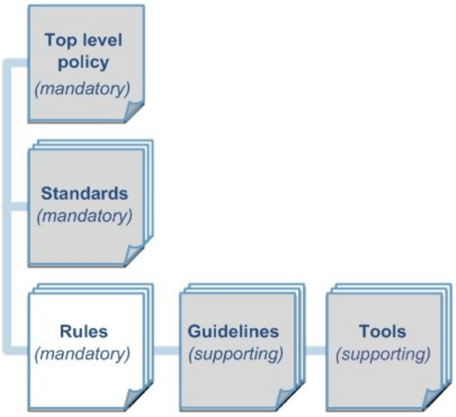
KPN Security Policy

KSP – Rule

Title	**Cryptography**	 Top level policy (mandatory) Standards (mandatory) Rules (mandatory) Guidelines (supporting) Tools (supporting)
ID	**KSP-FA05-RL07**	
Funct. Area	FA05 – System and Network Security	
Date	3 February 2017	
Version	v2.9	
Status	Approved	
Owner	CISO	

Summary

This document describes the requirements for all cases of encrypted communication, signed communication, use of PKI certificates, and use and management of encryption keys.
This document excludes requirements for when to use cryptography as those are described in other parts of the policy and those parts will refer to this document for the how-to.

Version history

Version	*Date*	*Comments*
v1.0	17 September 2013	Approved in SSM
v1.1	11 October 2013	Update based on consistency check
v2.0	31 March 2014	Update based on use of policy and changes due to recent developments.
v2.1	1 August 2014	Added cross references between related rules, added related documents, new are rules.
v2.2	15 October 2014	Look up of the safe curves are now more explicit
v2.3	23 January 2015	Various updates based on feedback in order to make clearer what is expected.
v2.4	20 April 2015	Updated the rekey wording and added examples, made the encrypted private key transport explicit, emphasized that non-Perfect-Forward-Secrecy ciphers may be used to cover compatibility. Renamed R23 by removing "Web", clarify TLSv1.2, TLSv1.1 and TLSv1.0 usage.
v2.5	13 November 2015	Update based on feedback from the organisation
v2.6	5 February 2016	Removed non-existing SHA version. Added details to HAVAL use. Added examples of salt use. Explicitly adding prevention to downgrade attacks. Added AES-XTS.
v2.7	29 July 2016	R01: Urandom has preference. R02: Minimum is FIPS140-2 level 2. R03: Certificate registration details points to KSP-FA05-GL04. R06: Clarifications. R09: Added OCSP Stapling. R12: Additions to allowed wildcard usage. R14: Blowfish, Twofish and ECDH added.

		R24: IPSec statement expanded. R26: Added statement on Puppet use. R27: ECC options made explicit. R28: Textual tightening. R29: Textual tightening. R30 new: on key destruction.
v2.8	2 November 2016	R12: Adjusted wildcard usage to focus on making the private key stored in a tamperproof way R21: Mixed-content moved and added to KSP-FA05-RL11 - (Web) Application Security as requirement R39
v2.9	3 February 2017	R01: Enhanced random number generation. R04: Widened application for a private key. R09: Besides clarity on PKI, also added PGP/GPG in a WoT. R12: Widened the application of wildcard certificates, in particular a risk analyses is now leading for external infrastructure suppliers. R14: Added IDEA, including application specifications. Blowfish is now deprecated to legacy use only. R15: Added EdDSA and a Post Quantum list of algorithms. R16: Minimal key length for DSA enhanced. R17: MQV is now removed and minimum key length enhanced. The ECDH and DH parameters must be freshly generated. R18: WIRLPOOL-T and HAVAL are deprecated to legacy applications. Application of MSCHAPv2 explicitly scoped. Added GOST, Skein, JH, Grostl, BLAKE and BLAKE2 R19: HMAC keys enhanced from 112 to 128 bits including clarification to several usage example. R23: Added TLSv1.3. TLSv1.1 en TLSv1.0 are now legacy protocols. The highest possible TLS version must be activated. R26: Equal to the use of a Puppet CA a VMWare VMCA may also be used, when strict demands are met.

Disclaimer

The content of this document is to describe KPN's policy on this specific topic. If and when this document is partly or fully disclosed to parties outside of KPN, it's important to hereby note towards those parties that this contains KPN's intended policy and cannot in any way be read or construed to be an explicit or implied formal guarantee or promise that its content can always be fully executed or complied to.

ID	KSP-FA05-RL07-R01
Title	Cryptographic Key Generation, Random Bit Generator
Description	Use a known good entropy source to generate cryptographic keys, identifiers or random seeds. Known good entropy sources for an application combine several random sources and use a cryptographic secure hash algorithm over the values of the sources. Known good sources are: - On Apple iOS use SecRandomCopyBytes - On Android use java.security.SecureRandom and must not be combined with setSeed(). - On Unix and Linux systems use /dev/urandom - On Windows use CryptGenRandom or RtlGenRandom - In .Net use System.Security.Cryptography.RNGCryptoServiceProvider - In Java use java.security.SecureRandom - In Perl use Math::Random::Secure - In PHP use openssl_random_pseudo_bytes or mcrypt_create_iv - In Python use os.urandom - In Ruby use SecureRandom At (re)boot time of a system, when there isn't sufficient entropy gathered yet, all processes generating new key material must block until sufficient entropy has been gathered. Random bit generators must be compliant with one of the following standards: - SP 800-90A, revision 1. - ANSI X9.62:2005, Annex D. The use of the following methods or entropy sources are forbidden: - EC_Dual_DRBG
Relating document	NIST Special Publication 800-90A (revision 1): Recommendation for Random Number Generation Using Deterministic Random Bit Generators

ID	KSP-FA05-RL07-R02
Title	Cryptographic Key Generation, Cryptographic Module
Description	For high-security services where the entropy source needs to be protected from tampering a cryptographic hardware module must be used. The Cryptography Module of the product used must be compliant with the FIPS-140-2 level 2 standard.
Relating document	FIPS PUB 140-2: Security Requirements for Cryptographic Modules

ID	KSP-FA05-RL07-R03
Title	Registration of Key Pair properties
Description	For each public/private key pair the following must be registered: - The owner - The intended use (infrastructure on which deployed) - Key length - Key Algorithm (including curve if Elliptic Curve is used) - Hash function - CA used for signing - Serial number (if applicable, like for certificates) - Validity from and to dates Registration may be omitted when the certificates are ordered through the central certificate application process. Note: This rule is implicitly satisfied when the certificates are ordered via internal processes.
Relating document	KSP-FA05-GL04 - Certificate handling: pre- and post-ordering process and checks

ID	KSP-FA05-RL07-R04
Title	Key pair privacy
Description	The private part of the key pair should be generated on the device on which it will be used. To support this: - Certificate signing request must be submitted by CSR (Certificate Signing Request). - Alternatively, key pairs must be generated locally by the key-pair owner or a delegated party within KPN. - For load balancing purposes, it is allowed to copy the private key into multiple devices.
Relating document	CSR: http://en.wikipedia.org/wiki/Certificate_signing_request Requirement: KSP-FA05-RL07-R06 (Private Key transport and storage)

ID	KSP-FA05-RL07-R05
Title	Key Compromise
Description	Compromised keys must be regenerated and rekeyed, not updated. During generation, the new key must be generated from a new set of data (no re-use of data used to generate the compromised key) to ensure its full independence from the compromised key. For PKI, the CA must be informed of the compromise by means of the contract manager with the main purpose to invalidate the trust in a key. Example keys involved are: • SSH private keys for hosts or users • Private keys associated to PKI, PGP and other types of certificates • Diffie-Hellman param files • Group keys • Key used for symmetric encryption of e.g. files, databases, file-systems or any other type of arbitrary data
Relating document	N/A

ID	KSP-FA05-RL07-R06
Title	Private key transport and storage
Description	Private keys are one of the foundations for the security of a service and its data. A private key must be protected during both transport and its storage: Storage: - Based on the business impact assessment the necessity must become conclusive if storing the private key securely in a Hardware Security Module is a requirement. - Keys stored on a file system must be protected with the strictest possible file system permissions. - Physical security steps must be taken to limit access to the key to authorized personnel. Any form of physical security in addition to building access, that allows verification of access (see point below) will do. - If a stored key is accessed this must be verifiable/detectable. Transport: Before transporting a private key between systems, the private key must be encrypted and use message integrity rules to provide tamper resistance. For key encryption and integrity, the following rules are mandatory requirements: • KSP-FA05-RL07-R14 (Encryption Algorithms) • KSP-FA05-RL07-R18 (Hash Algorithms) • KSP-FA05-RL01-R01 (Password length) – for static passwords • KSP-FA05-RL01-R02 (Password complexity) In addition: • The transport method must be encrypted itself, e.g. use SSH, SFTP, HTTPS or FTP-SSL. • Use a shared secret (e.g. a passphrase or HMAC) or Digital Signatures to authenticate the sender of a private key when the sender and receiver are different entities.
Relating document	Requirements: KSP-FA05-RL07-R14 (Encryption Algorithms) KSP-FA05-RL07-R15 (Digital Signatures Algorithms) KSP-FA05-RL07-R18 (Hash Algorithms) KSP-FA05-RL07-R19 (HMAC)

ID	KSP-FA05-RL07-R07
Title	Public Key Exchange
Description	To authenticate a service, host, machine or user a public key must be exchanged using a secure key exchange method. This is to challenge the ownership of the private key and by doing so to prevent identity spoofing. Secure public key exchange methods are listed in KSP-FA05-TL02 - Cryptographic algorithms and cipher suites.
Relating document	Key exchange mechanisms (http://en.wikipedia.org/wiki/Key_exchange) KSP-FA05-TL02 - Cryptographic algorithms and cipher suites

ID	KSP-FA05-RL07-R08
Title	Certificate Authority
Description	Public Key Infrastructure builds trust relationships using trusted third parties, the Certificate Authorities. All used Certificate Authorities: - Must comply with the European Telecommunications Standards Institute (ETSI) standard "ETSI TS 101 456”. - Use FIPS 140-2 level 3 compliant hardware security modules or better. - Have a published CPS (Certification Practice Statement), this also means that our use of the certificate must follow the CPS.
Relating document	ETSI: http://www.etsi.org/deliver/etsi_ts/101400_101499/101456/01.04.03_60/ts_101456v010403p.pdf FIPS: http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

ID	KSP-FA05-RL07-R09
Title	Certificates
Description	Certificates in the format X.509 can be used in a Public Key Infrastructure or web-of-trust. In a Public Key Infrastructure (PKI) context the follow applies: - Certificates can identify hosts, servers, users, processes, end-points and (individual) products. - Certificates comply to RFC5280. - Domain validation when used for identification. - Revocation must be implemented using CRLs (RFC5280), OCSP (RFC2560, RFC5019 or RFC6990) or OCSP stapling (RFC6066 and RFC6961). In web-of-trust context the following applies: - The certificates must comply to PGP/GPG standard, i.e. RFC 4880 and additional RFCs. - The certificates are created for a group or individual. - The certificates are digitally signed by others or a master key, after the full fingerprint is compared and matches. PGP/GPG keys which have been used publicly must be invalidated. The invalidation must be published.
Relating document	RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile Requirement: KSP-FA05-RL07-R08 (Certificate Authority)

ID	KSP-FA05-RL07-R10
Title	Use of certificates
Description	Certificates must be verified in compliance to RFC 6125. When this fails, the connection must not be used anymore. Example reasons to disconnect: certificate not valid, certificate not verifiable (including by configuration error), untrustworthy protocol usage, or other error from the TLS, IPsec or EAP handshake.
Relating document	RFC 6125: Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)

ID	KSP-FA05-RL07-R11
Title	Binding Certificates
Description	Each certificate must be bound to use for an as small as possible set of identities, example one host, virtual machine, one service, person or department. An SSL off-loader or load-balancer may hold the certificate and private key to serve/off-load the SSL sessions for one cluster of nodes serving the same service.
Relating document	N/A

ID	KSP-FA05-RL07-R12
Title	Wildcard Certificates
Description	Wildcard certificates must be scoped as much as possible to specific subdomains and are not allowed to be effective for the first subdomain. This is to limit the impact of a compromise. (example: *.webmail.cm.kpn.com is better than *.cm.kpn.com for consumer market webmail servers and *.kpn.com is never allowed). To limit impact in case of compromise the use of wildcard certificate is: - limited to a single service-type and purpose, i.e. exclusively for mail servers or another specific service-type; - must be scoped to the most specific subdomain possible, i.e. *.webmail.cm.kpn.com is better than *.cm.kpn.com for the consumer market webmail servers; - not allowed to be used for the first subdomain, i.e. it is not allowed to be used as *.kpn.com or *.kpn.net. Alternatively, a single service may also use a customer_name.service.domain.tld scheme. The solution is only acceptable when the setup honours the single service-type restriction, is scoped to the most specific subdomain possible and in addition the private key must be stored according to KSP-FA05-RL07-R06. For environments with a high demand on integrity and confidentiality the use of tamperproof solutions must be assessed. E.g. an HSM with a certification of FIPS14-2 level 2 or better in the context of health sector related services and vital infrastructure. Only when wildcard certificates are in use on the infrastructure from an external party a security officer must first perform a risk analyses to determine if the external party can be exempt from this policy. Possible exception: if the first subdomain is limited to a single service-type and purpose. Example: *.kpnxchange.com as a mail-cluster environment.
Relating document	http://en.wikipedia.org/wiki/Wildcard_certificate Requirement: KSP-FA05-RL07-R04 (Key pair privacy)

ID	KSP-FA05-RL07-R13
Title	Lifetimes for keys
Description	Keys used must have a maximum lifetime of 36 months. Examples of keys are: - Keys belonging to certificates - Diffie-Hellman parameters - Static passwords - pre-shared keys (PSK) - master keys - SSH keys for systems and administrators - PGP keys Exception to this is the key pairs used by a Certificate Authority.
Relating document	Baseline Requirements Certificate Policy for the Issuance and Management of Publicly-Trusted Certificates (current: 1.3.4)

ID	KSP-FA05-RL07-R14
Title	Encryption Algorithms
Description	One of the following encryption primitives must be used for encryption and decryption: - AES-256, AES-192 and AES-128 - XSalsa20/20 - Salsa20/20 - Twofish - IDEA; the key must be generated using a hash algorithm from KSP-FA05-RL07-R18, like SHA2. For AES use known good AES-authenticated modes: - GCM - CCM Use non-authenticated AES modes only when explicitly required: - CTR - XTS The following encryption primitives should not be used. Use only for legacy support or explicit compatibility requirements: - AES-256-CBC, AES-192-CBC and AES-128-CBC - Three-key Triple DES - Blowfish All not explicitly mentioned encryption algorithms are not allowed. Example are: - RC4 - All EXPORT ciphers - All encryption algorithms resulting in less than 112 security bits The use of a random nonce or initialisation vector (IV) with sufficient length is mandatory with each of these encryption algorithms. To generate a good nonce or IV use a good random bit generator.
Relating document	NIST Special Publication 800-131Ar1: Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths KSP-FA05-TL02 - Cryptographic algorithms and cipher suites Requirement: KSP-FA05-RL07-R01 (Cryptographic Key Generation, Random Bit Generator)

ID	KSP-FA05-RL07-R15
Title	Digital Signatures Algorithms
Description	One of the following digital signature algorithms must be used: - CECPQ1-ECDSA - EdDSA - ECDSA - RSA - DSA For Post Quantum resistance, the following algorithms must be used: - CECPQ1-ECDSA (New Hope) - NTRU-6130 (Lattice-based) - McEliece or Goppa-based McEliece - SPHINCS-256 (hash based signatures) These algorithms can be used in authentication phases or integrity checks.
Relating document	NIST Special Publication 800-131Ar1: Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths KSP-FA05-TL02 - Cryptographic algorithms and cipher suites Requirement: KSP-FA05-RL07-R27 (Choosing safe curves for elliptic curve cryptography)

ID	KSP-FA05-RL07-R16
Title	Digital Signature Generation and Verification
Description	Digital signatures must have at least 112 bits of security strength. This means: - For EC: key length ≥ 224 - For RSA: key length ≥ 2048 - For DSA: key length 3072/256 or 4096/256.
Relating document	NIST Special Publication 800-131Ar1: Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths KSP-FA05-TL02 - Cryptographic algorithms and cipher suites

ID	KSP-FA05-RL07-R17
Title	Key Agreement
Description	For Key agreement one of the following must be used: - ECDH (Diffie-Hellman) with a minimal key length of 256 bits. - DH (Diffie-Hellman) with a minimal key length of 2048 bits. All ECDH and DH parameters must be newly generated before use to assure unicity and avoid default parameters reuse between various installations.
Relating document	NIST Special Publication 800-131Ar1: Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths KSP-FA05-TL02 - Cryptographic algorithms and cipher suites

ID	KSP-FA05-RL07-R18
Title	Hash Algorithms
Description	One of the following hash algorithms must be used: - SHA-2: SHA-512, SHA-384, SHA-256 or better - SHA-3 - GOST R 34.11-94 (256 bit hash) - Skein - JH - Grøstl - BLAKE and BLAKE2 The following hash algorithms should not be used. Use only for legacy support or explicit compatibility requirements: - SHA-1: for Non-digital signature generation applications only, not for Digital signature verification nor Digital signature generation after 2013 - SHA-224: for Non-digital signature generation applications only, not for Digital signature verification nor Digital signature generation after 2014 - WIRLPOOL-T - HAVAL, using >= 160 bit with 3 rounds The following hash algorithms must not be used: - SHA-0 - HAVAL, using 128 bit with 3 rounds - RIPEMD - MD5 - MD4 - MD2 Exception: the use of MS-CHAPv2 is allowed, when transported as payload within an encrypted protocol, like TLS or various EAP protocols using a TLS based outer layer.
Relating document	NIST Special Publication 800-131Ar1: Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths KSP-FA05-TL02 - Cryptographic algorithms and cipher suites

ID	KSP-FA05-RL07-R19
Title	HMAC
Description	HMAC is a keyed-hash message authentication code and must use: - A hash algorithm as defined in KSP-FA05-RL07-R18 - A key with a length ≥ 128 bits - The key should be generated using a known good random bit generator Known good examples are: - HMAC-SHA1 with a key length of 160 bit. - HMAC-SHA2 with a key length of 256 bit. - HMAC-SHA3 with a key length of 256 bit. HMAC-MD5 must not be used.
Relating document	http://csrc.nist.gov/publications/nistpubs/800-107-rev1/sp800-107-rev1.pdf Requirements: KSP-FA05-RL07-R01 (Cryptographic Key Generation, Random Bit Generator) KSP-FA05-RL07-R18 (Hash Algorithms)

ID	KSP-FA05-RL07-R20
Title	Salt use
Description	The length of the randomly-generated portion of the salt must be at least 128 bits. The salt must be generated using a known good random bit generator. Example uses for a salt: • KSP-FA05-RL01-R11 (Password storage) • KSP-FA05-RL07-R14 (Encryption Algorithms) • KSP-FA05-RL07-R28 (Password hashing) • KSP-FA05-RL07-R29 (Key stretching algorithms)
Relating document	Requirement: KSP-FA05-RL07-R01 (Cryptographic Key Generation, Random Bit Generator)

ID	KSP-FA05-RL07-R22
Title	Maximum token lifetime
Description	Authentication tickets/tokens, e.g. Kerberos, AFS and Windows logon, must have a maximum lifetime of 6 hours. During their period of validity tokens may be refreshed automatically.
Relating document	N/A

ID	KSP-FA05-RL07-R23
Title	Application data encryption
Description	For encryption of transported application data applications: • The highest available TLS version must be activated. • Protection against downgrade attacks must be activated. When this feature is absent: TLSv1.0 must be de-activated. • TLSv1.3 must be enabled, when available. • TLSv1.2 must be enabled. • TLSv1.1 may also be enabled. • TLSv1.0 may only be enabled when there is a need to be able to communicate with legacy systems. When this need is absent, it must be disabled. • SSLv3 is not allowed to be enabled and must be disabled. • SSLv2 is not allowed to be enabled and must be disabled.
Relating document	http://tools.ietf.org/html/rfc5246 KSP-FA05-TL02 - Cryptographic algorithms and cipher suites

ID	KSP-FA05-RL07-R24
Title	Use Perfect Forward Secrecy
Description	Perfect Forward Secrecy must be used when setting up encrypted connections with any of the following protocols: - IPSEC (Internet Protocol Security) met Group 14 (or better) - SSH (Secure Shell) - TLS (Transport Layer Security for web traffic) - OTR (Off-The-Record messaging for instant messaging) Non-perfect forward secrecy protocols are allowed for legacy support and compatibility only. TLS cipher suite configuration should explicitly prefer ECDHE and DHE/EDH cipher suites above other cipher suites.
Relating document	http://en.wikipedia.org/wiki/Forward_secrecy KSP-FA05-TL02 - Cryptographic algorithms and cipher suites

ID	KSP-FA05-RL07-R25
Title	Use of multi-domain certificates
Description	Certificates must be scoped to only one application. The application may use multiple FQDNs (Fully Qualified Domain Names) to be identified. The FQDNs must share the same domain name. Example: • "www.kpn.com" and "kpn.com" can be combined • "www.kpn.com" and "kpninternational.com" cannot be combined • "reporting.kpn.com" and "www.kpn.com" and "kpn.com" may be combined in one certificate when the "reporting" hostname is explicitly part of the overall application.
Relating document	N/A

ID	KSP-FA05-RL07-R26
Title	Use of untrusted certificates
Description	The use of untrusted certificates is not allowed. Untrusted certificates are: - Self-signed certificates, i.e. certificates which have self-vetted and self-validated their own information and key material by signing itself. - Certificates signed by an untrusted, unknown or vendor supplied CA, i.e. certificates which have not been vetted and validated by an open or known process. - Certificates using key material not generated nor controlled by KPN. Exception: Exclusively for Puppet and VMWare VMCA it can provide and apply their own certificated. These certificates are explicitly scoped to the protocol used for inter-VMWare cluster or Puppet Master-Agent communication. Any other application for these certificates is prohibited.
Relating document	Requirements: KSP-FA05-RL07-R09 (Certificates) KSP-FA05-RL07-R10 (Use of certificates) KSP-FA05-RL07-R11 (Binding Certificates)

ID	KSP-FA05-RL07-R27
Title	Choosing safe curves for elliptic curve cryptography
Description	The use of safe elliptic curves is mandatory. Specific elliptic curves are regarded as safe after having passed (cryptographic) peer review. Applications can use the following safe curves: - M-221 (Curve2213) - E-222 - Curve1174 - Curve25519 - E-382 - M-383 - Curve383187 - Curve41417 - Ed448-Goldilocks - M-511 - E-521 Exception: If no safe curves are supported, the following elliptic curves are acceptable for usage: - P-256 - P-384 - P-521
Relating document	KSP-FA05-TL02 - Cryptographic algorithms and cipher suites SafeCurves: http://safecurves.cr.yp.to/

ID	KSP-FA05-RL07-R28
Title	Password hashing
Description	Passwords must be hashed and stored using known good salted password hashing methods. The following list of actions must be taken for each password from the service/tooling: • Use a good random salt, see KSP-FA05-RL07-R20 (Salt use) • Use a known good hash algorithm, see KSP-FA05-RL07-R18 (Hash Algorithms) • Use a random salt per password • In client/server scenarios, like web applications, always hash on the server side • To make cracking harder use key stretching to protect the passwords Exception for high-volume environments where key stretching is not applicable for performance reasons: use HMAC to protect the passwords with a key per password stored securely in an HSM solution.
Relating document	Requirements: KSP-FA05-RL07-R18 (Hash Algorithms) KSP-FA05-RL07-R19 (HMAC) KSP-FA05-RL07-R20 (Salt use) KSP-FA05-RL07-R29 (Key stretching algorithms)

ID	KSP-FA05-RL07-R29
Title	Key stretching algorithms
Description	Apply known good key-stretching algorithms: • PBKDF2, when FIPS certification or enterprise support on many platforms is required. Only use in combination with hash algorithms and a salt as mentioned in this document. ○ On mobile devices ▪ Minimum: 5.000 rounds ▪ Norm: 10.000 rounds ○ On servers and workstations: ▪ Minimum: 50.000 rounds ▪ Norm: 100.000 rounds • Scrypt, where resisting any/all hardware accelerated attacks is necessary but support isn’t. ○ On mobile devices ▪ Norm: N = 2^14, r = 8, p = 1 ○ On servers and workstations: ▪ Norm: N = 2^20,r = 8, p = 1 • Bcrypt, where PBKDF2 or scrypt support is not available ○ On mobile devices ▪ Norm: cost = 13 ○ On servers and workstations: ▪ Norm: cost = 16
Relating document	Requirements: KSP-FA05-RL07-R18 (Hash Algorithms) KSP-FA05-RL07-R20 (Salt use)

ID	KSP-FA05-RL07-R30
Title	Key destruction
Description	Cryptographic keys must be destroyed in such a way that restoration is impossible. This procedure must take platform specific properties into account, like removal of a file does not implicitly wipe the key from the disk nor does it implicitly nullify the data in memory.
Relating document	N/A