Full credit is given to the above companies including the OS that this PDF file was generated!

### *Rocky Enterprise Linux 9.2 Manual Pages on command 'crypto-policies.7'*

***$ man crypto-policies.7***

CRYPTO-POLICIES(7) CRYPTO-POLICIES(7)

NAME

crypto-policies - system-wide crypto policies overview

DESCRIPTION

The security of cryptographic components of the operating system does not remain constant over time. Algorithms, such as cryptographic hashing and encryption, typically have a lifetime, after which they are considered either too risky to use or plain insecure. That means, we need to phase out such algorithms from the default settings or completely disable them if they could cause an irreparable problem.

While in the past the algorithms were not disabled in a consistent way and different applications applied different policies, the system-wide crypto-policies followed by the crypto core components allow consistently deprecating and disabling algorithms system-wide.

Several preconfigured policies (DEFAULT, LEGACY, FUTURE, and FIPS) and subpolicies are included in the crypto-policies(7) package. System administrators or third-party vendors can define custom policies.

For rationale, see RFC 7457 for a list of attacks taking advantage of legacy crypto algorithms.

## COVERED APPLICATIONS

Crypto-policies apply to the configuration of the core cryptographic subsystems, covering TLS, IKE, IPSec, DNSSec, and Kerberos protocols; i.e., the supported secure communications protocols on the base operating system.

Once an application runs in the operating system, it follows the default or selected policy and refuses to fall back to algorithms and protocols not within the policy, unless the user has explicitly requested the application to do so. That is, the policy applies to the default behavior of applications when running with the system-provided configuration but the user can override it on an application-specific basis.

The policies currently provide settings for these applications and libraries:

? BIND DNS name server daemon (scopes: BIND, DNSSec)

? GnuTLS TLS library (scopes: GnuTLS, SSL, TLS)

? OpenJDK runtime environment (scopes: java-tls, SSL, TLS)

? Kerberos 5 library (scopes: krb5, Kerberos)

? Libreswan IPsec and IKE protocol implementation (scopes: libreswan, IPSec, IKE)

? NSS TLS library (scopes: NSS, SSL, TLS)

? OpenSSH SSH2 protocol implementation (scopes: OpenSSH, SSH)

? OpenSSL TLS library (scopes: OpenSSL, SSL, TLS)

? libssh SSH2 protocol implementation (scopes: libssh, SSH)

Applications using the above libraries and tools are covered by the cryptographic policies unless they are explicitly configured otherwise.

## PROVIDED POLICIES

### LEGACY

This policy ensures maximum compatibility with legacy systems at the cost of being less secure. It allows the TLS 1.2, and TLS 1.3 protocols, as well as IKEv2 and SSH2.  DSA is not allowed, while RSA and Diffie-Hellman parameters are accepted if no less than 2048 bits. This policy provides at least 80-bit security.

? MACs: all HMAC with SHA-1 or better + all modern MACs (Poly1305 etc.)

? Curves: all prime >= 255 bits (including Bernstein curves)

? Signature algorithms: with SHA1 hash or better (no DSA)

? Ciphers: all available >= 112-bit key, >= 128-bit block (excluding 3DES and RC4)

? Key exchange: ECDHE, RSA, DHE (no DHE-DSS)

? DH params size: >= 2048

? RSA keys size: >= 2048

? TLS protocols: TLS >= 1.2, DTLS >= 1.2

DEFAULT

The DEFAULT policy is a reasonable default policy for today?s standards. It allows the TLS 1.2, and TLS 1.3 protocols, as well as IKEv2 and SSH2. The Diffie-Hellman parameters are accepted if they are at least 2048 bits long. This policy provides at least 112-bit security.

? MACs: all HMAC with SHA-1 or better + all modern MACs (Poly1305 etc.)

? Curves: all prime >= 255 bits (including Bernstein curves)

? Signature algorithms: with SHA-224 hash or better (no DSA)

? TLS Ciphers: >= 128-bit key, >= 128-bit block (AES, ChaCha20, including AES-CBC)

? non-TLS Ciphers: as TLS Ciphers

? Key exchange: ECDHE, RSA, DHE (no DHE-DSS)

? DH params size: >= 2048

? RSA keys size: >= 2048

? TLS protocols: TLS >= 1.2, DTLS >= 1.2

FUTURE

A conservative security policy that is believed to withstand any near-term future attacks. This policy does not allow the use of SHA-1 in signature algorithms. The policy also provides some (not complete) preparation for post-quantum encryption support in form of 256-bit symmetric encryption requirement. The RSA and Diffie-Hellman parameters are accepted if larger than 3071 bits. This policy provides at least 128-bit security.

? MACs: all HMAC with SHA-256 or better + all modern MACs (Poly1305 etc.)

? Curves: all prime >= 255 bits (including Bernstein curves)

? Signature algorithms: with SHA-256 hash or better (no DSA)

? TLS Ciphers: >= 256-bit key, >= 128-bit block, only Authenticated Encryption (AE) ciphers, no CBC ciphers

? non-TLS Ciphers: same as TLS ciphers with added non AE ciphers, CBC ones enabled only in Kerberos

? Key exchange: ECDHE, DHE (no DHE-DSS, no RSA)

? DH params size: >= 3072

? RSA keys size: >= 3072

? TLS protocols: TLS >= 1.2, DTLS >= 1.2

FIPS

A policy to aid conformance to the FIPS 140-2 requirements. This policy is used internally by the fips-mode-setup(8) tool which can switch the system into the FIPS 140-2 mode. This policy provides at least 112-bit security.

? MACs: SHA-256 or better

? Curves: all prime >= 256 bits

? Signature algorithms: with SHA-256 hash or better (no DSA)

? TLS Ciphers: >= 128-bit key, >= 128-bit block (AES, excluding AES-CBC)

? non-TLS Ciphers: same as TLS Ciphers

? Key exchange: ECDHE, DHE (no DHE-DSS, no RSA)

? DH params size: >= 2048

? RSA params size: >= 2048

? TLS protocols: TLS >= 1.2, DTLS >= 1.2

EMPTY

All cryptographic algorithms are disabled (used for debugging only, do not use).

## CRYPTO POLICY DEFINITION FORMAT

The crypto policy definition files have a simple syntax following an INI file key = value syntax with these particular features:

? Comments are indicated by # character. Everything on the line following the character is ignored.

? Backslash \ character followed immediately with the end-of-line character indicates line continuation. The following line is concatenated to the current line after the backslash and end-of-line characters are removed.

? Value types for integer options can be decimal integers (option = 1).

? Multiple-choice options can be specified by setting them to a list of values (option = value1 value2). This list can further be altered by prepending/omitting/appending values (option = prepended -omitted appended). A follow-up reassignment will reset the list. The latter syntax cannot be combined with the former one in the same directive. Setting an option to an empty list is possible with option =.

? Asterisk sign can be used for wildcard matching as a shortcut for specifying multiple values when setting multiple-choice options. Note that wildcard matching can lead to future updates implicitly enabling algorithms not yet available in the current version. If this is a concern, do not use wildcard-matching outside of algorithm-omitting directives.

? In order to limit the scope of the directive and make it affect just some of the backends, the following extended syntax can be used: option@scope = ..., option@{scope1,scope2,...} = .... Negation of scopes is possible with option@!scope / 'option@{scope1,scope2,...}. Scope selectors are case-insensitive.

The available options are:

? mac: List of allowed MAC algorithms

? group: List of allowed groups or elliptic curves for key exchanges for use with other protocols

? hash: List of allowed cryptographic hash (message digest) algorithms

? sign: List of allowed signature algorithms

? cipher: List of allowed symmetric encryption algorithms (including the modes) for use with other protocols

? key_exchange: List of allowed key exchange algorithms

? protocol: List of allowed TLS, DTLS and IKE protocol versions; mind that some backends do not allow selectively disabling protocols versions and only use the oldest version as the lower boundary.

? min_dh_size: Integer value of minimum number of bits of parameters for DH key exchange

? min_dsa_size: Integer value of minimum number of bits for DSA keys

? min_rsa_size: Integer value of minimum number of bits for RSA keys

? sha1_in_certs: Value of 1 if SHA1 allowed in certificate signatures, 0 otherwise (Applies to GnuTLS back end only.)

? arbitrary_dh_groups: Value of 1 if arbitrary group in Diffie-Hellman is allowed, 0 otherwise

? ssh_certs: Value of 1 if OpenSSH certificate authentication is allowed, 0 otherwise

? ssh_etm: Value of 1 if OpenSSH EtM (encrypt-then-mac) extension is allowed, 0 otherwise

Full policy definition files have suffix .pol, subpolicy files have suffix .pmod. Subpolicies do not have to have values set for all the keys listed above.

The effective configuration of a policy with subpolicies applied is the same as a configuration from a single policy obtained by concatenating the policy and the subpolicies in question.

Policy file placement and naming:

The policy files shipped in packages are placed in /usr/share/crypto-policies/policies and the subpolicies in /usr/share/crypto-policies/policies/modules.

Locally configured policy files should be placed in /etc/crypto-policies/policies and subpolicies in

/etc/crypto-policies/policies/modules.

The policy and subpolicy files must have names in upper-case except for the .pol and .pmod suffix as the update-crypto-policies command always converts the policy name to upper-case before searching for the policy on the filesystem.

## COMMANDS

update-crypto-policies(8)

This command manages the policies available to the various cryptographic back ends and allows the system administrator to change the active cryptographic policy.

fips-mode-setup(8)

This command allows the system administrator to enable, or disable the system FIPS mode and also apply the FIPS cryptographic policy which limits the allowed algorithms and protocols to these allowed by the FIPS 140-2 requirements.

## NOTES

Known notable exceptions

? Go-language applications do not yet follow the system-wide policy.

? GnuPG-2 application does not follow the system-wide policy.

In general only the data-in-transit is currently covered by the system-wide policy.

If the system administrator changes the system-wide policy with the update-crypto-policies(8) command it is advisable to restart the system as the individual back-end libraries read the configuration files usually during their initialization. The changes in the policy thus take place in most cases only when the applications using the back-end libraries are restarted.

Removed cipher suites and protocols

The following cipher suites and protocols are completely removed from the core cryptographic libraries listed above:

? DES

? All export grade cipher suites

? MD5 in signatures

? SSLv2

? SSLv3

? All ECC curves smaller than 224 bits

? All binary field ECC curves

Cipher suites and protocols disabled in all predefined policies

The following ciphersuites and protocols are available but disabled in all predefined crypto policies:

? DH with parameters < 2048 bits

? RSA with key size < 2048 bits

? Camellia

? RC4

? ARIA

? SEED

? IDEA

? Integrity only ciphersuites

? TLS CBC mode ciphersuites using SHA-384 HMAC

? AES-CCM8

? all ECC curves incompatible with TLS 1.3, including secp256k1

? IKEv1

Notable irregularities in the individual configuration generators

? OpenSSL and NSS: Disabling all TLS and/or all DTLS versions isn?t actually possible. Trying to do so will result in the library defaults being applied instead.

? OpenSSL: The minimum length of the keys and some other parameters are enforced by the @SECLEVEL value which does not provide a fine granularity. The list of TLS ciphers is not generated as an exact list but by subtracting from all the supported ciphers for the enabled key exchange methods. For that reason there is no way to disable a random cipher. In particular all AES-128 ciphers are disabled if the AES-128-GCM is not present in the list; all AES-256 ciphers are disabled if the AES-256-GCM is not present. The CBC ciphers are disabled if there isn?t HMAC-SHA1 in the hmac list and AES-256-CBC in the cipher list. To disable the CCM ciphers both

AES-128-CCM and AES-256-CCM must not be present in the cipher list.

? GnuTLS: The minimum length of the keys and some other parameters are enforced by min-verification-profile setting in the GnuTLS configuration file which does not provide fine granularity.

? GnuTLS: PSK key exchanges have to be explicitly enabled by the applications using them.

? GnuTLS: HMAC-SHA2-256 and HMAC-SHA2-384 MACs are disabled due to concerns over the constant-timedness of the implementation.

? OpenSSH: DH group 1 is always disabled on server even if the policy allows 1024 bit DH groups in general. The OpenSSH configuration option HostKeyAlgorithms is set only for the SSH server as otherwise the handling of the existing known hosts entries would be broken on client.

? Libreswan: The key_exchange parameter does not affect the generated configuration. The use of regular DH or ECDH can be limited with appropriate setting of the group parameter.

HISTORY

The ECDHE-GSS and DHE-GSS algorithms are newly introduced and must be specified in the base policy for the SSH GSSAPI key exchange methods to be enabled. Previously the legacy SSH GSSAPI key exchange methods were automatically enabled when the SHA1 hash and DH parameters of at least 2048 bits were enabled.

Before the introduction of the custom crypto policies support it was possible to have an completely arbitrary crypto policy created as a set of arbitrary back-end config files in /usr/share/crypto-policies/<POLICYNAME> directory. With the introduction of the custom crypto policies it is still possible but there must be an empty (possibly with any comment lines) <POLICYNAME>.pol file in /usr/share/crypto-policies/policies so the update-crypto-policies command can recognize the arbitrary custom policy. No subpolicies must be used with such an arbitrary custom policy. Modifications from local.d will be appended to the files provided by the policy.

The use of the following historaically available options is discouraged:

? min_tls_version: Lowest allowed TLS protocol version (recommended replacement: protocol@TLS)

? min_dtls_version: Lowest allowed DTLS protocol version (recommended replacement: protocol@TLS)

The following options are deprecated, please rewrite your policies:

? ike_protocol: List of allowed IKE protocol versions (recommended replacement: protocol@IKE, mind the relative position to other protocol directives).

? tls_cipher: list of allowed symmetric encryption algorithms for use with the TLS protocol (recommended replacement: cipher@TLS, mind the relative position to other cipher directives).

? ssh_cipher: list of allowed symmetric encryption algorithms for use with the SSH protocol (recommended replacement: cipher@SSH, mind the relative position to other cipher directives).

? ssh_group: list of allowed groups or elliptic curves for key exchanges for use with the SSH protocol (recommended replacement: group@SSH, mind the relative position to other group directives).

? sha1_in_dnssec: Allow SHA1 usage in DNSSec protocol even if it is not present in the hash and sign lists (recommended replacements: hash@DNSSec, sign@DNSSec).

## FILES

/etc/crypto-policies/back-ends

The individual cryptographical back-end configuration files. Usually linked to the configuration shipped in the crypto-policies package unless a configuration from local.d is added.

/etc/crypto-policies/config

A file containing the name of the active crypto-policy set on the system.

/etc/crypto-policies/local.d

Additional configuration shipped by other packages or created by the system administrator. The contents of the

<back-end>-file.config is appended to the configuration from the policy back end as shipped in the crypto-policies package.

/usr/share/crypto-policies/policies

System policy definition files.

/usr/share/crypto-policies/policies/modules

System subpolicy definition files.

/etc/crypto-policies/policies

Custom policy definition files as configured by the system administrator.

/etc/crypto-policies/policies/modules

Custom subpolicy definition files as configured by the system administrator.

/usr/share/crypto-policies/<'POLICYNAME'>

Pre-generated back-end configurations for policy POLICYNAME.

## SEE ALSO

update-crypto-policies(8), fips-mode-setup(8)

## AUTHOR

Written by Tom?? Mr?z.

crypto-policies 12/15/2022 CRYPTO-POLICIES(7)