- Process Information: PID, executable, crypto libraries
- Filesystem Certificates: File paths with certificate popups
- Memory Libraries: Crypto libraries with API detection
- Archive Files: Outlook archives with encryption status
- Quantum Readiness: System PQC assessment with scoring dashboardNEW

Technical Implementation

Styling & Layout

- Tailwind CSS: Utility-first styling
- Responsive Design: Mobile-friendly layout
- Color Coding: Security level indicators
- Print Optimized: Professional PDF generation

JavaScript Features

- Vanilla JS: No external dependencies
- Modal System: Certificate detail popups
- Search Engine: Real-time filtering
- Data Embedding: JSON data in template

Template Functions

Function	Purpose	Example
formatTime	Format timestamps	"2025-09-02 09:00:17"
getSecurityClass	CSS class for security level	"security-high"
toJSON	Embed data for JavaScript	JSON.stringify(object)
truncateString	Limit string length	"long string..." (50 chars)

Customization & Integration

Report Customization

BASH:

```
# Custom branding with tags
.\certscanner-windows-amd64.exe -host internal-servers.txt -cipherscan `
  -tags "CompanyName,Q4-2025,Security-Audit" `
  -outputformat html -output Q4-Security-Report.html

# Focused filesystem audit
.\certscanner-windows-amd64.exe -mode local -scanfilesystem -scanoutlookarchives
`
  -tags "Certificate-Inventory,Compliance" `
  -outputformat html -output Certificate-Audit.html
```

Template Structure

The HTML template includes these main sections:

- Header: Scan summary with system information
- Network Results: Host-by-host analysis with port details
- Filesystem Results: Certificate files with metadata
- Memory Results: Process crypto libraries
- Archive Results: Outlook archives with encryption status
- •JavaScript: Search, modal, and interaction logic

Sample Report Structure

HTML Report Layout

HTML:

```
<!-- Report Header -->
<div class="scan-summary">
  <h1>TYCHON Quantum Readiness Security Report</h1>
  <div class="scan-metadata">
    <span>Target: example.com</span>
    <span>Timestamp: 2025-09-02 09:00:17</span>
    <span>Version: 1.0.43</span>
  </div>
</div>

<!-- Search Interface -->
<div class="search-container">
  <input type="text" id="searchInput" placeholder="Search certificates, hosts,
ciphers...">
</div>

<!-- Network Results -->
<div class="network-results">
  <h2>Network Scan Results</h2>
  <div class="host-result">
    <h3>example.com (93.184.216.34)</h3>
    <div class="port-result">
      <h4>Port 443 (TLS)</h4>
      <table class="certificate-table">
        <tr onclick="showCertDetails(...)">
          <td>example.com</td>
          <td>DigiCert</td>
          <td class="status-valid">Valid</td>
        </tr>
      </table>
      <table class="cipher-table">
        <tr class="security-high">
          <td>TLS_AES_256_GCM_SHA384</td>
          <td>256-bit</td>
          <td>High Security</td>
        </tr>
      </table>
    </div>
  </div>
</div>

<!-- Certificate Modal -->
<div id="certificateModal" class="modal">
  <div class="modal-content">
    <h3>Certificate Details</h3>
    <div class="cert-details">
      <!-- Complete X.509 certificate information -->
    </div>
  </div>
</div>
```

Use Cases

Executive Reporting

- Security Dashboards: Visual summaries for leadership
- Compliance Reports: Professional presentation format
- Audit Documentation: Complete certificate inventories
- Risk Assessment: Security level visualizations

Technical Analysis

- Certificate Management: Detailed X.509 inspection
- Network Security: Cipher suite analysis
- System Auditing: Process and library review
- Troubleshooting: Interactive data exploration

Example Workflows

BASH:

```
# Generate executive security report
.\certscanner-windows-amd64.exe -host critical-infrastructure.txt -cipherscan `
  -tags "Executive-Report,Q4-2025,Critical-Systems" `
  -outputformat html -output Executive-Crypto-Report.html

# Local system security audit
.\certscanner-windows-amd64.exe -mode local -scanfilesystem -scanmemory -
scanconnected `
  -tags "System-Audit,Workstation-Security" `
  -outputformat html -output System-Security-Audit.html

# Certificate expiration dashboard
.\certscanner-windows-amd64.exe -mode local -scanfilesystem `
  -tags "Certificate-Management,Expiration-Tracking" `
  -outputformat html -output Certificate-Dashboard.html
```

JavaScript API Reference

Core Functions

Function	Purpose	Parameters
showCertDetails()	Display certificate popup	certificate object
showFileCertDetails()	Display filesystem cert popup	certificate object
closeModal()	Close certificate modal	none
filterResults()	Search and filter data	search term

CSS Classes

Class	Purpose	Visual Effect
.security-high	High security cipher	Green background
.security-medium	Medium security cipher	Yellow background
.security-low	Low security cipher	Red background
.status-valid	Valid certificate	Green text
.status-expired	Expired certificate	Red text

Browser Compatibility

Supported Browsers

Reports work offline and require no external dependencies

EventLog Format

Overview

The EventLog format writes scan results directly to the Windows Event Log, enabling seamless integration with Windows-based SIEM systems and enterprise monitoring solutions.

Platform Support

- Windows: Native Event Log API (Application log)
- macOS: Unified logging system (visible via log show commands)
- Linux: System log (rsyslog/journald)
- Event Source: "Quantum Readiness Scanner"
- New: VPN client and IPSec tunnel eventsNEW

Usage

POWERSHELL:

```
`.\certscanner-windows-amd64.exe -mode local `
  -outputformat eventlog
```

Requires Administrator privileges

BASH:

```
# Intel Macs
./certscanner-darwin-amd64 -mode local -outputformat eventlog

# Apple Silicon Macs
./certscanner-darwin-arm64 -mode local -outputformat eventlog
```

Events written to unified logging system

BASH:

```
./certscanner-linux-x64 -mode local -outputformat eventlog
```

Events written to system log (journald/rsyslog)

Event Log Structure

Event Properties

Property	Value	Description
Source	Quantum Readiness Scanner	Application event source
Log	Application	Windows Application Event Log
Event Type	Information	Informational event level
Event ID	1001-1004	Different IDs for each asset type
Message	JSON Data	Complete JSON with all scan data

Event ID Mapping

Event ID	Event Type	Description
1001	Cipher Discovery	TLS cipher suite discovered
1002	Certificate Discovery	Filesystem certificate found

1003	Library Discovery	Crypto library in memory
1004	Archive Discovery	Outlook archive file found
1005	Quantum Readiness Assessment	System quantum readiness evaluation (local mode only)NEW

Event Message JSON Schema

Each event log entry contains a complete JSON message with scan data. To ensure reliable delivery to system logs, verbose fields are automatically removed including: raw certificate data (raw_pem, signature_hex, rsa_modulus_hex, raw_der_base64), detailed certificate extensions (basic_constraints, subject_alternative_names), non-essential timestamps, and verbose process details.

Message Optimization

To ensure reliable delivery to system logs, verbose certificate fields are automatically removed before logging. All essential certificate metadata is preserved for analysis.

Complete Schema Reference

All event types share common fields with type-specific additions. Optional fields marked with * may not be present in all events.

Field Path	Type	Events	Description
@timestamp	string	All	ISO 8601 timestamp
the scanner.type	string	All	Event type: cipher, filesystem, library, java_crypto_library, archivefile
the scanner.scan_mode	string	All	Scan mode used
the scanner.scan_timestamp	timestamp	All	When scan was performed
the scanner.scanner_version	string	All	Scanner version number
the scanner.host.machine_serial_number*	string	All	Machine serial number
the scanner.active*	boolean	All	Whether asset is currently active
the scanner.last_seen*	timestamp	All	Last seen timestamp (when not active)
certificate.is_file	string	All	"true" or "false" - whether cert is file-based
tags*	array	All	User-defined tags
Cipher Events (ID 1001) - Network TLS connections			

the scanner.pqc_vulnerable*	boolean	1001	Whether cipher is post-quantum vulnerable
the scanner.cipher.detail.security*	string	1001	Security level (low, medium, high)
the scanner.cipher.detail.is_quantum_ready*	boolean	1001	Post-quantum readiness
the scanner.cipher.detail.algo.auth*	string	1001	Authentication algorithm
the scanner.cipher.is_preferred*	boolean	1001	Whether cipher is server's preferred choice
server.address	string	1001	Target server address
server.ip	string	1001	Target server IP address
server.port	integer	1001	Target server port
service.protocol.type	string	1001	Protocol type (TLS)
service.protocol.name	string	1001	Protocol version name
tls.cipher	string	1001	TLS cipher suite name
tls.version_protocol	string	1001	TLS protocol name
tls.version	string	1001	TLS version number
X.509 Certificate Fields - Present in events 1001, 1002			
x509.version_number	integer	1001,1002	X.509 certificate version
x509.serial_number	string	1001,1002	Certificate serial number
x509.signature_algorithm	string	1001,1002	Signature algorithm used
x509.issuer.common_name	string	1001,1002	Issuer common name
x509.issuer.country	string	1001,1002	Issuer country code
x509.issuer.locality	string	1001,1002	Issuer locality
x509.issuer.organization	string	1001,1002	Issuer organization
x509.issuer.organizational_unit	string	1001,1002	Issuer organizational unit
x509.issuer.state_or_province	string	1001,1002	Issuer state or province
x509.subject.*	string	1001,1002	Same fields as issuer for subject

x509.not_before	timestamp	1001,1002	Certificate valid from date
x509.not_after	timestamp	1001,1002	Certificate expiration date
x509.is_valid	boolean	1001,1002	Whether certificate is currently valid
x509.public_key_algorithm	string	1001,1002	Public key algorithm
x509.public_key_size	integer	1001,1002	Public key size in bits
x509.is_self_signed	boolean	1001,1002	Whether certificate is self-signed
x509.hash	string	1001,1002	SHA256 fingerprint
Process Information - Present in events 1001, 1003			
process.pid	integer	1001,1003	Process ID
process.name	string	1001,1003	Process name
process.executable	string	1001,1003	Executable file path
File Information - Present in events 1001, 1002, 1003, 1004			
file.path	string	All	Full file path
file.name	string	All	File name only
file.size*	integer	1001,1002,1004	File size in bytes
file.mtime*	timestamp	1001,1002,1004	Last modification time
file.hash.sha1*	string	1001,1002,1004	SHA1 hash
file.hash.sha256*	string	1001,1002,1004	SHA256 hash
PE/Library Information - Present in events 1001, 1003			
pe.file_version*	string	1001,1003	File version from PE header
pe.product_version*	string	1001,1003	Product version
Archive Information - Present in event 1004			
archive.encrypted	boolean	1004	Whether archive is encrypted
archive.type	string	1004	Archive type (PST, OST)
Quantum Readiness Assessment - Present in event 1005NEW			

quantum.assessment_id	string	1005	Unique assessment identifier
quantum.system_type	string	1005	System classification (workstation, server)
quantum.criticality_level	string	1005	System criticality level
quantum.overall_score	integer	1005	Total quantum readiness score (0-100)
quantum.readiness_status	string	1005	Overall readiness status
quantum.hardware_score	integer	1005	Hardware assessment score (0-40)
quantum.os_score	integer	1005	Operating system score (0-30)
quantum.crypto_score	integer	1005	Crypto library score (0-25)
quantum.network_score	integer	1005	Network readiness score (0-5)
quantum.recommendations	string	1005	Comma-separated actionable recommendations
quantum.timeline	string	1005	Estimated timeline to quantum readiness
archive.type	string	1004	Archive type (e.g., outlook_pst)
archive.encryption.enabled	boolean	1004	Whether archive is encrypted
archive.encryption.type	string	1004	Encryption type
archive.encryption.strength	string	1004	Encryption strength
archive.format.version	string	1004	Archive format version

Cipher Event Schema (Event ID 1001) - Cross-Platform

JSON:

```
{
  "@timestamp": "2025-09-09T13:00:17.000Z",
  "tychon": {
    "type": "cipher",
    "scan_mode": "local",
    "scan_timestamp": "2025-09-09T13:00:17.000Z",
```

```
  "scanner_version": "1.0.43",
  "host": {
    "machine_serial_number": "ABC123DEF456"
  },
  "pqc_vulnerable": true,
  "cipher": {
    "detail": {
      "security": "high",
      "is_quantum_ready": false,
      "algo": {
        "auth": "RSA"
      }
    },
    "is_preferred": true
  },
  "certificate": {
    "validity": {
      "duration_days": 365
    },
    "public_key": {
      "rsa_exponent": 65537
    }
  },
  "active": true,
  "last_seen": "2025-09-09T12:30:17.000Z"
},
"certificate": {
  "is_file": "false"
},
"server": {
  "address": "192.168.1.10",
  "ip": "192.168.1.10",
  "port": 443
},
"service": {
  "protocol": {
    "type": "TLS",
    "name": "TLSV1_3"
  }
},
"tls": {
  "cipher": "TLS_AES_256_GCM_SHA384",
  "version_protocol": "TLS",
  "version": "1.3"
},
"x509": {
  "version_number": 3,
  "serial_number": "12345678901234567890",
  "signature_algorithm": "SHA256-RSA",
  "issuer": {
    "common_name": "Company Internal CA",
    "country": "US",
    "organization": "Company Inc"
```

```
    },
    "subject": {
      "common_name": "internal.company.com",
      "country": "US",
      "organization": "Company Inc"
    },
    "not_before": "2024-09-09T00:00:00.000Z",
    "not_after": "2025-09-09T23:59:59.000Z",
    "is_valid": true,
    "public_key_algorithm": "RSA",
    "public_key_size": 2048,
    "is_self_signed": false,
    "hash":
"ab:cd:ef:12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:e
f:12"
  },
  "process": {
    "pid": 1234,
    "name": "firefox",
    "executable": "/usr/bin/firefox"
  },
  "file": {
    "path": "/usr/bin/firefox",
    "name": "firefox",
    "size": 2097152,
    "mtime": "2024-08-20T14:15:00.000Z",
    "hash": {
      "sha1": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
      "sha256":
"e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855"
    }
  },
  "pe": {
    "file_version": "127.0.6533.120",
    "product_version": "127.0.6533.120"
  },
  "tags": ["production", "web-servers"]
}
```

Filesystem Certificate Schema (Event ID 1002)

JSON:

```
{
  "@timestamp": "2025-09-09T13:00:17.000Z",
  "tychon": {
    "type": "filesystem",
    "scan_mode": "local",
    "scan_timestamp": "2025-09-09T13:00:17.000Z",
    "scanner_version": "1.0.43",
    "host": {
      "machine_serial_number": "ABC123DEF456"
    },
    "active": true
  },
```

```
  "certificate": {
    "is_file": "true"
  },
  "file": {
    "path": "/etc/ssl/certs/ca-certificate.crt",
    "name": "ca-certificate.crt",
    "size": 4096,
    "mtime": "2024-09-01T10:00:00.000Z",
    "hash": {
      "sha1": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
      "sha256":
"e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855"
    }
  },
  "x509": {
    "version_number": 3,
    "serial_number": "98765432109876543210",
    "signature_algorithm": "SHA256-RSA",
    "subject": {
      "common_name": "Company Root CA",
      "country": "US",
      "organization": "Company Inc"
    },
    "issuer": {
      "common_name": "Company Root CA",
      "country": "US",
      "organization": "Company Inc"
    },
    "not_before": "2020-01-01T00:00:00.000Z",
    "not_after": "2030-01-01T23:59:59.000Z",
    "is_valid": true,
    "public_key_algorithm": "RSA",
    "public_key_size": 4096,
    "is_self_signed": true,
    "hash":
"12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:ef:12:34:56:78:90:ab:cd:ef:12:34:5
6:78"
  }
}
```


Crypto Library Schema (Event ID 1003)

JSON:


```
{
  "@timestamp": "2025-09-09T13:00:17.000Z",
  "tychon": {
    "type": "library",
    "scan_mode": "local",
    "scan_timestamp": "2025-09-09T13:00:17.000Z",
    "scanner_version": "1.0.43",
    "host": {
      "machine_serial_number": "ABC123DEF456"
    },
    "active": true
```

```
  },
  "certificate": {
    "is_file": "true"
  },
  "process": {
    "pid": 5678,
    "name": "java",
    "executable": "/usr/bin/java"
  },
  "file": {
    "path": "/usr/lib/jvm/java-11-openjdk/lib/security/bcprov.jar",
    "name": "bcprov.jar"
  },
  "pe": {
    "file_version": "1.70.0.0",
    "product_version": "1.70"
  }
}
```

Archive File Schema (Event ID 1004)

JSON:

```
{
  "@timestamp": "2025-09-09T13:00:17.000Z",
  "tychon": {
    "type": "archivefile",
    "scan_mode": "local",
    "scan_timestamp": "2025-09-09T13:00:17.000Z",
    "scanner_version": "1.0.43",
    "host": {
      "machine_serial_number": "ABC123DEF456"
    },
    "active": true
  },
  "certificate": {
    "is_file": "true"
  },
  "file": {
    "path": "/home/john/Documents/email/archive.mbox",
    "name": "archive.mbox",
    "size": 2147483648,
    "mtime": "2025-09-08T15:30:00.000Z",
    "hash": {
      "sha1": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
      "sha256":
"e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855"
    }
  },
  "archive": {
    "type": "mbox",
    "encryption": {
      "enabled": true,
      "type": "password_protected",
      "strength": "medium"
```

```
    },
    "format": {
      "version": "2019"
    }
  }
}
```

SIEM Integration

Windows Event Forwarding

- WEF Compatible: Standard Windows Event Log format
- SIEM Ready: Pure JSON in message field
- No Timestamps: Clean JSON without prefixes
- ECS Compliant: Elasticsearch Common Schema fields

Enterprise Monitoring

- Splunk: Direct Windows Event Log ingestion
- Microsoft Sentinel: Azure Event Log connector
- IBM QRadar: Windows Event Log DSM
- LogRhythm: Native Event Log parsing

Cross-Platform Automated Deployment

BATCH:

```
# Windows Scheduled Task
schtasks /create /tn "TYCHON-PQC-Scanner-Hourly" /tr "C:\Tools\certscanner.exe -
mode local -outputformat eventlog" /sc hourly

# PowerShell automation
$scannerPath = "C:\Tools\certscanner.exe"
Start-Process -FilePath $scannerPath -ArgumentList "-mode local -outputformat
eventlog" -Wait
```

BASH:

```
# macOS LaunchDaemon (system-wide)
sudo cp com.tychon.pqc-scanner.plist /Library/LaunchDaemons/
sudo launchctl load /Library/LaunchDaemons/com.tychon.pqc-scanner.plist

# User-level LaunchAgent
cp com.tychon.pqc-scanner.plist ~/Library/LaunchAgents/
launchctl load ~/Library/LaunchAgents/com.tychon.pqc-scanner.plist
```

BASH:

```
# Linux Cron Job
# Add to crontab for hourly execution
0 * * * * /usr/local/bin/certscanner -mode local -outputformat eventlog

# Systemd Timer (preferred for modern Linux)
sudo cp tychon-pqc-scanner.service /etc/systemd/system/
sudo cp tychon-pqc-scanner.timer /etc/systemd/system/
sudo systemctl enable --now tychon-pqc-scanner.timer
```

Viewing Events

Platform-Specific Event Viewing

1. Open Windows Event Viewer
2. Navigate to: Windows Logs → Application
3. Filter by Source: "the scanner-PQC-Scanner"
4. View JSON data in the event details
5. Open Console.app or use command line
6. Filter by Process: "the scanner-pqc-scanner"
7. Use unified log commands for structured queries
8. JSON data appears in log messages
9. Use journalctl or syslog viewers
10. Filter by identifier: "the scanner-pqc-scanner"
11. Parse structured JSON from log entries
12. Integration with rsyslog/syslog-ng for forwarding

Platform-Specific Queries

POWERSHELL:

```
# Get all TYCHON-PQC-Scanner events
Get-WinEvent -FilterHashtable @{LogName='Application'; ProviderName='TYCHON-PQC-
Scanner'}

# Get cipher discovery events (ID 1001)
Get-WinEvent -FilterHashtable @{LogName='Application'; ProviderName='TYCHON-PQC-
Scanner'; ID=1001}

# Export to JSON for analysis
Get-WinEvent -FilterHashtable @{LogName='Application'; ProviderName='TYCHON-PQC-
Scanner'} |
  Select-Object TimeCreated, Id, @{Name='Message';Expression={$_.Message}} |
  ConvertTo-Json -Depth 10 | Out-File -FilePath "tychon-pqc-scanner-events.json"
```

BASH:

```
# View recent TYCHON EventLog entries (recommended for SIEM)
log show --predicate 'eventMessage CONTAINS "TYCHON_SCAN"' --last 1h

# View all TYCHON events with syslog formatting
log show --style syslog --last 1h | grep "TYCHON_SCAN"

# Monitor TYCHON events in real-time
log stream --predicate 'eventMessage CONTAINS "TYCHON_SCAN"'

# View TYCHON events by logger process (all entries)
log show --predicate 'process == "logger" AND eventMessage CONTAINS "tychon-pqc-
scanner"' --last 24h

# Export TYCHON events to file for analysis
log show --predicate 'eventMessage CONTAINS "TYCHON_SCAN"' --last 7d --style
ndjson > tychon-events.ndjson
```

BASH:

```
# View TYCHON events using journalctl
sudo journalctl -t tychon-pqc-scanner --since "1 hour ago"

# Search for specific scan events
sudo journalctl -t tychon-pqc-scanner -g "TYCHON_SCAN" --since today

# Monitor TYCHON events in real-time
sudo journalctl -t tychon-pqc-scanner -f

# Export recent events to file
sudo journalctl -t tychon-pqc-scanner --since "24 hours ago" --no-pager >
tychon-events.log
```

Troubleshooting

Best Practices

- Configure scheduled scans using platform-appropriate methods (Windows Task Scheduler, macOS LaunchDaemons, Linux cron/systemd)
- Set up log forwarding for centralized collection (WEF, syslog, journald forwarding)
- Use specific tags to categorize different scan types and environments
- Monitor log storage and configure appropriate retention policies per platform

Enterprise Integration Examples

Cross-Platform Scheduled Monitoring

BATCH:

```
REM Create scheduled task for daily crypto monitoring
schtasks /create /tn "TYCHON-PQC-Scanner-Daily" ^
  /tr "C:\Security\certscanner.exe -mode local -scanfilesystem -scanmemory -
outputformat eventlog -tags daily-scan" ^
  /sc daily /st 02:00 /ru SYSTEM

REM Create task for network monitoring
schtasks /create /tn "TYCHON-PQC-Scanner-Network" ^
  /tr "C:\Security\certscanner.exe -scanconnected -outputformat eventlog -tags
network-monitoring" ^
  /sc hourly /ru SYSTEM
```

XML:

```
<!-- /Library/LaunchDaemons/com.tychon.pqc-scanner.daily.plist -->
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Label</key>
    <string>com.tychon.pqc-scanner.daily</string>
    <key>ProgramArguments</key>
    <array>
        <string>/usr/local/bin/certscanner</string>
        <string>-mode</string>
        <string>local</string>
```

```
        <string>-scanfilesystem</string>
        <string>-scanmemory</string>
        <string>-outputformat</string>
        <string>eventlog</string>
        <string>-tags</string>
        <string>daily-scan</string>
    </array>
    <key>StartCalendarInterval</key>
    <dict>
        <key>Hour</key>
        <integer>2</integer>
        <key>Minute</key>
        <integer>0</integer>
    </dict>
</dict>
</plist>
```

INI:

```
# /etc/systemd/system/tychon-pqc-scanner-daily.service
[Unit]
Description=TYCHON PQC Scanner Daily Crypto Monitoring
After=network.target

[Service]
Type=oneshot
ExecStart=/usr/local/bin/certscanner -mode local -scanfilesystem -scanmemory -
outputformat eventlog -tags daily-scan
User=root

# /etc/systemd/system/tychon-pqc-scanner-daily.timer
[Unit]
Description=Run TYCHON PQC Scanner daily
Requires=tychon-pqc-scanner-daily.service

[Timer]
OnCalendar=daily
Persistent=true

[Install]
WantedBy=timers.target

# Enable with: sudo systemctl enable --now tychon-pqc-scanner-daily.timer
```

SIEM Query Examples

SQL:

```
-- Splunk: Find PQC-vulnerable ciphers (Cross-Platform)
index=* (source="WinEventLog:Application" SourceName="TYCHON-PQC-Scanner") OR
(sourcetype="syslog" "tychon-pqc-scanner") EventCode=1001
| spath input=Message
| search "cipher.intel.pqc_ready"=false
| table _time, target_host.address, cipher.name, cipher.intel.security_level

-- Microsoft Sentinel: Certificate expiration monitoring (Cross-Platform)
```

```
Event
| where (Source == "TYCHON-PQC-Scanner" and EventID == 1002) or (SyslogTag ==
"tychon-pqc-scanner" and EventID == 1002)
| extend CertData = parse_json(RenderedDescription)
| where todatetime(CertData.certificate.not_after) < now() + 30d
| project TimeGenerated, Computer, CertData.certificate.subject.common_name,
CertData.certificate.not_after
```

Native Format

Overview

The the scanner format is an enhanced NDJSON output optimized for security platforms and SIEM systems. It includes threat intelligence, ECS-compliant field names, and denormalized data for efficient searching and analysis.

Important: Flat JSON Structure

All the scanner format events usecompletely flat JSONwith dot notation (e.g.,crypto.key_algorithm,tls.certificate.subject). No nested objects are supported - all fields are at the root level with descriptive dot-separated names.

Key Features

- ECS (Elastic Common Schema) compliant
- Enhanced threat intelligence
- Post-quantum vulnerability analysis
- Asset tracking with active/inactive status
- Enterprise VPN client discovery
- IPSec tunnel configuration analysis
- Digital certificate keystore discovery
- System quantum readiness assessment

Usage

BASH:

```
.\certscanner-windows-amd64.exe -host example.com `
  -outputformat tychon `
  -output security-scan.tychon.ndjson
```

Event Types

Cipher Events (the scanner.type: "cipher")

Network TLS cipher suites with complete certificate and threat intelligence

Library Events (the scanner.type: "library")

Cryptographic libraries discovered in process memory

Filesystem Events (the scanner.type: "filesystem")

Certificates and crypto files found on disk

Archive Events (the scanner.type: "archivefile")

Encrypted Outlook archive files discovered

VPN Client Events (the scanner.type: "vpn_client")

VPN client software discovered on the system

IPSec Tunnel Events (the scanner.type: "ipsec_tunnel")

IPSec tunnel configurations discovered

Quantum Readiness Events (event.dataset: "the scanner.quantum_readiness")NEW

System quantum readiness assessment (local mode only)

Keystore Events (event.dataset: "the scanner.keystore")NEW

Digital certificate keystores discovered on the system

Keystore Certificate Events (event.dataset: "the scanner.keystore_certificate")NEW

Individual certificate details from keystore discovery

Core Schema Fields

Universal Fields (All Events)

Field	Type	Description
@timestamp	String	ISO 8601 event timestamp
id	String	Unique SHA-1 hash for finding
the scanner.type	String	Event type classification
the scanner.scan_mode	String	Scan mode (local/remote/connected)
the scanner.active	Boolean	Asset active status from tracking
the scanner.pqc_vulnerable	Boolean	Quantum vulnerability assessment
tags	Array	User-defined scan tags

Certificate Fields

Field	Type	Description
certificate.subject.common_name	String	Certificate subject CN
certificate.issuer.common_name	String	Certificate issuer CN
certificate.not_after	String	Certificate expiration
certificate.signature_algorithm	String	Signature algorithm
certificate.fingerprint_sha256	String	SHA-256 fingerprint
certificate.is_file	String	"true" for filesystem certs

Sample Records

Cipher Event Record

JSON:


```
{"@timestamp":"2025-09-
02T13:00:17.123Z","id":"abc123def456","tychon.type":"cipher","tychon.scan_mode":
"remote","tychon.active":true,"tychon.pqc_vulnerable":false,"server.address":"ex
ample.com","server.port":443,"service.protocol.type":"TLS","service.protocol.nam
e":"TLSV1.3","cipher.name":"TLS_AES_256_GCM_SHA384","cipher.key_length_bits":256
,"cipher.negotiated_group":"X25519MLKEM768","cipher.is_preferred":true,"cipher.i
ntel.security_level":"high","cipher.intel.pqc_ready":true,"certificate.subject.c
ommon_name":"example.com","certificate.issuer.common_name":"DigiCert TLS RSA
SHA256 2020 CA1","certificate.not_after":"2025-03-
01T23:59:59Z","certificate.fingerprint_sha256":"ab:cd:ef:12:34:56:78:90","certif
icate.is_file":"false","tags":["production","quarterly-scan"]}
```


Library Event Record

JSON:

{"@timestamp":"2025-09-02T13:00:17.456Z","id":"def789abc123","tychon.type":"library","tychon.scan_mode":"local","tychon.active":true,"process.pid":1234,"process.name":"nginx","process.executable":"/usr/sbin/nginx","file.path":"/usr/lib/libssl.so.3","library.name":"libssl.so.3","library.crypto_type":"TLS Library","library.company_name":"OpenSSL Project","library.detected_apis":"SSL_connect,SSL_accept,TLS_method","tags":["server","crypto-audit"]}

Filesystem Certificate Record

JSON:

{"@timestamp":"2025-09-02T13:00:17.789Z","id":"ghi456jkl789","tychon.type":"filesystem","tychon.scan_mode":"local","tychon.active":true,"file.path":"/etc/ssl/certs/ca-cert.pem","certificate.subject.common_name":"Internal Root CA","certificate.issuer.common_name":"Internal Root CA","certificate.not_after":"2033-01-01T00:00:00Z","certificate.signature_algorithm":"SHA256-RSA","certificate.is_self_signed":true,"certificate.is_file":"true","tags":["filesystem-scan"]}

Keystore Certificate Record

JSON:

{"@timestamp":"2025-09-17T20:35:45.267Z","event.action":"keystore_certificate_detected","event.category":"security","event.type":"info","event.dataset":"tychon.keystore_certificate","id":"52d4362845b11fdaadc28c459bb340211a33cc72","observer.hostname":"workstation-01","observer.name":"tychon-pqc-scanner","observer.type":"scanner","observer.version":"1.0.61","file.path":"macOS:Login","file.name":"macOS:Login","file.extension":"","file.mtime":"2025-09-17T16:23:20.093843-04:00","crypto.key_algorithm":"RSA","crypto.key_size":2048,"crypto.signature_algorithm":"SHA256-RSA","crypto.fingerprint_sha1":"7afc9d01a62f03a2de9637936d4afe68090d2de18d03f29c88cfb0b1ba63587f","tls.certificate.subject":"CN=Developer ID Certification Authority,OU=Apple Certification Authority,O=Apple Inc.,C=US","tls.certificate.issuer":"CN=Apple Root CA,OU=Apple Certification Authority,O=Apple Inc.,C=US","tls.certificate.serial_number":"1763908746353189132","tls.certificate.not_before":"2012-02-01T22:12:15Z","tls.certificate.not_after":"2027-02-01T22:12:15Z","tls.certificate.version":3,"tls.certificate.alias":"Developer ID Certification Authority","tls.certificate.is_ca":true,"tls.certificate.is_self_signed":false,"tls.certificate.has_private_key":false,"keystore.type":"macOS-Keychain","keystore.accessible":true,"keystore.requires_auth":false,"keystore.cert_count":18,"vulnerability.is_vulnerable":false,"vulnerability.risk_level":"medium","vulnerability.risk_reason":"RSA key size below recommended 3072 bits for PQC","pqc.vulnerable":true,"pqc.reason":"RSA key size 2048 bits is below recommended 3072 bits for PQC","tychon.scan_mode":"local","tychon.asset_type":"keystore_certificate","tychon.keystore_type":"macOS-Keychain","tychon.certificate_usage":"stored"}

Keystore Summary Record

JSON:


```
{"@timestamp":"2025-09-
17T20:35:45.123Z","event.action":"keystore_discovered","event.category":"securit
y","event.type":"info","event.dataset":"tychon.keystore","id":"8a1b2c3d4e5f6789a
bcd","observer.hostname":"workstation-01","observer.name":"tychon-pqc-
scanner","observer.type":"scanner","observer.version":"1.0.61","file.path":"macO
S:System","file.name":"macOS:System","file.extension":"","file.mtime":"2025-09-
17T16:23:20.093Z","keystore.type":"macOS-
Keychain","keystore.accessible":true,"keystore.requires_auth":false,"keystore.ce
rt_count":45,"keystore.vulnerable_certificates":8,"keystore.pqc_vulnerable_certi
ficates":12,"keystore.expired_certificates":2,"keystore.expiring_soon_certificat
es":3,"keystore.certificate_types":"ca:15,end_entity:30","keystore.key_algorithm
s":"RSA:35,ECDSA:8,DSA:2","tychon.scan_mode":"local","tychon.asset_type":"keysto
re","tychon.keystore_type":"macOS-Keychain"}
```


Enhanced Intelligence Features

Post-Quantum Analysis

- the scanner.pqc_vulnerable: Quantum attack vulnerability
- cipher.intel.pqc_ready: Post-quantum readiness
- the scanner.host.os.quantum_ready: OS PQC support
- cipher.negotiated_group: Hybrid PQC groups detected

Asset Tracking

- the scanner.active: Current asset status
- the scanner.last_seen: Last detection timestamp
- Historical tracking: Asset lifecycle management
- Change detection: New and removed assets

Complete Schema Reference

Core Fields (All Events)

Field	Type	Description
@timestamp	String	ISO 8601 timestamp for the event (formatted as "2006-01-02T15:04:05.000Z")
id	String	A unique SHA-1 hash identifying the specific finding
the scanner.type	String	Event type: "cipher", "library", "filesystem", "java_crypto_library", "archivefile", or "connected"
the scanner.scan_mode	String	Scan mode: "local", "remote", or "connected"

the scanner.scan_timestamp	String	ISO 8601 timestamp when the scan was performed
the scanner.scanner_version	String	Version of the certscanner tool that generated this event
the scanner.active	Boolean	True if the asset is currently active/seen (from DB tracking)
the scanner.last_seen	String	ISO 8601 timestamp of when an inactive asset was last seen
the scanner.pqc_vulnerable	Boolean	True if the asset is vulnerable to quantum computing attacks
the scanner.host.machine_serial_number	String	Hardware serial number of the scanning machine
the scanner.host.bios_serial_number	String	BIOS serial number of the scanning machine
the scanner.host.organization	String	Organization name of the scanning system
the scanner.host.domain	String	Domain of the scanning system
the scanner.host.os.quantum_ready	Boolean	Whether the host OS is considered Post-Quantum ready
the scanner.host.os.quantum_ready_when	String	Estimate of when the host OS will no longer be secure (e.g., "now", "2030", "never")
certificate.is_file	String	"true" if certificate found on filesystem, "false" if from network
tags	Array	User-defined tags applied to the scan results

Network Service Fields (cipher/connected types)

Field	Type	Description
server.address	String	The IP address or hostname of the scanned target
server.ip	String	The IP address of the target
server.port	Integer	The port number of the service
service.protocol.type	String	The high-level protocol ("TLS", "SSH")
service.protocol.name	String	The specific protocol version (e.g., "TLSV1.2")

Process & File Fields (library/filesystem types)

Field	Type	Description
process.pid	Integer	Process ID
process.name	String	Process name
process.command_line	String	Full command line of the process
process.owner	String	User that owns the process
process.executable	String	Path to the process executable
file.path	String	Path to the file (executable or library)
file.name	String	Name of the file
file.directory	String	Directory of the file
file.extension	String	File extension
file.size	Integer	File size in bytes
file.created	String	ISO 8601 timestamp of file creation
file.accessed	String	ISO 8601 timestamp of last file access
file.mtime	String	ISO 8601 timestamp of last file modification
file.hash.sha1	String	SHA-1 hash of the file
file.hash.sha256	String	SHA-256 hash of the file
pe.file_version	String	File version from PE header
pe.product_version	String	Product version from PE header
pe.description	String	File description from PE header
pe.company	String	Company name from PE header
pe.product	String	Product name from PE header

TLS & Cipher Fields (cipher type)

Field	Type	Description
tls.cipher	String	IANA name of the cipher suite
tls.cipher_openssl	String	OpenSSL name of the cipher suite
tls.curve	String	The key exchange group/curve used
tls.mac	String	The MAC algorithm used in the cipher suite

tls.version	String	The TLS version number (e.g., "1.2")
tls.version_protocol	String	The protocol name ("TLS")
tls.server.protocol.weight	Integer	A risk score based on the protocol version
tls.server.cipher.weight	Integer	A risk score based on the cipher suite strength
tls.server.signature_hash.weight	Integer	A risk score based on the signature hash
the scanner.cipher.is_preferred	Boolean	True if this is the server's preferred cipher
the scanner.cipher.key_length_bits	Integer	The bit length of the symmetric encryption key
the scanner.cipher.ephemeral_key_length_bits	Integer	The bit length of the ephemeral key
the scanner.cipher.peer_signing_digest	String	The digest used for peer signing
the scanner.cipher.alpn_protocol	String	The negotiated ALPN protocol
the scanner.cipher.session_id	String	The session ID of the TLS session
the scanner.cipher.session_ticket_lifetime_hint_seconds	Integer	The lifetime hint for the session ticket
the scanner.cipher.extended_master_secret_supported	Boolean	True if Extended Master Secret is supported
the scanner.cipher.tls13_early_data_supported	Boolean	True if TLS 1.3 early data is supported
the scanner.cipher.renegotiation_forbidden	Boolean	True if renegotiation is forbidden
the scanner.cipher.compression_method	String	The compression method used

X.509 Certificate Fields (cipher/filesystem types)

Field	Type	Description
x509.version_number	Integer	The X.509 version

x509.serial_number	String	The certificate's serial number
x509.signature_algorithm	String	The algorithm used to sign the certificate
x509.issuer.common_name	String	Issuer's Common Name
x509.issuer.country	String	Issuer's Country
x509.issuer.distinguished_name	String	Issuer's full Distinguished Name
x509.issuer.locality	String	Issuer's Locality
x509.issuer.organization	String	Issuer's Organization
x509.issuer.organizational_unit	String	Issuer's Organizational Unit
x509.issuer.state_or_province	String	Issuer's State or Province
x509.subject.common_name	String	Subject's Common Name
x509.subject.country	String	Subject's Country
x509.subject.distinguished_name	String	Subject's full Distinguished Name
x509.subject.locality	String	Subject's Locality
x509.subject.organization	String	Subject's Organization
x509.subject.organizational_unit	String	Subject's Organizational Unit
x509.subject.state_or_province	String	Subject's State or Province
x509.not_before	String	ISO 8601 timestamp for the start of validity
x509.not_after	String	ISO 8601 timestamp for the end of validity
x509.is_valid	Boolean	True if the certificate is currently valid
x509.public_key_algorithm	String	The public key algorithm
x509.public_key_size	Integer	The bit size of the public key
x509.public_key_curve	String	The curve name for EC keys
x509.key_usage	String	Comma-separated list of key usages
x509.enhanced_key_usage	String	Comma-separated list of extended key usages
x509.is_self_signed	Boolean	True if the certificate is self-signed
x509.hash	String	The SHA-256 fingerprint of the certificate
x509.subject_key_identifier	String	The subject key identifier

Library Fields (library/java_crypto_library types)

Field	Type	Description
library.name	String	The name of the library
library.version	String	The version of the library

library.path	String	The path to the library file or JAR
library.type	String	The type of library (e.g., "java_crypto")
library.crypto_features	String	Comma-separated list of crypto features
library.detection_time	String	ISO 8601 timestamp of when the library was detected
java.vendor	String	The vendor of the Java runtime
java.version	String	The version of the Java runtime
java.manifest	Object	A map of key-value pairs from the JAR's MANIFEST.MF file

Archive Fields (archivefile type)

Field	Type	Description
archive.type	String	Type of archive (e.g., "outlook_pst", "outlook_ost")
archive.encryption.enabled	Boolean	True if the archive is encrypted/password-protected
archive.encryption.type	String	Type of encryption used for the archive
archive.encryption.strength	String	Description of encryption strength
archive.format.version	String	Version of the archive file format

VPN Client Fields (vpn_client type)

Field	Type	Description
vpn.client_name	String	Name of the VPN client software
vpn.vendor	String	VPN vendor/manufacturer
vpn.version	String	VPN client version
vpn.type	String	VPN type (SSL, IPSec, OpenVPN, etc.)
vpn.config_count	Integer	Number of configured VPN profiles
vpn.is_active	Boolean	Whether VPN is currently active
vpn.install_path	String	Installation path of VPN client

vpn.config_path	String	Path to VPN configuration files
vpn.service_name	String	Name of the VPN service
vpn.service_status	String	Status of VPN service (running, stopped)
vpn.last_connection	String	Timestamp of last VPN connection
vpn.protocols_supported	Array	List of supported VPN protocols
vpn.detection_method	String	How the VPN was detected (registry, service, file)

IPSec Tunnel Fields (ipsec_tunnel type)

Field	Type	Description
ipsec.tunnel_name	String	Name of the IPSec tunnel
ipsec.tunnel_type	String	Type (site-to-site, client-to-site)
ipsec.local_endpoint	String	Local endpoint IP address
ipsec.remote_endpoint	String	Remote endpoint IP address
ipsec.local_subnet	String	Local subnet CIDR
ipsec.remote_subnet	String	Remote subnet CIDR
ipsec.authentication_method	String	Authentication method (PSK, Certificate)
ipsec.encryption_algorithm	String	Encryption algorithm used
ipsec.integrity_algorithm	String	Integrity/hash algorithm used
ipsec.dh_group	String	Diffie-Hellman group
ipsec.key_lifetime	Integer	Key lifetime in seconds
ipsec.pfs_enabled	Boolean	Perfect Forward Secrecy enabled
ipsec.status	String	Tunnel status (connected, disconnected)
ipsec.bytes_in	Integer	Bytes received through tunnel
ipsec.bytes_out	Integer	Bytes sent through tunnel
ipsec.last_connected	String	Last connection timestamp
ipsec.pqc_vulnerable	Boolean	Whether tunnel is vulnerable to quantum attacks

Cipher Intelligence Fields

Field	Type	Description
the scanner.cipher.detail.nist_security_category	String	NIST security classification (e.g., "Recommended", "Legacy-Use")

the scanner.cipher.detail.is_quantum_ready	Boolean	Whether the cipher is resistant to quantum attacks
the scanner.cipher.detail.friendly_name	String	Human-readable name for the cipher algorithm
the scanner.cipher.detail.algo.auth	String	Authentication algorithm (e.g., "RSA", "ECDSA")
the scanner.cipher.detail.algo.hash	String	Hash/MAC algorithm (e.g., "SHA256", "AEAD")
the scanner.cipher.detail.algo.vulnerabilities	String	Comma-separated list of known vulnerabilities
the scanner.cipher.detail.security	String	NIST security category (e.g., "Recommended", "Legacy-Use")
the scanner.cipher.detail.overall_risk	String	Overall risk assessment (e.g., "Low", "Medium", "High")
the scanner.cipher.detail.recommendations	String	Security recommendations for the cipher
the scanner.cipher.detail.bit_operator	String	Hexadecimal cipher suite identifier (e.g., "0xC0,0x30")
the scanner.cipher.detail.openssl_name	String	OpenSSL name for the cipher suite

Keystore Fields

Fields present in keystore and keystore_certificate events (local mode only).Note:All the scanner format fields use flat JSON with dot notation - no nested objects.

Field	Type	Description
keystore.type	String	Type of keystore (Windows-CAPI, macOS-Keychain, PKCS12, JKS, etc.)
keystore.accessible	Boolean	Whether the keystore is accessible/readable
keystore.requires_auth	Boolean	Whether keystore requires authentication
keystore.cert_count	Integer	Total number of certificates in keystore
keystore.owner	String	Owner of the keystore file/object
keystore.permissions	String	File system permissions
keystore.error_message	String	Error message if keystore access failed

crypto.key_algorithm	String	Public key algorithm (RSA, ECDSA, etc.)
crypto.key_size	Integer	Key size in bits
crypto.signature_algorithm	String	Signature algorithm used
crypto.fingerprint_sha1	String	SHA-1 fingerprint of certificate
tls.certificate.subject	String	Certificate subject DN
tls.certificate.issuer	String	Certificate issuer DN
tls.certificate.serial_number	String	Certificate serial number
tls.certificate.not_before	String	Certificate valid from timestamp
tls.certificate.not_after	String	Certificate valid until timestamp
tls.certificate.alias	String	Certificate alias/friendly name in keystore
tls.certificate.is_ca	Boolean	Whether certificate is a Certificate Authority
tls.certificate.is_self_signed	Boolean	Whether certificate is self-signed
tls.certificate.has_private_key	Boolean	Whether private key is present in keystore
vulnerability.is_vulnerable	Boolean	Whether certificate has known vulnerabilities
vulnerability.risk_level	String	Risk level (low, medium, high, critical)
vulnerability.risk_reason	String	Reason for vulnerability assessment
pqc.vulnerable	Boolean	Whether certificate is vulnerable to quantum attacks
pqc.reason	String	Reason for post-quantum vulnerability
observer.hostname	String	Hostname of scanning system
observer.name	String	Name of scanning tool
observer.type	String	Type of observer (scanner)
observer.version	String	Version of scanning tool
observer.fips_mode_enabled	Boolean	FIPS 140-2 mode status
observer.bigfix_client_installed	Boolean	Indicates if BigFix client is installed
observer.bigfix_client_id	String	BigFix client ID for asset correlation

Quantum Readiness Assessment Fields

Fields present in quantum_readiness events (local mode only)

Field	Type	Description

quantum.assessment_id	String	Unique identifier for the quantum readiness assessment
quantum.fips_mode_enabled	Boolean	FIPS 140-2 mode status at assessment time
quantum.system_type	String	Classification of system type (workstation, server)
quantum.criticality_level	String	System criticality (critical, important, standard)
quantum.overall_score	Integer	Total quantum readiness score (0-100)
quantum.hardware_score	Integer	Hardware assessment score (0-40)
quantum.hardware_max_score	Integer	Maximum possible hardware score
quantum.os_score	Integer	Operating system score (0-30)
quantum.os_max_score	Integer	Maximum possible OS score
quantum.crypto_score	Integer	Crypto library score (0-25)
quantum.crypto_max_score	Integer	Maximum possible crypto score
quantum.network_score	Integer	Network readiness score (0-5)
quantum.network_max_score	Integer	Maximum possible network score
quantum.readiness_status	String	Readiness status (Ready, Partially Ready, Update Required, Not Ready)
quantum.status_color	String	Status visualization color (green, yellow, orange, red)
quantum.ready_timeline	String	Estimated timeline to quantum readiness
quantum.recommendations	String	Comma-separated actionable recommendations

SIEM Integration

Elasticsearch Integration

BASH:

```
# Direct streaming to Elasticsearch
.\certscanner-windows-amd64.exe -host production-hosts.txt -cipherscan `
  -outputformat tychon `
  -posttoelastic -elasticnode "https://siem.company.com:9200" `
  -elasticapikey "$env:ELASTIC_API_KEY" `
  -elasticindex "crypto-intelligence"
```

```
# Continuous monitoring with asset tracking
.\certscanner-windows-amd64.exe -mode local -scanfilesystem -scanmemory `
  -outputformat tychon -tags "continuous-monitoring" `
  -posttoelastic -elasticnode "$env:ELASTIC_URL"
```

Query Examples

JSON:

```
# Find PQC-vulnerable assets
GET crypto-intelligence/_search
{
  "query": {
    "term": { "tychon.pqc_vulnerable": true }
  }
}

# Find inactive crypto libraries
GET crypto-intelligence/_search
{
  "query": {
    "bool": {
      "must": [
        { "term": { "tychon.type": "library" }},
        { "term": { "tychon.active": false }}
      ]
    }
  }
}

# Find expiring certificates
GET crypto-intelligence/_search
{
  "query": {
    "range": {
      "certificate.not_after": {
        "lte": "now+30d"
      }
    }
  }
}
```

VPN Client Support

Overview

The Quantum Readiness Scanner supports detection of 19 VPN clients across Windows, macOS, and Linux platforms. Each client is assessed for post-quantum cryptography (PQC) readiness and migration status.

Detection Methods

- Package: Package manager detection (Linux only - high confidence)

- Filesystem: Installation directory and executable detection
- Process: Active process and service detection
- Configuration: Client configuration file analysis

PQC Assessment Levels

- Ready: Full PQC algorithm support available
- Partial: Limited PQC support or in development
- Not Ready: No PQC support currently available

Usage

Enable VPN client detection using the-detect-vpn-clientsflag in local scanning mode:

BASH:

```
.\certscanner-windows-amd64.exe -mode local -detect-vpn-clients -output vpn-scan.json
```

VPN Client Support Matrix

VPN Client	Vendor	Windows	macOS	Linux	PQC Readiness	Detection Method
Palo Alto GlobalProtect	Palo Alto Networks	✅	✅	✅	Partial	Filesystem, Process
Cisco AnyConnect	Cisco Systems	✅	✅	✅	Partial	Filesystem, Process, Package
FortiClient	Fortinet Inc.	✅	✅	✅	Not Ready	Filesystem, Process, Package
Check Point Endpoint Security	Check Point Software	✅	✅	✅	Not Ready	Filesystem, Process, Package
SonicWall NetExtender	SonicWall Inc.	✅	✅	✅	Not Ready	Filesystem, Process
Zscaler	Zscaler Inc.	✅	✅	✅	Partial	Filesystem, Process, Package
Cloudflare WARP	Cloudflare Inc.	✅	✅	✅	Ready	Filesystem, Process, Package
NordLayer	Nord Security	✅	✅	✅	Not Ready	Filesystem, Process
Perimeter 81	Perimeter 81 Ltd.	✅	✅	✅	Not Ready	Filesystem, Process
Twingate	Twingate Inc.	✅	✅	✅	Partial	Filesystem, Process
OpenVPN Connect	OpenVPN Technologies	✅	✅	✅	Partial	Filesystem, Process, Configuration, Package
OpenConnect	Open Source	❌	❌	✅	Not Ready	Filesystem, Package

SoftEther VPN	SoftEther Project	✅	✅	✅	Not Ready	Filesystem, Process, Package
WireGuard	WireGuard LLC	✅	✅	✅	Not Ready	Filesystem, Process, Configuration, Package
Pritunl Client	Pritunl Inc.	✅	✅	✅	Not Ready	Filesystem, Process
Viscosity	SparkLabs Pty Ltd.	✅	✅	❌	Not Ready	Filesystem, Process
Tunnelblick	Tunnelblick Project	❌	✅	❌	Not Ready	Filesystem, Process
ProtonVPN	Proton Technologies	✅	✅	✅	Not Ready	Filesystem, Process, Package
ExpressVPN	Express VPN International Ltd.	✅	✅	✅	Not Ready	Filesystem, Process, Package
NordVPN	Nord Security	✅	✅	✅	Partial	Filesystem, Process, Package
Surfshark	Surfshark B.V.	✅	✅	✅	Not Ready	Filesystem, Process, Package
CyberGhost	CyberGhost S.A.	✅	✅	✅	Not Ready	Filesystem, Process, Package
Built-in VPN (Windows)	Microsoft Corporation	✅	❌	❌	Not Ready	Configuration
Built-in VPN (macOS)	Apple Inc.	❌	✅	❌	Not Ready	Configuration
Built-in VPN (Linux)	NetworkManager	❌	❌	✅	Not Ready	Configuration

IPSec Implementation Support

The Quantum Readiness Scanner also detects IPSec tunnel configurations across multiple implementations:

Open Source Implementations

STRONGSWAN

Linux, macOS, Windows support with extensive configuration parsing

LIBRESWAN

Linux and Unix systems

OPENSWAN

Legacy Linux implementation (detected but deprecated)

Built-in OS Implementations

WINDOWS IPSEC
Built-in Windows IPSec policy and configuration

MACOS IPSEC
Native macOS IPSec via System Configuration

LINUX IPSEC
Kernel-level IPSec with XFRM

Detection Examples

Sample VPN Client Detection

JSON:


```
{
  "source_id": "d87c1d880886fd83db018456d742cb83efa0758e",
  "client_name": "Palo Alto GlobalProtect",
  "vendor": "Palo Alto Networks",
  "version": "6.3.2-525",
  "install_path": "/Applications/GlobalProtect.app",
  "status": "active",
  "detection_method": "filesystem",
  "detection_confidence": "high",
  "pqc_assessment": {
    "is_pqc_ready": true,
    "quantum_resistance": "medium",
    "pqc_migration_status": "partial",
    "supported_pqc_algorithms": ["ML-KEM-512"],
    "last_assessed": "2025-09-12T12:54:43.476222-04:00"
  }
}
```


Sample IPSec Tunnel Detection

JSON:


```
{
  "source_id": "90e2352de5c7c9d856327dcfef4ffbd89c2634a1",
  "tunnel_name": "strongSwan Site-to-Site",
  "implementation": "strongswan",
  "config_path": "/etc/ipsec.conf",
  "status": "inactive",
  "tunnel_details": {
    "local_subnet": "192.168.1.0/24",
    "remote_subnet": "10.0.0.0/24",
    "gateway": "203.0.113.1",
    "encryption_algorithms": ["aes256"],
    "integrity_algorithms": ["sha256"],
    "key_exchange_groups": ["modp2048"]
  },
  "pqc_assessment": {
    "is_pqc_ready": false,
    "quantum_resistance": "low",
```

```
      "pqc_migration_status": "not_ready"
    }
  }
```

Current Limitations

Known Limitations

- Security Policy: Detection methods never execute third-party VPN binaries to prevent privilege escalation attacks
- Corporate Compliance: Detection methods avoid PowerShell execution to comply with enterprise security policies
- Version Detection (Linux): Linux version detection relies on package managers and filesystem analysis only, not binary execution
- Process Detection: Client must be running or have been recently active for process-based detection
- PQC Assessment: PQC readiness is based on published vendor documentation and may change with updates
- Linux Variations: Detection supports dpkg, rpm, and pacman package managers across major distributions

Output Features: Split Outputs & Detail Level Control

Overview

The TYCHON PQC Scanner provides two powerful features for controlling output organization and size:

Split Outputs

Break up large scan reports into separate files per dataset type for easier parsing and processing.

- One file per dataset (quantum, network, memory, etc.)
- Maintains complete metadata in each file
- Automatically skips empty datasets
- Works with most output formats

Detail Levels

Control output verbosity to reduce file sizes while maintaining essential security information.

- **Full:** All fields (current behavior)
- **Standard:** ~30-40% size reduction
- **Minimal:** ~60-70% size reduction
- PQC-critical fields always included

Split Outputs Feature

Problem Statement

By default, the scanner outputs all datasets to a single file (e.g., scan_report.json), which can become very large (100MB+) when scanning systems with many certificates, network connections, and running processes. This makes the files:

- Difficult to parse and process
- Slow to load in text editors and analysis tools
- Challenging to integrate with systems expecting specific data types
- Inefficient for streaming analytics (must process entire file)

Solution: Split by Dataset Type

The -split-outputs flag breaks the report into separate files per dataset:

Dataset Name	File Suffix	Contains
quantum	_quantum.json	Quantum readiness assessment with 100-point scoring
network	_network.json	Host/port scan results with TLS certificates
memory	_memory.json	Process memory crypto library detection
filesystem	_filesystem.json	Certificates found on filesystem
keystore	_keystore.json	Keystore scan results (PKCS12, JKS, system stores)
outlook	_outlook.json	Outlook archive files (.pst, .ost)
vpn	_vpn.json	VPN client detection and PQC assessment
ipsec	_ipsec.json	IPSec tunnel configuration and security analysis

Usage Examples

WINDOWS

```
# Basic split outputs
.\certscanner-windows-amd64.exe -local -split-outputs -output report.json

# Split outputs + keep consolidated file
.\certscanner-windows-amd64.exe -local -split-outputs -keep-consolidated -output
report.json
```

LINUX

```
# Basic split outputs
./certscanner-linux-x64 -local -split-outputs -output report.json

# Split outputs + keep consolidated file
./certscanner-linux-x64 -local -split-outputs -keep-consolidated -output
report.json
```

MACOS

```
# Intel Macs
./certscanner-darwin-amd64 -local -split-outputs -output report.json

# Apple Silicon Macs
./certscanner-darwin-arm64 -local -split-outputs -output report.json
```

```
# With consolidated file
./certscanner-darwin-arm64 -local -split-outputs -keep-consolidated -output
report.json
```

Metadata in Split Files

Each split file is self-contained and includes:

- **Complete scanning_system_info**: hostname, OS, IP addresses, BigFix client ID
- **Scan context**: scan_type, target, timestamp, tags
- **New metadata fields**:
  - dataset_type: Identifies which dataset (e.g., "quantum", "network")
  - record_count: Number of records in this file
  - split_mode: true when file is from split output

EMPTY DATASETS

Files are NOT created for empty datasets. For example, if no VPN clients are detected, report_vpn.json will not be created. The scanner logs which files were created and which were skipped.

Detail Levels Feature

Problem Statement

The scanner currently includes ALL fields in output, regardless of their importance or use case. This results in:

- Large file sizes (unnecessary for real-time monitoring)
- Slower ingestion into SIEM and log aggregation systems
- Higher storage costs
- Verbose output for executive dashboards
- Bandwidth constraints in remote scanning scenarios

Solution: Three Detail Levels

The -detail-level flag controls which fields are included in output:

FULL LEVEL (DEFAULT)

Includes: All fields

Size Reduction: 0% (baseline)

Use Cases:

- Forensic analysis
- Deep troubleshooting
- Compliance audits
- Detailed reporting

Command:

-detail-level full

STANDARD LEVEL

Removes:

- Raw binary data (PEM, DER, hex)
- Verbose metadata (file permissions, ownership)
- Internal tracking fields (Active, LastSeen)
- Low-level details (memory addresses, kernel versions)
- Statistical data (byte counts, latency)

Keeps: All security-critical fields, PQC flags, crypto parameters

Size Reduction: ~30-40%

Use Cases:

- SIEM integration
- Security monitoring
- Dashboard ingestion
- Regular scanning

Command:

-detail-level standard

MINIMAL LEVEL

Includes ONLY:

- System/asset identifiers
- Security status indicators
- PQC assessment flags
- Key crypto parameters
- Critical timestamps
- Security scores and ratings
- Top recommendations (limited)

Size Reduction: ~60-70%

Use Cases:

- Real-time alerting
- Executive dashboards
- Bandwidth-constrained environments
- Streaming analytics
- High-frequency scanning

Command:

-detail-level minimal

Field Distribution Statistics

Across all dataset types in the documentation:

Pattern	Field Count	Visualization	Meaning
✓ ✓ ✓	**83 fields**		In ALL three levels (essential security fields)

Pattern	Field Count	Visualization	Meaning
✓ ✓ ✗	**124 fields**		In Full + Standard (useful metadata, not essential)
✓ ✗ ✗	**29 fields**		Full only (verbose/debug data)

Total: 236 fields documented across all dataset types

Size Reduction Benchmarks

Expected size reductions by dataset type when using Standard and Minimal detail levels:

Dataset Type	Full Size	Standard Size	Minimal Size	Standard %	Minimal %
Quantum Readiness	8-12 KB	5-8 KB	2-3 KB	-35%	-70%
Network (per host)	50-200 KB	30-120 KB	15-50 KB	-35%	-70%
Process Memory	5-20 KB	3-14 KB	1-5 KB	-30%	-70%
Filesystem Certs	3-8 KB	2-5 KB	1-2 KB	-35%	-70%
Keystores	5-30 KB	3-20 KB	1-8 KB	-35%	-75%
Outlook Archives	1-3 KB	0.8-2 KB	0.4-0.8 KB	-30%	-65%
VPN Clients	4-15 KB	2-9 KB	1-3 KB	-35%	-75%
IPSec Tunnels	3-10 KB	2-6 KB	1-3 KB	-35%	-75%

Combined Usage: Split + Detail Levels

Both features can be used together for maximum flexibility and efficiency:

Example: Real-Time Security Monitoring

WINDOWS

```
# Maximum efficiency: Split outputs + minimal detail
.\certscanner-windows-amd64.exe -local -split-outputs -detail-level minimal -
output report.json

# Result: 8 small files instead of 1 large file
# - report_quantum.json    (2-3 KB instead of 12 KB)
# - report_network.json    (15-50 KB instead of 200 KB)
# - report_memory.json     (1-5 KB instead of 20 KB)
# - report_filesystem.json (1-2 KB per cert)
# - ... etc
```

LINUX

```
# Maximum efficiency: Split outputs + minimal detail
./certscanner-linux-x64 -local -split-outputs -detail-level minimal -output
report.json

# Result: 8 small files instead of 1 large file
# - report_quantum.json    (2-3 KB instead of 12 KB)
# - report_network.json    (15-50 KB instead of 200 KB)
# - report_memory.json     (1-5 KB instead of 20 KB)
# - report_filesystem.json (1-2 KB per cert)
# - ... etc
```

MACOS

```
# Maximum efficiency: Split outputs + minimal detail
./certscanner-darwin-arm64 -local -split-outputs -detail-level minimal -output
report.json

# Result: 8 small files instead of 1 large file
# - report_quantum.json    (2-3 KB instead of 12 KB)
# - report_network.json    (15-50 KB instead of 200 KB)
# - report_memory.json     (1-5 KB instead of 20 KB)
# - report_filesystem.json (1-2 KB per cert)
# - ... etc
```

Benefits:

- Each dataset in its own file (easy parsing)
- ~60-70% size reduction per file
- Fast ingestion into SIEM systems
- Reduced network bandwidth
- Lower storage costs

Example: Comprehensive Audit with Organization

```
# Split outputs with full detail for forensic analysis
./certscanner -local -split-outputs -detail-level full -output audit_report.json

# Keep consolidated file for compliance
./certscanner -local -split-outputs -keep-consolidated -detail-level full -
output audit_report.json
```

BENEFITS:

- All fields retained (full detail)
- Organized by dataset type (easy to find specific data)
- Optional consolidated file for compliance requirements
- Suitable for deep analysis and troubleshooting

Format and Integration Compatibility

Output Format Support

Format	Split Outputs	Detail Levels	Notes
json	✓ Full	✓ Full	8 separate JSON files when split
flatndjson	✓ Full	✓ Full	8 separate NDJSON files when split
tychon	✓ Full	✓ Full	8 separate NDJSON files when split
cbom	✓ Full	✓ Full	8 separate CBOM files when split
html	⚠ Partial	✓ Full	Creates single consolidated HTML only
eventlog	✗ Not Compatible	✓ Full	Writes to system logs, not files

Integration Platform Support

Platform	Split Outputs	Detail Levels	Notes
Elasticsearch	✓	✓	Posts to separate indices per dataset
Kafka	✓	✓	Posts to topic-per-dataset pattern
Splunk	✓	✓	Uses different sourcetypes per dataset
S3 Upload	✓	✓	Uploads all split files maintaining folder structure

Use Case Recommendations

Real-Time Security Monitoring

-split-outputs -detail-level minimal

Maximum efficiency for continuous monitoring. Small files, fast processing, essential security data only.

SIEM Integration

-split-outputs -detail-level standard

Organized datasets with all security-relevant fields. Good balance of detail and size.

Compliance Audits

-split-outputs -keep-consolidated -detail-level full

Complete records organized by type. Consolidated file for archival requirements.

Executive Dashboards

-detail-level minimal

Single file with high-level security indicators, scores, and PQC readiness flags.

Forensic Investigation

-split-outputs -detail-level full

All fields retained, organized for analysis. Access raw certificates, memory addresses, full metadata.

Bandwidth-Constrained Remote Scanning

-detail-level minimal

Minimize data transfer. Essential fields only, ~60-70% smaller outputs.

Command-Line Reference

-split-outputs

Type: Boolean flag

Default: false
Description: Enable split output files (one per dataset type)

-keep-consolidated
Type: Boolean flag
Default: false
Requires: -split-outputs
Description: Keep consolidated file when using split outputs (creates both split files AND the main file)

-detail-level <level>
Type: String
Default: full
Valid Values: full, standard, minimal
Description: Control output verbosity (full = 0% reduction, standard = ~30-40% reduction, minimal = ~60-70% reduction)

Appendix B: Network Security

Security Architecture Overview

Important Security Notice

The scanner implements context-aware network security measures. SSRF protection applies only to integration service configurations (Kafka, Elasticsearch, Splunk, S3) while allowing full network access for legitimate security scanning via-hostflags.

Protected Against

- SSRF Attacks: Server-Side Request Forgery prevention
- Port Scanning: Blocked access to dangerous internal ports
- Internal Network Access: Protected localhost and private IP ranges
- DNS Rebinding: Hostname resolution validation
- URL Injection: Malformed URL and encoding attacks
- Credential Exposure: Secure logging with pattern redaction

Legitimate Use Cases

- Integration Services: Kafka, Elasticsearch, Splunk on localhost (with SSRF protection)
- S3-Compatible Storage: MinIO and other local object storage (with SSRF protection)
- Security Scanning: Any target via-host(no restrictions)
- Certificate Discovery: Internal and external TLS endpoints (no restrictions)
- Network Assessment: Private networks, localhost, cloud metadata (no restrictions)
- Vulnerability Testing: All ports and protocols for scan targets (no restrictions)

SSRF Protection Framework

Context-Aware Validation

The scanner uses a sophisticated context-aware validation system that applies different security rules based on how network inputs are being used:

Validation Contexts

Service Configuration Examples

BASH:

```
# Integration services - PROTECTED by SSRF validation
-elasticnode "http://localhost:9200"     # Internal Elasticsearch - ALLOWED
-kafkabrokers "localhost:9092"           # Internal Kafka - ALLOWED
-splunkurl "http://10.0.0.15:8088"       # Internal Splunk - ALLOWED
-s3endpoint "http://localhost:9000"      # MinIO local storage - ALLOWED

# Integration services - BLOCKED by SSRF protection
-elasticnode "http://169.254.169.254/"   # AWS metadata - BLOCKED
-kafkabrokers "127.0.0.1:6379"          # Redis on non-Kafka port - BLOCKED
-splunkurl "http://localhost:22"         # SSH port - BLOCKED

# Scan targets - NO SSRF restrictions (legitimate security scanning)
-host "localhost:8080"                   # Local web app - ALLOWED for scanning
-host "192.168.1.100:443"               # Internal server - ALLOWED for scanning
```

```
-host "10.0.0.50:22"                    # SSH service - ALLOWED for scanning
-host "169.254.169.254:80"               # Even metadata service - ALLOWED for
scanning
```

Blocked IP Addresses & Ranges

For Integration Service URLs Only

These IP ranges are blocked only when configuringintegration services(Kafka, Elasticsearch, Splunk, S3). Scan targets using-hostcan access any IP range for legitimate security assessments.

IPv4 Address Ranges

Loopback Addresses

Range	Description
127.0.0.0/8	Localhost loopback
127.0.0.1	Primary localhost
127.1	Short form localhost
2130706433	Decimal form of 127.0.0.1
017700000001	Octal form of 127.0.0.1
0x7f000001	Hex form of 127.0.0.1

Private Network Ranges

Range	Description
10.0.0.0/8	Private Class A
172.16.0.0/12	Private Class B
192.168.0.0/16	Private Class C
169.254.0.0/16	Link-local APIPA

Special Use Ranges

Range	Description
0.0.0.0/8	"This network"
224.0.0.0/4	Multicast addresses
240.0.0.0/4	Reserved/Experimental

IPv6 Address Ranges

Loopback & Local

Range	Description
::1	IPv6 loopback
[::1]	Bracketed IPv6 loopback
fe80::/10	Link-local addresses

Private & Special

Range	Description
fc00::/7	Unique local addresses
fd00::/8	Unique local addresses
::ffff:0:0/96	IPv4-mapped IPv6
ff00::/8	Multicast addresses

Blocked Ports for Integration Services

Port Restrictions Apply to Integration Services Only

These ports are blocked only when configuring integration services (Kafka, Elasticsearch, Splunk, S3) to prevent SSRF attacks. Scan targets using-hostcan access any port for legitimate security scanning.

Common Service Ports

Port	Service
22	SSH
23	Telnet
25	SMTP
53	DNS
110	POP3
143	IMAP
993	IMAPS
995	POP3S

Database Ports

Port	Database
3306	MySQL
5432	PostgreSQL
6379	Redis
11211	Memcached
27017	MongoDB

Internal Web Ports

Port	Service
8080	HTTP Alt
8443	HTTPS Alt
9200	Elasticsearch HTTP
9300	Elasticsearch Transport

Note: Internal services like Elasticsearch on localhost:9200 or Kafka on localhost:9092 can use these ports without restriction because they use the ContextInternalService validation context. Scan targets using-hostcan always access these ports for legitimate security assessments.

Security Best Practices

Network Security

Safe Practices

- Use HTTPS for all external communications
- Configure internal services with proper authentication
- Use network segmentation for sensitive systems
- Regularly update TLS certificates
- Monitor network access logs

Avoid These

- Exposing internal services to the internet
- Using weak or default credentials
- Disabling SSL certificate verification
- Storing credentials in plain text
- Bypassing security validations

Configuration Security

Secure Configuration

BASH:

```
# Use encrypted credential storage
certscanner -config

# Configure with secure endpoints
certscanner -config-elasticnode "https://elastic.internal:9200"
certscanner -config-kafkabrokers "kafka1.internal:9092"
certscanner -config-splunkurl "https://splunk.company.com:8088"
```

Environment Variables

BASH:

```
# Set via environment variables
export ELASTIC_NODE="https://localhost:9200"
export KAFKA_BROKERS="localhost:9092"
export SPLUNK_URL="http://splunk.internal:8088"
export SPLUNK_TOKEN="your-secure-token-here"
```

Advanced Security Features

Secure Logging

The scanner includes intelligent credential detection and redaction in logs:

Redacted Patterns

- password=***REDACTED***
- apikey=***REDACTED***
- token=***REDACTED***
- Certificate data (PEM blocks)
- Environment variables with sensitive names

File Security

Enhanced file permission security for temporary files and outputs:

Secure Permissions

- Temporary files:0600(owner read/write only)
- Output files:0600(owner read/write only)
- Directories:0700(owner access only)
- Configuration files:0600

Cross-Platform Security

Windows

- Native Windows API usage
- No PowerShell dependencies
- Registry-based detection
- NTFS permissions

Linux

- Filesystem permissions
- Process isolation
- SELinux compatibility
- Container support

macOS

- Keychain integration
- Application bundle detection
- System Integrity Protection
- Gatekeeper compatibility

Security Troubleshooting

Common Issues & Solutions

Problem: Getting connection errors when trying to configure Kafka on localhost or private IP.
Solution: Ensure you're using the correct flag for internal service configuration:

```
certscanner -posttokafka -kafkabrokers "localhost:9092"
```

This uses ContextInternalService validation which allows localhost and private IPs.

Problem: Cannot configure local Elasticsearch instance.
Solution: Use proper internal service configuration:

```
certscanner -posttoelastic -elasticnode "http://localhost:9200"
```

The ContextInternalService validation allows access to blocked ports for configured services.

Problem: Cannot connect to IPv6 internal services.
Solution: IPv6 localhost and private addresses are supported for internal services:

```
certscanner -kafkabrokers "[::1]:9092"
certscanner -elasticnode "http://[fd00::100]:9200"
```

Use bracket notation for IPv6 addresses in URLs.

Getting Help

If you encounter security-related issues or have questions about network configuration, check the main documentation or contact your system administrator for guidance on enterprise security policies.

Appendix C: Additional Security Considerations

Scanning Considerations

- Memory scanning requires elevated privileges
- Network scanning may trigger security monitoring
- Large CIDR ranges generate significant traffic
- Cipher enumeration involves multiple TLS handshakes

Data & Privacy

- Cloud downloads only when explicitly enabled
- Web service integration transmits scan results
- Filesystem scanning requires read access
- Database tracking stores asset history locally

Antivirus Configurations

Enterprise antivirus solutions may flag the scanner components due to their network scanning capabilities and cryptographic analysis functions. Configure your antivirus policies to trust the following files and certificates to prevent false positives and ensure proper operation.

Scanner Executables

Add these file patterns to antivirus exclusions:

```
Windows:
• cryptographic-analyzer-windows-*-*.exe
• certscanner-windows-*.exe
• certscanner.exe
Linux:
• cryptographic-analyzer-linux-*-*
• certscanner-linux-*
• certscanner
macOS:
• cryptographic-analyzer-darwin-*-*
• certscanner-darwin-*
```

OpenSSL Components

Bundled OpenSSL 3.5.4 binaries extracted to temporary directories:
The scanner extracts OpenSSL to a temporary directory pattern. Add these paths to antivirus exclusions:

```
Windows:
• %TEMP%\certscanner_openssl_*\openssl.exe
SHA256: 61857d55998dfaf7e855c00c3afbf9e2b7e024f44cd854180da330b0623c1635
Linux:
• /tmp/certscanner_openssl_*/openssl
SHA256: 89677c4272fac1aead18dd65143dc3c0bc13f0c1c22f19a538be59db0a30d2e8
macOS (Intel):
• /tmp/certscanner_openssl_*/openssl
SHA256: 8702d163004c63d84c0d589d11291f23ac230ad2495f3243a26e720373a6fbb5
macOS (Apple Silicon):
• /tmp/certscanner_openssl_*/openssl
SHA256: 9fb92ca4f707b7d452b2ca8dd2a293c3cb0789e47d80f1acdda06c0042a59f73
```

Note: The * represents a random suffix generated by the operating system for each execution.

Code Signing Certificates

Scanner executables are signed with the following certificates. Configure your antivirus to trust these certificate authorities and signatures:

Windows Code Signing Certificate

```
Subject:
CN=TYCHON, LLC
OU=Engineering
O=TYCHON, LLC
L=Fredericksburg
S=Virginia
C=US
Certificate Details:
Serial: 068E0290AC164B8C1151C071B7CAE96A
SHA-256 Thumbprint:
D24DCF9372E73CAE4D93B8D34E43E3E24BDADC83
Issuer:
DigiCert Trusted G4 Code Signing RSA4096 SHA384 2021 CA1
Valid: Sep 25, 2023 - Nov 10, 2026
```

macOS Code Signing Certificate

```
Developer ID:
Developer ID Application: Tychon, LLC
Team Identifier:
XSTMP7MDL9
Certificate Chain:
Developer ID Certification Authority
Apple Root CA
CDHash (SHA-256):
f1254bc1837276097dfa758325abb5e92a5bb698
Full SHA-256:
f1254bc1837276097dfa758325abb5e92a5bb69873260b37cf4a19fd64cb9614
Runtime:
Hardened Runtime Enabled (v12.0.0)
```

Common Antivirus Platform Configuration

Windows Defender:

- Add folder exclusions for the scanner installation directory
- Configure process exclusions for certscanner*.exe
- Allow certificate-based trust for signed binaries

CrowdStrike Falcon:

- Create IOA exclusions for the scanner processes
- Add certificate-based allow policies
- Configure custom hash-based exclusions

Symantec Endpoint:

- Add application control exceptions
- Configure file exclusions in real-time scan
- Trust publisher certificates in policy

McAfee/Trellix:

- Configure VirusScan exclusions
- Add DLP policy exceptions for scan output
- Set application control trusted publishers

Appendix D: CPU Throttling Guidance

CPU Throttling Overview

The-cputhrottle flag provides comprehensive control over CPU usage across all resource-intensive operations in the Quantum Readiness Scanner. This feature prevents system overload, ensures smooth operation alongside other applications, and provides predictable performance characteristics.

Key Benefits

- Prevents system resource exhaustion
- Reduces memory consumption by 40-70%
- Allows concurrent workload execution
- Adaptive throttling based on system load

Real-World Performance Metrics

Based on testing with a 14-core system scanning 516 cipher suites and 2,300+ filesystem certificates:

Metric	None	Low	Medium	High
CPU Usage	100%	50%	70%	90%
Memory Usage	~5.2 MB	~1.3 MB	~2.6 MB	~2.6 MB
Scan Time (relative)	1.0x	4.0x	2.0x	1.5x
Concurrent Operations	28	7	14	14
Filesystem Workers	56	14	28	28
System Responsiveness	Fair	Excellent	Good	Fair

Memory Efficiency Gains

Without Throttling (None):

- Peak Memory: 5.2 MB
- Buffer Allocations: 516+ per operation
- Goroutines: 28-56 concurrent
- Risk: Can reach GB-scale with large CIDR blocks

With Throttling (Low):

- Peak Memory: 1.3 MB(-75%)
- Buffer Allocations: 100 max (capped)
- Goroutines: 7-14 concurrent
- Risk: Protected against memory exhaustion

Operations Affected by CPU Throttling

CPU throttling comprehensively controls resource usage across all scanner operations:
Cryptographic Operations

- Cipher Suite: Enumeration TLS/SSL cipher testing and negotiation
- Certificate Analysis: X.509 certificate parsing and validation
- SSH Key Exchange: SSH protocol negotiation and testing

Filesystem Operations

- Certificate Discovery: PEM, DER, CRT, CER file scanning
- Outlook Archive Scanning: PST, OST, PAB file analysis
- Trust Store Analysis: System and Java keystore inspection

Network Operations

- Port Scanning: Local and remote port discovery
- Host Discovery: Multi-host parallel scanning
- DNS Enumeration: Subdomain discovery and resolution

System Operations

- Memory Scanning: Process crypto library detection
- VPN Client Detection: 30+ VPN client identification
- IPSec Tunnel Analysis: IKE/IPSec configuration parsing

Recommended Use Cases

Production Server Scanning

Why Low: Minimizes impact on production services, uses only 50% CPU, allows normal server operations to continue unimpeded.

Developer Workstation

Why Medium: Default setting provides good balance, allows concurrent development work while scanning completes reasonably quickly.

Overnight Security Audit

Why High: Maximizes scanning speed during off-hours, completes comprehensive audits faster while system is idle.

CI/CD Pipeline Integration

Why Low: Ensures pipeline stability, prevents build agent resource exhaustion, predictable execution time.

Dedicated Security Scanner

Why None: Maximizes throughput on dedicated hardware, completes large-scale assessments in minimum time.

Best Practices & Recommendations

Do's

- Start with Medium (default): Test and adjust based on system response
- Use Low for production systems: Prioritize stability over speed
- Monitor with -logfile: Track performance metrics and throttling effects
- Consider scan scope: Large CIDR blocks need more aggressive throttling
- Use High for idle periods: Maximize efficiency during maintenance windows

Don'ts

- Avoid None on shared systems: Can cause severe performance degradation
- Don't scan /8 networks without throttling: 16M+ IPs can exhaust memory
- Don't ignore memory warnings: High memory alerts indicate throttling needed
- Avoid frequent throttle changes: Let adaptive throttling stabilize
- Don't disable for "speed": Often counterproductive due to resource contention

Critical Warning: Large Network Scans

Scanning large CIDR blocks (e.g., /8, /16) without proper throttling can cause:

- Memory exhaustion (GB-scale allocations)

- System crashes (especially on Windows)
- Network infrastructure overload
- Security monitoring alerts

Always use -cputhrottle low or medium for large-scale scans!