CA Certificate Injection via SAN Parsing in OpenXPKI

Security Advisory	
Advisory ID:	WRS-SA-2026-001
Severity:	critical
CVE:	pending
Affected:	OpenXPKI Enterprise Edition and Community Edition through 3.32.15 (fixed in 3.32.16)
Published:	2026-04-23
Status:	Draft v1.0

Executive Summary

A critical vulnerability in the certificate enrollment process of **OpenXPKI Enterprise Edition** and **OpenXPKI Community Edition** allows an attacker to inject arbitrary OpenSSL configuration directives through malformed Subject Alternative Name (SAN) values. Successful exploitation produces a certificate with `basicConstraints: CA:TRUE`, granting the attacker a fully trusted rogue CA certificate within the PKI. The injection technique is not limited to `basicConstraints` and may allow insertion of further arbitrary X.509 extensions, including Key Usage and Extended Key Usage.

The vulnerability constitutes a **trust boundary bypass**: OpenXPKI correctly rejects CSRs that explicitly request `CA:TRUE` in their X.509 extension block, but this security control is circumvented when the same constraint is injected through SAN values that reach the internal certificate-signing code without being sanitised. Any SAN type accepted by an enrollment profile is a potential injection vector if that SAN type is not subject to input filtering before it reaches the vulnerable code. In the default configuration, DNS SAN values are filtered and cannot be used for injection; IP address and URI SAN values carry no such filtering and are exploitable whenever the relevant SAN type is permitted by the active profile.

The most accessible attack path is available in a default **OpenXPKI Enterprise Edition** installation and uses the **OpenXPKI RPC API**. The default EE configuration exposes an RPC endpoint that accepts IP address SANs and does not require authentication. An attacker submits a crafted JSON API request — no PKCS#10 CSR construction is required. Under the default configuration, anonymous submissions enter a work queue for manual

RA Operator approval. The standard RA review interface does not prominently surface raw SAN values, making it likely that a diligent but uninformed operator would approve the request without recognising the injection payload.

Both the EE and CE are also exploitable via malicious PKCS#10 CSRs submitted through any supported enrollment protocol (SCEP, EST, and RPC), provided an unfiltered SAN type is accepted by the active enrollment profile.

All versions of OpenXPKI EE and CE up to and including 3.32.15 are affected. A hotfix script covering versions 3.30 through 3.32.15 is available immediately from White Rabbit Security. Upgrading to version 3.32.16 or later provides a permanent fix.

In all customer environments managed by White Rabbit Security, Issuing CAs have been configured with `pathLen:0`, which limits the usability of any rogue CA certificate issued via this vulnerability and substantially reduces the practical impact.

> **Note:** This advisory is issued in draft status pending assignment of a CVE identifier. The technical details and remediation information are complete.

Affected Systems

Vulnerable Versions

Product	Versions	Status
OpenXPKI Enterprise Edition (EE)	All versions through 3.32.15	**Vulnerable**
OpenXPKI Community Edition (CE)	All versions through 3.32.15	**Vulnerable**
OpenXPKI CE / EE	3.32.16 and later	Fixed

Versions of OpenXPKI prior to 3.30 are also vulnerable but are not covered by the hotfix script. Those versions are end-of-life; upgrading to 3.32.16 or later is the only remediation available.

Affected Enrollment Interfaces

Protocol	Affected	Condition for exploitation
RPC API — JSON parameters	Conditional	EE default configuration is vulnerable; CE requires a non-default profile permitting IP or URI SANs
RPC API — PKCS#10 CSR	Yes	Any profile accepting an unfiltered SAN type; IP SANs require a malformed CSR crafted with custom ASN.1 tooling
SCEP (Simple Certificate Enrollment Protocol)	Yes	Any profile accepting an unfiltered SAN type; IP SANs require a malformed CSR crafted with custom ASN.1 tooling
EST (Enrollment over Secure Transport, RFC 7030)	Yes	Any profile accepting an unfiltered SAN type; IP SANs require a malformed CSR crafted with custom ASN.1 tooling
ACME (Automatic Certificate Management Environment)	**No**	Input data is properly sanitised in both the CE and EE default configurations
Management GUI (web interface)	**No**	—

An "unfiltered SAN type" is one whose value passes through to the certificate signing code without normalisation or character stripping that would remove the newline characters required for injection. In the default configuration, DNS SANs are filtered and cannot be exploited; IP address and URI SANs are unfiltered.

OpenXPKI Community Edition — Default Configuration

The default CE configuration exposes SCEP, EST, and RPC endpoints. The RPC endpoint's default `certificate_enroll` profile does **not** accept IP address SANs as plain JSON parameters, so the JSON-parameter RPC attack is not available on a default CE installation.

However, all CE endpoints accept PKCS#10 CSRs. A PKCS#10 CSR containing a malicious SAN extension, e. g. malformed IP address SAN (crafted with custom ASN.1 tooling) or a URI SAN (if enabled in the profile) will trigger the vulnerability on any of these endpoints.

Submissions are anonymous and require manual RA Operator approval before a certificate is

issued. The standard RA review interface does not prominently surface raw SAN field values, making it likely that an operator would approve a malicious request without recognising the injection payload.

OpenXPKI Enterprise Edition — Default Configuration

The default EE configuration exposes SCEP, EST, and RPC endpoints. Unlike the CE, the EE's default RPC `certificate_enroll` profile **does** accept IP address SANs as plain JSON parameters, making the JSON-parameter attack available without any non-default configuration. An attacker only needs to submit a crafted HTTP POST — no PKCS#10 CSR construction is required.

Additionally, all EE endpoints accept PKCS#10 CSRs, and the malformed-CSR attack paths available to CE installations apply equally to EE installations.

As with CE, anonymous submissions require manual RA Operator approval under the default configuration, and the standard review interface does not make the injection payload readily apparent.

Determining Your Version

On the OpenXPKI server host, run:

```
openxpkictl version
```

Alternatively, query the installed package:

```
# OpenXPKI Community Edition (Debian / Ubuntu)
dpkg -l libopenxpki-perl | grep 'ii'

# OpenXPKI Enterprise Edition (Red Hat Enterprise Linux, SUSE SLES)
rpm -q myperl-openxpki-core

# OpenXPKI Enterprise Edition (Ubuntu LTS)
dpkg -l libopenxpki-myperl | grep 'ii'
```

Technical Details

Vulnerability Description

OpenXPKI generates an OpenSSL configuration file on-the-fly for each certificate signing operation. The affected code in `OpenXPKI::Crypto::Backend::OpenSSL::Config`

writes Subject Alternative Name (SAN) values directly into this configuration file using an unparameterized `sprintf` call, without first sanitizing values for characters that carry special meaning in the OpenSSL configuration file format:

```
# OpenXPKI/Crypto/Backend/OpenSSL/Config.pm, line 641
sprintf '%s.%01d = "%s"', $entry->[0], $sectidx, $entry->[1]
```

If the SAN value extracted from the submitted CSR contains a newline character (`\n`) followed by an OpenSSL config section header and directives, those bytes are written verbatim into the generated `.cnf` file. OpenSSL processes the file sequentially and, when a section name appears more than once, the **last** definition wins (LHASH last-writer-wins semantics). An attacker exploits this to append a second definition of the `[v3ca]` extensions section that overrides the certificate profile's `CA:FALSE` with `CA:TRUE`.

Trust Boundary Bypass

OpenXPKI implements a security control that correctly rejects CSRs whose X.509 extension block explicitly requests `CA:TRUE`. This control functions as intended and was verified during the investigation. However, the same restriction is not applied to SAN field values before they are written into the dynamically generated OpenSSL configuration file. The vulnerability therefore bypasses a working, tested security check by taking an unguarded alternate input path — the classic definition of a trust boundary violation.

Attack Mechanism

Any SAN value that reaches `Config.pm` without filtering is a viable injection carrier. For PKCS#10 CSRs this is most straightforwardly achieved with a URI SAN (an IA5String that can hold arbitrary text) or a deliberately malformed IP address SAN. For the RPC JSON interface the payload is submitted as a plain text parameter. The example below uses a URI SAN value (shown with visible `\n` for readability):

```
legit.example.com"\n
[v3ca]
basicConstraints = critical,CA:true
#
```

After `sprintf` expansion, the generated OpenSSL configuration file contains:

```
subjectAltName.0 = "legit.example.com"
[v3ca]
basicConstraints = critical,CA:true
#"
```

The injected `[v3ca]` section re-opens the extensions section that the legitimate certificate profile already defined with `basicConstraints = CA:FALSE`. The closing `"` from `sprintf`'s format string becomes part of a comment (after `#`) and does not cause a parse error. Because OpenSSL's LHASH processes sections in file order and uses the last value seen, `CA:TRUE` takes precedence and is written into the issued certificate.

Attack Vectors

Three distinct exploitation paths exist, in increasing order of required attacker capability.

Vector 1 — OpenXPKI RPC API with IP address SAN (EE default configuration)

The OpenXPKI RPC interface accepts certificate enrollment requests as JSON parameters. SAN values provided as JSON text are passed through the same unsanitized code path as CSR-derived SAN values. The default OpenXPKI EE configuration exposes an RPC endpoint that accepts IP address SANs. Because the JSON parameters are plain text, no PKCS#10 CSR construction and no special tooling are required — the attacker submits a standard HTTP POST with a crafted JSON body containing the injection payload as the IP SAN value.

Anonymous RPC submissions are accepted but held for manual RA Operator review before a certificate is issued. The standard RA review interface does not prominently expose raw SAN field content; reviewing it requires the operator to open the workflow instance and explicitly display the request context. An operator following a routine approval workflow is therefore likely to approve the request without recognising the malicious payload.

Environments where the RPC enrollment workflow is configured with automatic approval (no RA review required) are exploitable with no human interaction and represent the most severe scenario.

Vector 2 — URI SAN injection via PKCS#10 CSR

URI SANs are a supported but uncommon SAN type. Because a URI SAN is encoded as an IA5String in ASN.1, it can carry arbitrary text including the injection payload. If the enrollment profile for an active RPC, SCEP, or EST endpoint is configured to permit URI SANs, an attacker can craft a standards-conformant PKCS#10 CSR using off-the-shelf cryptographic libraries (e.g., Python `cryptography`, Java BouncyCastle) with the injection

payload as the URI SAN value. URI SANs are not enabled by default in standard OpenXPKI profile templates.

Vector 3 — Malformed IP address SAN via PKCS#10 CSR

An IP address SAN is ASN.1-encoded as a fixed-width binary octet string (4 bytes for IPv4, 16 bytes for IPv6). A standards-conformant PKCS#10 CSR cannot carry a text payload in an IP SAN field. However, using custom ASN.1-encoding tooling to produce a deliberately malformed IP SAN extension, an attacker can embed the injection payload in that field. This vector requires non-standard tooling to produce the malformed CSR, but works against any RPC, SCEP, or EST endpoint whose profile accepts IP address SANs — including the default CE and EE configurations.

> **Note on DNS SANs:** The vulnerability exists in principle for every SAN type that reaches `Config.pm` without filtering. Whether a given SAN type is exploitable depends on whether input filtering is applied at the configuration layer before the value is written to the OpenSSL config file. In the default configuration, DNS SAN values are processed through a dedicated template filter that normalises them in a way that prevents successful injection. IP address and URI SAN values carry no equivalent filter. Operators who have removed or customised the DNS SAN filter should treat DNS SANs as a further potential injection vector.

Impact

The primary demonstrated impact is the issuance of a certificate with `basicConstraints: CA:TRUE`, producing a rogue CA certificate that is fully trusted by any relying party within the PKI. Because the injection mechanism allows arbitrary OpenSSL configuration directives to be appended to the certificate extensions section, the technique is not limited to `basicConstraints`. It may also allow an attacker to inject additional X.509 extensions, including Key Usage and Extended Key Usage extensions with attacker- chosen values, and possibly custom extensions. Full enumeration of injectable extensions has not been completed; this assessment is indicative of the scope of the injection primitive.

CVSS Score

The score below reflects the most accessible default-configuration attack path (Vector 1: anonymous RPC submission, manual RA approval required).

Metric	Value
CVSS v3.1 Base Score	**9.3 (Critical)**
Attack Vector	Network
Attack Complexity	Low
Privileges Required	None
User Interaction	Required
Scope	Changed
Confidentiality	High
Integrity	High
Availability	None

CVSS v3.1 vector: `CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:H/A:N`

> **Elevated score for auto-approval configurations:** Environments where the enrollment workflow is configured with automatic approval (no RA review) have User Interaction: None, yielding a base score of **10.0 (Critical)**: `CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:N`.

Attack Complexity is rated Low because the default OpenXPKI EE configuration is vulnerable via Vector 1 without any non-default prerequisites. User Interaction is Required because the default configuration requires an RA Operator to approve the anonymous submission. The Scope metric is Changed because a successfully issued rogue CA certificate extends the attacker's reach to any relying party that trusts the PKI.

Remediation

Recommended Fix: Upgrade to OpenXPKI 3.32.16

The vulnerability is resolved in OpenXPKI CE and EE version 3.32.16 by properly sanitizing SAN values before incorporating them into the dynamically generated OpenSSL configuration file. Upgrade to version 3.32.16 or any later version as soon as the release becomes available.

Immediate Fix: Apply the Hotfix Script

For installations running OpenXPKI 3.30 through 3.32.15 that cannot be upgraded immediately, White Rabbit Security provides a hotfix shell script. The script:

1. Locates `OpenXPKI/Crypto/Backend/OpenSSL/Config.pm` under the active Perl library path
2. Verifies the SHA-256 hash of the installed file against known-vulnerable values
3. Creates a timestamped backup of the original file
4. Replaces the vulnerable file with a corrected version embedded in the script as a here-document
5. Verifies the SHA-256 hash of the replacement before and after installation

Contact security@whiterabbitsecurity.com to obtain the hotfix script.

Applying the Patch

Copy the hotfix script `WRS-SA-2026-001-patch-openxpki-config-sh` to the OpenXPKI server host.

If the script is executed without additional options, the script performs a read-only test and verifies if the system is vulnerable.

```
# WRS-SA-2026-001-patch-openxpki-config.sh
[INFO]  Searching for 'OpenXPKI/Crypto/Backend/OpenSSL/Config.pm' under '/usr/share/perl5 /opt/myperl' ...
[INFO]  Found: /opt/myperl/lib/vendor_perl/5.40.1/OpenXPKI/Crypto/Backend/OpenSSL/Config.pm
[INFO]  SHA256 (current): b3424aa1286bd9c2401df88000eae127d4c9b90d291ac95086a2cedc21aea300
[INFO]  Detected source version: v3.32
[INFO]  Patch required for version v3.32. Rerun script with --apply to apply.
```

If the system is vulnerable, run the hotfix script again as root on the OpenXPKI server host and specify the `--apply` option:

```
sudo bash WRS-SA-2026-001-patch-openxpki-config.sh --apply
```

After successful patch application, restart the OpenXPKI server process so that the corrected module is loaded:

```
openxpkictl restart server
```

> **Important:** Reinstalling or upgrading vulnerable OpenXPKI packages will overwrite the patched file and restore the vulnerable version. The hotfix must be re-applied after any such package operation until the installation is upgraded to OpenXPKI 3.32.16 or later.

Expected Script Output

Case: patch not yet applied (first run on a vulnerable system)

```
# sudo bash WRS-SA-2026-001-patch-openxpki-config.sh --apply
[INFO]  Searching for 'OpenXPKI/Crypto/Backend/OpenSSL/Config.pm' under '/usr/share/perl5 /opt/myperl' ...
[INFO]  Found: /opt/myperl/lib/vendor_perl/5.40.1/OpenXPKI/Crypto/Backend/OpenSSL/Config.pm
[INFO]  SHA256 (current): b3424aa1286bd9c2401df88000eae127d4c9b90d291ac95086a2cedc21aea300
[INFO]  Detected source version: v3.32
[INFO]  Backup created: /opt/myperl/lib/vendor_perl/5.40.1/OpenXPKI/Crypto/Backend/OpenSSL/Config.pm.bak.20
[INFO]  SHA256 (temp file): 0a5009d93ba68873c7489182cfdbe3432555191ed885d73034f92de30b3cd02f
[INFO]  Hash verification successful - replacing target file.
[INFO]  File replaced: /opt/myperl/lib/vendor_perl/5.40.1/OpenXPKI/Crypto/Backend/OpenSSL/Config.pm
[INFO]  SHA256 (after patch): 0a5009d93ba68873c7489182cfdbe3432555191ed885d73034f92de30b3cd02f
[INFO]  Hash verification successful.
[INFO]  Patch complete: v3.32 -> 0a5009d93ba68873c7489182cfdbe3432555191ed885d73034f92de30b3cd02f
```

Case: patch applied successfully

```
# sudo bash WRS-SA-2026-001-patch-openxpki-config.sh --apply
[INFO]  Searching for 'OpenXPKI/Crypto/Backend/OpenSSL/Config.pm' under '/usr/share/perl5 /opt/myperl' ...
[INFO]  Found: /opt/myperl/lib/vendor_perl/5.40.1/OpenXPKI/Crypto/Backend/OpenSSL/Config.pm
[INFO]  SHA256 (current): 0a5009d93ba68873c7489182cfdbe3432555191ed885d73034f92de30b3cd02f
[INFO]  File is already up to date (0a5009d93ba68873c7489182cfdbe3432555191ed885d73034f92de30b3cd02f). No pa
```

Mitigation: Pathlen Constraint on Issuing CAs

In all customer environments managed by White Rabbit Security, every Issuing CA certificate carries `basicConstraints: CA:TRUE, pathLen:0`. This constraint restricts the CA to signing end-entity certificates only; any rogue CA certificate issued via this vulnerability would be rejected by compliant certificate validators when used to sign further certificates, substantially reducing the blast radius of a successful exploit.

Verify that your Issuing CAs have the pathlen constraint in place:

```
openssl x509 -in <issuing-ca-cert.pem> -text -noout | grep -A1 'Basic Constraints'
```

Expected output for a mitigated Issuing CA:

```
X509v3 Basic Constraints: critical
    CA:TRUE, pathlen:0
```

If `pathlen` is absent or set to a value greater than 0, a rogue CA issued via this vulnerability could be used to sign further certificates and the full impact applies.

Checking for Prior Exploitation

White Rabbit Security is developing a verification script that queries the OpenXPKI database for certificates that bear evidence of this injection attack. A negative result indicates that the vulnerability was not exploited (or that any such certificates have since been deleted from the database). The script will be made available to affected customers; contact security@whiterabbitsecurity.com for access.

Disclosure Timeline

Date	Event
2026-04-14	Vulnerability reported to White Rabbit Security by Alexander Klink; initial analysis begins; vulnerability confirmed on the same day
2026-04-15	Assessment extended to identify all affected versions and enumerate exploitable SAN types
2026-04-17	Hotfix development begins; testing on affected installations starts
2026-04-20	Root cause analysis complete; code fix committed to mainline for inclusion in OpenXPKI 3.32.16
2026-04-21	Hotfix script testing completed across multiple installations
2026-04-22	Draft advisory distributed to affected customer environments; CVE registration initiated
TBD	CVE identifier assigned; advisory updated to Final status and published

This advisory follows a coordinated vulnerability disclosure process. White Rabbit Security thanks Alexander Klink for identifying the vulnerability and for the professional and responsible manner in which it was reported and handled throughout the disclosure process.

References

- **CVE identifier** — pending assignment; this advisory will be updated upon registration
- **OpenXPKI Community Edition source repository** — `https://github.com/openxpki/openxpki`

- **Affected source file** — `OpenXPKI/Crypto/Backend/OpenSSL/Config.pm`, line 641
- **OpenSSL configuration file format reference** — `https://www.openssl.org/docs/manmaster/man5/`
- **CVSS v3.1 Calculator** — `https://www.first.org/cvss/calculator/3.1`

Credits

This vulnerability was discovered by **Alexander Klink** (security researcher) and reported to White Rabbit Security on 2026-04-14 together with a fully functional proof-of-concept demonstrating the trust boundary bypass. White Rabbit Security thanks Alexander Klink for the quality of the technical report and for adhering to a responsible disclosure process.

For questions or additional information regarding this advisory, contact security@whiterabbitsecurity.com.