From: "Don Johnson" <djohnson@certicom.com>
To: kmscomments@nist.gov
Date: Sat, 27 Oct 2001 10:05:19 -0400
Subject: Don Johnson comments for 2nd NIST KMS workshop

NIST,
I offer the following comments for consideration at the 2nd KMS workshop. Please confirm receipt. As I cannot attend at this short notice, I hope these comments help. It is fine with me if you distribute these comments at the meeting. If any questions arise, please send email. If possible, I would appreciate responses, at least on my important comments. Thanks for your consideration,
Don Johnson, Certicom

My biggest concern is that I think no method defined in these proposals should be removed from consideration at this Nov. 2001 meeting as we lack:
1) Security attributes table for all schemes.
2) Performance summary table for all schemes. Things just cannot be compared until then. In fact, I can see a use for each scheme, but if someone wants to remove one, we at least should have all the info to discuss it.

On discussion issues from agenda:
1) Encryption should include X9.63 ECIES even if X9.44 RSA is not added yet.
2) KC is often provided by immediate use of the derived key. Specifying an explicit form of KC is useful for safety reasons and/or if the derived key is not known to be used immediately, for example, a key management server should do KC as a service to its clients. But KC in a small device may be costly overhead.
3) KC should be mandatory if the key is not known to be used immediately and the security attributes need to be solidified. For example, in a high-risk or high-value transaction it can be prudent to do KC and it should be required if the keys are not used immediately. KC should be mandatory for a key management server, this is part of the service it provides.
4) Symmetry is desirable as it simplifies calculations, but the security properties of the identify OR/AND function are unclear. I do not think alphabetical order is a good idea, this seems costly in a small device. Where did the identity OR/AND function come from? Too bad this was not discussed in ANSI X9. As the OR/AND function (even if it really is XOR/AND) does not allow recovery of the identities, it seems spoofable to some degree (as it cannot distinguish which identity had a '1' and which a '0' bit when these bit locations are OR/XOR/ANDed). It does do something and is better than not doing anything I guess. Better would be some method that allows recovery of the identities, then it would not be spoofable. This area needs more thought.
5) It is conservative design to add the identities as input to the KDF. There are existing systems that do not use this as it was not discussed as a requirement at ANSI X9 meetings. Perhaps an acceptable compromise would be to require it for high-risk or high-value transactions.
6) X9.42 and X9.63 DH/MQV assumes 2 parties, although DH extends to multiparty. Is this what is meant by the question? If so, we could ask X9 if this functionality is desired. Does this mean generating lots of keys for different applications? If so, this can be handled by identifying the use of the derived key as part of the input to KDF.
7) Entity authentication is the big attribute. Forward secrecy is useful for data confidentiality.

8) (a) Rough performance comparisons in number of basic crypto operations (point multiplication, modular exponentiation) allows basic performance cost comparisons between the various methods. This is a factor in deciding what to use along with security attributes achieved. I can help with this, ISO 15946-3 did this for ECC methods.
(b) I think some discussion differentiating high-risk and/or high value transactions from normal ones is appropriate. Things that are appropriate for high-security needs may be too costly for normal everyday use. I think ways to protect $50 can be simpler than ways to protect $5M. Short term tactical plans can need less assurance of security than long term strategic ones. This is just accepting reality. For example, for high stuff, I would suggest requiring AES as a starter.
(c) Given all the security requirements stated, some mention of the potential for side-effect attacks seems warranted, as this can unravel a secret/private key by monitoring energy, time, radio waves, etc.
(d) Wild stuff like Quantum Computing. At least current analysis and assessment should be stated, else it will be a question.
9) A chart with time on one axis and size (perhaps in log scale) on other axis could track the largest DL/ECDL/RSA composite factored over time and would be a useful plot to try to extrapolate from. It is what we know and the past is the best indicator of the future. This gives useful info to the reader to allow him/her to make decisions.

On Schemes Workshop document:
1) I think that completion of at least a comparison table for section 5 (Security Attributes) and performance summary (see above) is essential before evaluating the various methods. I can help with this, I suggest extracting the tables from X9.42 and X9.63 as a start.
2) Section 6.1 DP: What one needs for domain parameters is "assurance of validity", else ALL intended security attributes (including privacy of the private key) should be assumed to be void; however, it is not true that one needs to validate the DP oneself, this is just one option. This assurance can be provided by selecting the DP from a list published by a trusted organization (e.g., the NIST or SECG recommended curves), a CA generating the DP itself or having the CA validate the DP if generated by someone else or if all other options fail, you need to realize you need to validate the DP yourself. Also, for 6.1.3 ANYONE should be able to generate DP, as long as they are validated somehow.) It is easier to validate than generate DP so this could be a useful service even if not totally trusted.
3) Section 6.2.2 PKV: Here are my first thoughts on PKV, this likely will need further discussion. I agree that static public keys should (probably always) be validated, as this is a one-time cost and can be done by the CA as part of the public key certification process. Ephemeral public keys need to (at least) have a cofactor used in their calculation (ECC only, for DL the performance cost was considered too high and was therefore not specified in X9.42) or be validated. By using cofactor, Lim-Lee attacks on your private key are thwarted. The other concern that PKV addresses is use of an invalid key, perhaps from an inadvertent bit flip during generation. I agree that PKV for ephemeral keys may be a cost effective choice for high-risk or high-value transactions, but think this is appropriate for the security policy to decide when PKV is appropriate. The "other side" of this concern is that, especially for constrained devices, PKV can be costly, while cofactor is not very costly.
4) Section 6.2.3 Cryptoperiod: If you destroy the private key, you cannot recreate any calculation. Some systems allow archiving due to records retention requirements. Certainly the

private key should not be used after cryptoperiod expiry. This stuff is covered in more detail in key management guideline and so should point to it.
5) Section 6.3 Use of entity identifiers in KDF: This is conservative and binds the users to the calculation. To achieve this binding the requirement is that the value input to the KDF, whatever it is, can be juggled to determine the exact values of U and V, this is not achieved by the identity OR/AND function, although it is clear it does restrict something and so is better than not doing anything in terms of nailing things down.
6) Section 6.8 Key Confirmation: Often KC is not needed, as the derived key is used right away and any error will be detected. On the other hand, it does solidify security attributes and so is a conservative choice. Might be a requirement for high-security applications.
7) Section 6.9 item 2 cofactor: As implied above, I agree with mandating use of cofactor for both static and ephemeral keys.
8) Section 7: great organization, great job.
9) Section 7.3 Static-Static DH: I think prudence would indicate that time variant info should ALWAYS be required to be input to the KDF.
10) Section 9 key transport should include ECIES from X9.63.
11) Section 11 key recovery: Forward secrecy and key recovery are incompatible, essentially by definition. This could be mentioned.

Comments on Key Management Guideline:
1) Section 3.2.4 keysizes: It might be good to clarify exactly what is contained in each key and how long it is in toto, considering everything. An implementation will need more than 1024 bits to store a 1024-bit DSA key, for example. The keysize as commonly stated is not the amount of storage needed to store the key. At the least this should be mentioned.
2) Section 3.2.4 keysizes: Point to section 5.2.1 and 5.2.2 for keysize tables. Also, point to 5.2.3 on effects of breaking a key.
3) Section 4.3 key install: Often split knowledge is a requirement for manual install, so that no single installer knows the key.
4) Section 4.4.1.1 item d: The "/or" should be removed. "and" is correct as to intent.
5) Section 4.4.1.1 (last sentence): The user should realize that anyone that knows the private key can simulate him/her. This can affect non-repudiation as more than one user might sign, but it also affects encryption, as more than one user might decrypt. My suggestion is that the sentence be generalized. From my way of thinking, if I am acting as an agent for another party (say I work in a company), it is fine for the other party to generate my key(s), but if not, not. That is, it should always be OK to generate my own key, but only sometimes is it reasonable for someone else to generate my key. As a balance to that, perhaps a hack does not generate keys right and uses a weak seed, but then he suffers. This needs guidance. Also, it is reasonable that I might want to have a "marriage key" where 2 people might sign with it or a "family" key where any family member can sign, etc. This means that one of the group signed, but it may not be able to be determined exactly which one, but there is still non-repudiation to the group. This often reflects reality. If the president is not around, two VPs can sign or whatever. The verifier just needs to use the company verification key and knows the contract will not be repudiated.
6) Section 4.7.1.6 In fact, it should be considered an attack if a private key is used with more than one set of domain parameters. Decoupling is not allowed.
7) Section 4.7.3 backup Table 2: A signing key might be used just for integrity purposes and non-repudiation is not needed. Also, I simply may not want non-repudiation as a property and I

should not be forced to have it. There should be no concern with backing up such a key. I think seed for canonical generation of DP needs to have a separate line from a RNG seed. Seed by itself is confusing. Also, for testing purposes sometimes an RNG reseed is useful, but not for operational use. Also, I notice RNG keys, this might be the RNG seed in which case seed (by itself) should be clarified to be DPG seed.
8) Section 5.1.2: Key usage granularity should be allowed to be as fine as desired by the designer of the solution. This should be clarified. The KMS document sets the minimum.
9) Section 5.1.3 Cryptoperiods: Maximums can be calculated based on the information leakage assumed by amount of use. (I cannot calculate this, but could assist in reviewing.) Time is just an approximation. Also, TDES should be assumed to leak info after 2**32 block encrypts due to blocksize of 64 bits and the effect is probabilistic with a quadratic decay; the point is to stay well away from 2**32. For example, if I want less than a 1 in a million chance for an adversary to be able to know ANY info about my plaintext, I should stop at 2**27 encrypted TDES blocks, which is only 1G bytes. Not sure how to factor this in, but this is a factor in deciding to use AES as it has a 128 bit blocksize.
10) Section 5.1.3 *2nd one* (typo): DPV and PKV: Assurance of POP and PKV is appropriate for any public key, not just EC and DL keys. When RSA is added, there needs to be some discussion of RSA PKV. I can help with this. Also, note that DSA DP are more restrictive than X9.42 DH keys, in that q is set at a specific size (in relation to p) with DSA while with DH, it must be that size or larger, so note 9 is not strictly correct.
11) Section 5.1.7 Key recovery: To my way of thinking, key recovery is an ownership issue and this simplifies the analysis considerably. For example, if a company owns the data, it has a right to recover it.
12) Section 5.2.1 keysize table: This may be obvious, but when an entry is omitted, one needs to go to the higher entry. Also, as an example, it should be made clear to all users that a TDES key that was protected using ECC 160 bits only has 80 bits of effective security, not 112 bits. I think any system should state its claimed security level and then everything needs to be done at least that good or better.
13) Section 5.2.2 keysize time: The basic assumptions that were used to create this table should be given. As it is, this info is supplied "nekkid". The rationale is important as (at best) this table is a guestimate. The assumptions might turn out to be false and we need to know if this looks like it is happening and not just blindly follow the table. Note also that as the blocksize of TDES is only 64 bits, after 2**32 data blocks are encrypted, information leakage should be expected to occur, which might violate security requirements. The point is that this table seems to only consider keysize. The point is that use of AES can be indicated for other than just keysize reasons. Also, I think more guidance should be given on using at least AES-128 appropriate keysizes. There are many useful methods that might lose a few bits of key security; having some extra key bits to spend in this way is not inappropriate. Also, there should be some discussion of lucky guessing, not just exhaustion, this mainly affects symmetric keysize. I might want to use a larger key to achieve a larger assurance against the potential for lucky guessing. If I want less than a billion chance of a lucky guess and think 2**80 is doable, I really want 2**112 and so forth.

Issues at end of above:
1) No comment.

2) It is safest to prohibit this mixing. It should definitely be disallowed for high-risk or high value transactions.
3) I think there should be a stated keysize security level and all crypto in the system MUST operate at that level or above.
4) Verification tag size is different than keysize. There can be valid reasons to allow shorter tag sizes, but it should be documented why the smaller size is (thought to be) OK. But this takes thought and the tag size should not be allowed to be shorter for the only reason that that is the way it is. The safe default is to let it be determined by keysize security level.
5) My take on the TLS PRF is that it did not meet its stated goals. Let us not propagate this design by endorsing it without a LOT of evidence that it is acceptable. It is fixable so that the revised method would clearly meet the goals and have appropriate security. I have proposed such a fix to ANSI X9 but we have not yet had a meeting to discuss.

From: Francois.Rousseau@CSE-CST.GC.CA
To: kmscomments@nist.gov
Subject: Key Establishment Schemes Document vs PKCS #11
Date: Mon, 19 Nov 2001 11:51:18 -0500

Hi everyone,
I am sorry, but because of other commitments I could not attend your Second Key Management Workshop on November 1-2. I have since read the workshop version of the Key Establishment Schemes document, which is available on your Web site, and have found it very comprehensive. However, I am not sure if the issues raised below were discussed during this workshop. As the author of the X9.42 and ECC key derivation mechanisms, which were recently added in the latest version 2.11 of the PKCS #11 standard (ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-11/v211/), I have the following comments against this document:

a. Although a new mechanism was added under PKCS #11 v2.11 for the generation of Domain Parameters for the schemes specified in ANSI X9.42, PKCS #11 does not currently have a mechanism to generate the Domain Parameters for the schemes specified in ANSI X9.63 as mandated in this document. It was instead expected that most ECC implementations would use the Named Curves specified in FIPS 186-2 as the Domain Parameters for these ECC key derivation mechanisms.

b. PKCS #11 does not currently have a class of mechanisms to specifically validate Domain Parameters as mandated in this document.

c. PKCS #11 does not currently have a class of mechanisms to specifically validate Public Keys as mandated in this document.

d. Although a NULL Key Derivation Function (KDF) and the three KDFs specified for the schemes in ANSI X9.42 and X9.63 were added under PKCS #11 v2.11, PKCS #11 does not currently have the distinct KDF specified in this document.

e. Implementation Validation through a MacTag is only explicitly mentioned for the X9.42 key derivation mechanisms under PKCS #11 v2.11 and not for the ECC key derivation mechanisms since this subject was not discussed under ANSI X9.63. It could however easily be added.

f. PKCS #11 does not currently have a class of mechanisms to specifically support Key Confirmation as suggested in this document.

g. PKCS #11 v2.11 supports X9.63 key derivation mechanisms with the Modified Diffie-Hellman primitive and also with the Standard Diffie-Hellman primitive contrary to what is specified in this document since it was felt that Patent issues with the Modified Diffie-Hellman primitive, which uses co-factor multiplication, could possibly cause problems.

h. The Full Unified Model C(2,2,DH,EC), dhHybridOneFlow C(1,2,DH,FF) and 1-PassUnified Model C(1,2,DH,EC) schemes specified in this document are not explicitly supported under

PKCS #11 v2.11, however they could be implemented through multiple calls against the existing mechanisms currently specified under PKCS #11 v2.11.

i. I have since noted that the MQV mechanism parameters for both the X9.42 and the ECC key derivation mechanisms specified in PKCS #11 v2.11 have errors since they do not currently provide all the required keys to the key derivation mechanisms. I was planning to eventually a Technical Corrigendum to get this corrected.

Feel free to contact me if you have any question related with these
comments.
Best regards,
Francois

Francois Rousseau
IT Standards, Senior Advisor
Communications Security Establishment
francois.rousseau@cse-cst.gc.ca
(613) 991-8364
Edward Drake Building
1500 Bronson, Ottawa, Ontario, K1G 3Z4