Comments on SP 800-90A (2014)

Paul Lambert 2
Ashit Vora, Acumen Security 3
Brian Smithson, Ricoh Americas 4
Sonu Shankar, Cisco 5
Gen'ya Sakurai, Information-technology Promotion Agency, Japan 6
Michael Harris, Centers for Disease Control and Prevention 8
Bjoern Fay, NXT Semiconductors 9
Rene Struik 10
Paul Duplys, Bosch 13
Thomas Ryan, CGI IT Global Security Labs 14

From: Paul Lambert <paul@marvell.com>
Date: Monday, April 21, 2014 5:28 PM

Dear Sirs,

Thank you for the opportunity to review the draft NIST SP 800-90-A

In reviewing the definition of DRBG usage, it appears that all of the recommendations and suggesting testing lead developers to seeding the DRBG once (initial seed) and then running the DRBG for a long period. Where sources of entropy are available, it would be a much better design to recommend more continual mixing in of new entropy. If for some reason the initial seed can be discovered or predicated, adding additional entropy through the period of usage would mitigate such threats.

As an example of the misrepresentation of "reseeding", in Section 8.6 it describes reseeding as:

> "Reseeding is a means of restoring the secrecy of the output of the DRBG if a seed or the internal state becomes known."

The NIST 800-90-A document should more strongly recommend reseeding as an ongoing part of any implementation. The value of adding entropy to the deterministic process should be better described and the benefits of being less deterministic (adding entropy) should be documented.

Regards,

Paul A. Lambert
paul@nymbus.net

From: Ashit Vora <avora@acumensecurity.net>
Date: Thursday, April 24, 2014 4:03 PM

Hello,

Acumen Security has two comments/questions:

- Section 8.4 has been updated to include a SHALL statement limiting usage of random bits. The example provided is that of a DRBG instantiated to 128 bit security strength but being asked to provide random bits to generate a 256 bit AES key. In order to generate a 256 bit AES key, two calls to the DRBG will be required, there by providing 128 bits of security strength for each 128 random bits. Would this be acceptable or does the DRBG need to be instantiated to 256 bits of security strength?

- Can you clarify what is meant by this statement (Section 8.5) "For distributed DRBGs, each sub-boundary is the same as or is fully contained within a separate cryptographic module boundary." Does this mean for a distributed DRBG, the sub boundary has to be part of a FIPS validated cryptographic module boundary? If so for software modules that depend on HW entropy source (e.g. RDrand), do the HW entropy sources need to be contained within a FIPS validated crypto module?

Thank you!

Ashit Vora
Laboratory Director
Cell: +1 (213) 268-2593
Web: www.acumensecurity.net

From: "Brian Smithson" <bsmithson@ricohsv.com>
Date: 5/15/14 9:15 PM

While the requirement to test the generate function "at reasonable intervals defined by the implementer" has been removed from 11.3.3 on page 74, the requirement to document the selection of such intervals remains in 11.1 on page 72, 9th bullet item: "Document the periodic intervals at which health testing is performed for the generate function and provide a justification for the selected intervals (see Section 11.3.3). That bullet item should also be removed.

--
Regards,
Brian Smithson
CISSP, CISA, PMP, CSM
Senior Security Architect
Global Solutions Engineering
Solutions Development Center
Ricoh Americas
bsmithson@ricohsv.com
(408)346-4435
After May 25, 2014:
(408)610-3113
675 Campbell Technology Park, Suite 200, Campbell CA, 95008

From: Sonu Shankar <sonshank@cisco.com>
Date: Thursday, May 22, 2014 6:06 PM

1) Section 11.3.1 proposes modifications to the existing health test requirements. We would like some clarification on the value provided by this change and how it provides a security benefit to the DRBG implementation.

This is a very significant change as it would require updates to all existing software and hardware implementations alike. The delays associated with propagating this change throughout the development infrastructure can be severe, especially considering SoC integration and adoption delays owing to the fact that these requirements may not be achievable without new silicon. We believe it is therefore critical for NIST to also recommend a reasonable transition plan for these requirements, should they be deemed mandatory.

2) Section 8.6.5 describes what appears to be restrictions on where the entropy input can be obtained from. Cryptographic libraries in software only include the implementation at the algorithm level. While the entropy is a critical, mandatory aspect of the DRBG, the deterministic nature of software makes it a poor choice for entropy sources which are therefore typically based on hardware based noise sources (outside the software implementation boundary) that depend on unpredictable physical phenomena.

Some clarification is necessary here on software DRBG implementations from the perspective of cryptographic module boundaries as well as on the requirements associated with the “secure channel” to protect for entropy input transport.

Thanks,
Sonu Shankar

From: "g-sakura@ipa.go.jp" <g-sakura@ipa.go.jp>
Date: 5/23/14 12:26 AM

Dear Cryptographic Technology Group experts,

I have several comments on Draft SP 800-90A rev.1 (2nd draft) as written below.

<<1. Section 10.3.1>>
It is hard to see the ceiling function in step 2 of 10.3.1.
Please use Equation Editor.

<<2. Section 11.3.1>>
Table 4 is added in this draft, but it is not clearly expressed about the timing or order of health tests for Uninstantiate function in the table.

In 11.3.5, it is said that "the uninstantiate function shall be tested whenever other functions are tested".
This means that the other functions shall not be used unless the health tests for uninstantiate function has been passed successfully.

Instead of the text just below Table 4, Table 4 should list the timing or order of health tests for Uninstantiate function.
For example, case 1:
Instantiate function only -> Instantiate and Uninstantiate functions.

<<3. Section 11.3.2>>
In P.68, the third bullet, it is said that "If the instantiate function for the DRBG resides in the same (sub)boundary as the DRBG's instantiate function, but there is no reseed function for that DRBG, proceed with instantiate-function testing (see Section 11.3.3)."
The text is hard to understand in the two senses:
- Are there "nested" instantiate functions?
- Section 11.3.3 is "Testing the Generate Function", while the text refers 11.3.3 as instantiate-function testing.

<<4. Section 11.3.3>>
The case 2 is defined as
"Immediately following the reseed-function testing for that DRBG when the reseed function and generate function reside in the same (sub)boundary."
In step 4 of 9.3.1, the generate function conditionally calls the reseed function. Therefore the following requirement should be added as one bullet for case 2:
If the reseed function for the DRBG resides in the same (sub)boundary as the DRBG's generate function, proceed with reseed function testing (see Section 11.3.4).

<<5. Section 11.3.3>>
In P.69, the third bullet for case 3, it is said that
"Otherwise, discontinue testing, indicating that the DRBG passed the instantiate-function test".
However, this is the section for testing the generating function. Therefore the "instantiate-function test" should be replaced by "generate-function test".

<<6. Section 11.3.4>>
In P.70, the third bullet for case 2, it is said that
" If the instantiate function for the DRBG resides in the same (sub)boundary as the DRBG's reseed function, proceed with instantiate-function testing (see Section 11.3.3)."
However, the case 2 is declared that there is no instantiate function for that DRBG within the same (sub)boundary as the reseed function.
Therefore the third bullet is inconsistent with the case definition and should be removed.

<<7. Annex D>>
FIPS 186 should be removed because there is no reference to FIPS 186(-3 or -4) in this document.

<<8. Annex D>>
References to SP 800-57 Part 1 and SP 800-67 (Rev.1) should be updated to the latest versions.

Sincerely,

Gen'ya

Gen'ya SAKURAI

IT Security Center (ISEC),
Technology Headquarters,
Information-technology Promotion Agency, JAPAN

E-mail:g-sakura@ipa.go.jp
Tel:+81-3-5978-7545 FAX:+81-3-5978-7548 URL:http://www.ipa.go.jp

From: <Harris>, "Michael W. (CDC/OCOO/OCIO)" <fnb0@cdc.gov>
Date: Friday, May 23, 2014 7:55 AM

CDC has no comments to provide on the *draft NIST Special Publication 800-90A Revision 1, Recommendation for Random Number Generation Using Deterministic Random Bit Generators*.

Thank you for the opportunity to review and comment.

Michael W. Harris, CISSP, Information Technology Specialist, Office of the Chief Information Security Officer (OCISO), Centers for Disease Control and Prevention (CDC) 770.488.8052, Cell: 770.283.9589, E-mail: fnb0@cdc.gov

From: Bjoern Fay <bjoern.fay@nxp.com>
Date: Friday, May 23, 2014 10:51 AM

Dear NIST,

Here some comments regarding the SP 800-90 A.

- In section 10.2.1 CTR_DRBG on page 52 it still says that for TDEA the keys shall be handled as 64-bit blocks containing 56 bits of key and 8 bits of parity. But then in the rest of the document you seem to handle it as 56-bit blocks. Actually it would be easier and more secure to implement it really using 64-bit blocks, so for updating the state you should be taking 64-blocks and use them as new keys (ignoring the parity bits). This would mean that seedlen would be 4*64 = 256 bits (even if 24 bits are ignored) instead of only 3*56 + 64 = 232 bits.
- It is unclear why you would need a full entropy source if you want to implement a CTR_DRBG without derivation function. In section 10.2.1 CTR_DRBG on pages 50/51 table 3 it says that the required minimum entropy for instantiate and reseed is the security strength, which is (at least) one block size (outlen) smaller than the seedlen. So even in the worst case (AES-256) it would be enough if you have only 0.7 bits of entropy per seed bit to fulfill this requirement. And after one update you also have distributed this entropy over the complete state. So I would suggest to remove this requirement for needing a full entropy source or derivation function (for anything), you just need to provide enough entropy as stated in the table, however you do this.

Best regards
Björn Fay

Dr. Björn Fay Cryptographer, Business Un
Semiconductors Germany GmbH Stresem annallee 101, 22529
Germany -Mail:+49 40 5613 1415, Fax:+49 40 5613 6
bjoern.fay@nxp.com, http://www.nxp.com Geschäftsführung: Ruediger Stroh (Vors.),
Frans Scheper, Kurt Sievers, Erik Just-Wartiainen, Achim Kempe /
Aufsichtsratsvorsitzender: Gernot Fiedler / Sitz: Hamburg / Registergericht: H
HRB 84 865 The information contained in this message is confidential and may be legally privileged. The message is intended solely for the addressee(s). If you are not the intended recipient, you are hereby notified that any use, dissemination, or reproduction is strictly prohibited and may be unlawful. If you are not the intended recipient, please contact the sender by return e-mail and destroy all copies of the original message.
Unless otherwise recorded in a written agreement, all sales transactions by NXP Semiconductors are subject to our general terms and conditions of commercial sale. These are published at: www.nxp.com/profile/terms/index.html

From: Rene Struik <rstruik.ext@gmail.com>
Date: Friday, May 23, 2014 6:27 PM

Dear NIST team:

I have reviewed NIST SP 800-90A, Rev. 2, as posted on April 21, 2014.

The main modification compared to the previous version that was opened for public comment is the removal of the Dual-EC-DRBG generator.

It is a pity that NIST, in its current draft, is suggesting to remove the Dual-EC-DRBG algorithm, rather applying one of the simple fixes I suggested in my comments submitted during the comment period ending November 6, 2013 (see pp. 17-36 of http://csrc.nist.gov/publications/drafts/800-90/draft_sp800_90a_comments_received.pdf). I would strongly suggest NIST to reconsider this and apply one of the simple fixes I suggested instead, with the following motivation:
a) Picking any of the five algorithms I suggested to "fix" the Dual-EC-DRBG algorithm (Algorithm A-E of my comments) demonstrably fixes any concern re a potential backdoor of this generator and re potential distinguishability of outputs.
b) Picking any of two last algorithms (algorithms D-E) I suggested would provide a DRBG in the NIST specification set that potentially offers strong resistance against quantum cryptographic attacks, should these ever become a reality: these algorithms potentially result in m/2-bits of security, if m-bit curves are used, with those two algorithms. In contrast, all remaining DRBG algorithms in the NIST specification set seem to offer at most m/3-bit quantum security for, e.g., an m-bit hash function.
c) While one might argue that any of the five algorithms I suggested would still result in a relatively slow DRNG, this fact was of course already long known when the original specification was under development. Moreover, algorithms picked are not simply picked based on security and speed alone, but also on implementation footprint. Any of the five algorithms I suggested as "fix" offers the prospect of fielding a cryptographic system without the requirement to implement a hash function. For constrained devices that have hardware support for scalar multiplication, not having to implement, e.g., a hash function based DRBG could be highly attractive, notwithstanding the trade-off of implementation cost and speed. NIST should not make speed trade-offs for others, since it cannot reliably predict optimum trade-offs in potential deployment scenarios it may not have insight in; it simply should provide a small, but rich enough set of potential constructs that it can stand behind from a security confidence perspective.

As very specific recommendation to NIST, I would like to suggest NIST to adopt Algorithm E I suggested in my earlier comments: it provides all the security features one may wish for, provides likely quantum crypto resilience, does away with the need to store any public keys, and offers potential for significant speed-ups, since one only performs scalar multiplications with respect to the base point.

NIST's motivation for scrapping the Dual-EC-DRBG seems to be "lack of public confidence" in the algorithm. Since the potential concerns that may have triggered this

lack of public confidence have been known for roughly eight years (well before the original SP 800-90A specification was published), I would suggest that it behooves NIST not to just raise a white flag, show more confidence in its own capabilities, and have the courage to apply the simple fixes that demonstrably take away those concerns and offer potential benefits on other axes than speed alone.

I am happy to help NIST with any of these confidence building efforts. For NIST's information: the algorithms I suggested for "fixing" the Dual-EC-DRBG will be peer-reviewed by the technical committee of a well-respected cryptographic conference. I will gladly share the outcome of this review with NIST staff, once available.

Best regards, Rene

-------- Original Message --------

Subject: Comments on NIST SP 800-90A, Rev1, and NIST SP 800-90B
Date: Thu, 07 Nov 2013 08:11:46 -0500
From: Rene Struik <rstruik.ext@gmail.com>
To: RBG_Comments@nist.gov

Dear NIST team:

I have the following comments on the NIST SP 800-90A draft:

1. Some of the formulae seem missing, e.g., p. 66, Step 1 of EC-DRBG Generate process.

2. I have two recommendations re the EC-DRBG generator:

a) Include the output of the verifiably random pick for G and Q in the specification (e.g., in Appendix A.2.1).
b) Change the EC-DRBG random number generator more fundamentally, so as to
(1) remove reliance on the public key Q;
(2) lower distinguishability of the output bit string;
(3) tighten security reductions;
(4) provide potential resilience against quantum cryptographic attacks (should these become a long-term threat).
c) For RNGs based on number-theoretic problems (such as EC-DRBG), it would be beneficial to produce outputs
in the underlying field, rather than bit strings, while specifying post-processing that would map elements of
the underlying field to binary strings.

For more details re #2 above, please see the attached summary document of how this could potentially be realized in detail.

I would be happy to discuss the technical proposal re "tweaks" to EC-DRBG in more detail with you.

I have the following comments on the NIST SP 800-90B draft:

1) At the review last year (2012) I expressed concern that the language of the draft would effectively rule out using
physically unclonable functions as a source of true randomness. This would be highly unfortunate, since this could
prove to become quite an attractive low-cost true randomness source with high min-entropy. Since the draft did not
change since last year, I thought I should re-emphasize this point.

2) I would like to make you aware of the recently published paper
Key Derivation Without Entropy Waste (Yevgeniy Dodis, Krzysztof Pietrzak, Daniel Wichs, IACR ePrint 2013-708).
It seems worth investigating whether the bounds on entropy-loss in that paper would justify loosening the
requirements on how much min-entropy a true randomness source would need to have to justify derived keys
to be considered fully random (it seems one would gain a factor almost 2x in efficiency this way).

This relates to my comment from the review last year re

Section 6.2, p. 24, 2nd item: It is not entirely clear where the "2*n* full entropy requirement" comes from. Assuming the input to the conditioning function (with output size *n* bits) to be "full entropy", this suggests that this requires at least 2*n*-bit entropy. Again, this suggests that – with the use of an only the SHA-x family of hash functions as approved conditioning functions – one is stuck with noise sources with 2*n*> 320 of entropy and cannot use a noise source offering *n*=128 bits of min-entropy in case on throws in a conditioning function. This needs some more explanation.

Best regards, Rene

PS: apologies for forwarding you my comments a few hours late.

PPS: the attached document is in dvi format (LaTeX viewer). I tried to attach the corresponding pdf version, but that document was 6 MB, a factor 200x bigger.

--
email: rstruik.ext@gmail.com | Skype: rstruik
cell: +1 (647) 867-5658 | US: +1 (415) 690-7363

From: "Duplys Paul (CR/AEA3)" <paul.duplys@de.bosch.com>
Date: Monday, May 26, 2014 5:54 AM

To whom it may concern,

In Section 11.3 “Health Testing” pertaining to the run-time self-test of the DRBG, it is not specified/recommended when and how often the self-test (i.e. the recommended known-answer-test) shall be run. In particular in case of embedded systems I am concerned about Fault Injection attacks that allow an attacker to render an (P)RNG completely useless. Have you thought about this attack vector? It seems that in order to counter this type of attacks the self-test must be executed at least in regular intervals. Better yet, one should have appropriate countermeasures built in into the system/design.

Best regards,
Paul Duplys

Mit freundlichen Grüßen / Best regards

Paul Duplys

Robert Bosch GmbH
System Software (CR/AEA3)
Postfach 30 02 40
70442 Stuttgart
GERMANY
www.bosch.com

Tel. +49(711)811-8083
Fax +49(711)811-5185813
paul.duplys@de.bosch.com

From: Thomas Ryan
Date: May 23, 2014

Hello,

Thank-you for the opportunity to comment on this draft NIST Special Publication. Please find the CGI ITSETF's (an NVLAP-accredited laboratory and a candidate NIAP Common Criteria Laboratory) comments on NIST DRAFT SP 800-90A revision1 in response to the public comment period on the NIST Computer Security Publications website.

CGI Comment #1: When reading the notification on the Cryptographic Toolkit website (http://csrc.nist.gov/groups/ST/toolkit/800-90A-RFC.html) it seems that the removal of the Dual_EC DRBG was the only change to the SP. Upon review of the updated document we were alarmed to learn that this is not the case – there are other changes proposed in this revision that will require changes to 800-90A DRBG implementations that are currently conformant in order to meet the new requirements proposed in revision 1. These changes include (but are not limited to) how the nonce must now be generated (Section 8.6.7), how the DRBG health tests must now be performed (Section 11.3) and Section 8.4 now requires that the security strength provided (by the DRBG) matches the number of bits requested for each generate request.

Many engineers and developers may not review the other changes to this draft SP as they have not been explicitly mentioned in the NIST announcement. In our opinion it is highly likely many will review the announcement and assume that the removal of the Dual_EC DRBG is the only change to the SP. This means NIST will not receive a sufficient amount of feedback and/or peer review on the other proposed changes to this revision. No reason or rationale has been provided for these changes – what is the reason NIST feels that these changes must be implemented in revision 1? What risks or issues do these proposed changes mitigate? For these reasons we request that NIST only remove the Dual_EC DRBG in revision 1 of SP 800-90A and implement the other changes in a future revision.

CGI Comment #2: The use of "Substantive" can be highly subjective. We request that instead of a substantive list summarizing the changes there should be an explicit list or a complete "diff" of all the changes made. Even the addition or removal of subtle wording can change the interpretation of a key definition or statement when it comes to technical publications – especially where there is conformance testing performed according to its requirements. In our opinion if all of the changes have not been clearly listed it makes it difficult to perform a proper review and provide feedback.

Page 112 of Appendix F lists the "Substantive" changes made to revision 2. Our concern with this listing is it does not explicitly list all changes made to the document. This list does not include a new shall statement added to Section 7 or the changes made to Section 8.4 – we are concerned that these and other changes will be potentially be overlooked as

a result. This is another reason why we request that NIST only remove the Dual_EC DRBG in revision 1 of SP 800-90A and implement the other changes in a future revision.

CGI Comment #3: We believe the DRBG mechanism and respective cryptographic module boundaries (both logical and physical) in revision 1 need to be better defined. Below are some specific examples:

· Section 8.6.5 states: "*The entropy input shall be obtained from within a cryptographic-module boundary, which could be a cryptographic module that does not contain an instantiate or reseed function; if the entropy input is obtained from a cryptographic module that does not contain the instantiate or reseed function with which the entropy input is used, then there shall be a secure channel to transport the entropy input between the cryptographic-module boundaries*."

· Does this Section refer to the module's physical boundary in the above statement? If not, from a FIPS 140-2 validation perspective we (as a CSTL) are unsure how to apply this statement to a software module that implements a DRBG. It may be very challenging for labs and vendors to design and validate software cryptographic libraries that conform to this SP. Typically, for these modules entropy is obtained from outside of the module's cryptographic logical boundary from device drivers like /dev/random or 3rd party products which implement a hardware noise source with an architecture similar to the one in Figure 4. The entropy source, however in our experience resides inside of a GPC or an appliance which is the module's physical boundary. In these situations the module receives a caveat stating "*no assurance of the minimum strength of externally provided entropy*".

· Section 8.6.7 states : "*a nonce shall be obtained within a cryptographic module*". Again, does this refer to the module's logical boundary or physical boundary (GPC or appliance casing)? Similar to the example above if the nonce must be obtained inside of the logical cryptographic boundary it presents challenges to software modules (specifically libraries). This clarification is also required for the distributed DRBG scenario referenced in SP 800-90C – if the nonce is generated outside of a crypto module's logical boundary but inside of the physical boundary is a secure channel still required? If so, we would appreciate a rationale for why NIST feels this is appropriate and practical.

· Section 7 states: "*A DRBG shall implement an approved DRBG mechanism from SP 800-90A and at least one approved source of entropy input, and may include additional optional sources, including sources for nonces, personalization strings, and additional input*" however the diagram in Figure 1 still appears to have the entropy input (source or sources) outside of the DRBG Mechanism boundary (marked by the dotted black line). Please clarify the DRBG mechanism and cryptographic module boundaries as they relate to entropy inputs and the nonce.

CGI Comment #4: NIST SP 800-90B is still in draft and there is currently no cryptographic algorithm validation testing available for Approved Entropy Sources or

Approved NRBGs. It is unclear what type of validation testing NIST require in order to demonstrate conformance for both Common Criteria evaluation and FIPS 140-2 validation purposes.

· Can you elaborate would NIST require CAVP testing alone to demonstrate the Entropy source is Approved? or would a full FIPS 140-2 module validation of the entropy source in order to consider it Approved?

CGI Comment #5: Section 8.4 adds a shall to the following statement "*These pseudorandom bits **shall** not be used for any application that requires a higher security strength than the DRBG is instantiated to support*". Section 8.4 appears to also include several new statements including:

"*Any security strength may be requested (up to a maximum of 256 bits), but the DRBG will only be instantiated to one of the four security strengths above, depending on the DRBG implementation. A requested security strength that is below the 112-bit security strength or is between two of the four security strengths will be instantiated to the next highest strength (e.g., a requested security strength of 80 bits will result in an instantiation at the 112-bit security strength).*"

"*When an instantiation is used for multiple purposes, the minimum entropy requirement for each purpose must be considered. The DRBG needs to be instantiated for the highest security strength required. For example, if one purpose requires a security strength of 112 bits, and another purpose requires a security strength of 256 bits, then the DRBG needs to be instantiated to support the 256-bit security strength.*"

We are concerned that these new and updated statements to Section 8.4 are not indicated in the Substantive list of changes on page 112. Again, we believe that the proposed changes to this SP have not been sufficiently identified and this revision will not undergo sufficient review as a result. We respectfully ask that NIST provide a full exhaustive list of all the proposed changes to SP 800-90A along with a rationale for the proposed changes then and go through another iteration of public comments. We again request that NIST only remove the Dual_EC DRBG in this proposed revision.

CGI Comment #6: In the description of the "Secure channel" in Section 8.5 it is unclear as to what protections mechanisms are required for the secure channel demonstrated in figure 4 in order to have a distributed DRBG. The definition and description of a secure channel in Section 4 seem to indicate that a physical method of protection can be used.

· Can NIST please clarify how replay protection and mutual authentication (in the definition of the secure channel) to be handled in use cases without using cryptography?

· Can NIST please clarify how this this relates to the Section 8.5 security strength requirement that the security provided by the secure channel be equal to that of a consuming application?

We hope that you will take our feedback into consideration. We would be happy to discuss the above questions and concerns at your convenience.

Regards,

Ryan Thomas on behalf of CGI ITSETF

Ryan Thomas CISA, CISSP
Program Manager
CGI Global IT Security Labs - Canada
T: 613-234-2155
1410 Blair Place, 7th floor
www.linkedin.com/in/ryathomas