Ron was wrong, Whit is right

Arjen K. Lenstra[1], James P. Hughes[2],
Maxime Augier[1], Joppe W. Bos[1], Thorsten Kleinjung[1], and Christophe Wachter[1]

[1] EPFL IC LACAL, Station 14, CH-1015 Lausanne, Switzerland
[2] Self, Palo Alto, CA, USA

Abstract. We performed a sanity check of public keys collected on the web. Our main goal was to test the validity of the assumption that different random choices are made each time keys are generated. We found that the vast majority of public keys work as intended. A more disconcerting finding is that two out of every one thousand RSA moduli that we collected offer no security. Our conclusion is that the validity of the assumption is questionable and that generating keys in the real world for "multiple-secrets" cryptosystems such as RSA is significantly riskier than for "single-secret" ones such as ElGamal or (EC)DSA which are based on Diffie-Hellman.
Keywords: Sanity check, RSA, 99.8% security, ElGamal, DSA, ECDSA, (batch) factoring, discrete logarithm, Euclidean algorithm, seeding random number generators, K_9.

1 Introduction

Various studies have been conducted to assess the state of the current public key infrastructure, with a focus on X.509 certificates (cf. [4]). Key generation standards for RSA (cf. [22]) have been analysed and found to be satisfactory in [18]. In [12] and [26] (and the references therein) several problems have been identified that are mostly related to the way certificates are used. In this paper we complement previous studies by concentrating on computational and randomness properties of actual public keys, issues that are usually taken for granted. Compared to the collection of certificates considered in [12], where shared RSA moduli are "not very frequent", we found a much higher fraction of duplicates. More worrisome is that among the 4.7 million distinct 1024-bit RSA moduli that we had originally collected, more than 12500 have a single prime factor in common. That this happens may be crypto-folklore, but it was new to us, and it does not seem to be a disappearing trend: in our current collection[3] of 7.1 million 1024-bit RSA moduli, almost 27000 are vulnerable and 2048-bit RSA moduli are affected as well. When exploited, it could affect the expectation of security that the public key infrastructure is intended to achieve.

We summarize our findings, referring to later sections for details. We collected as many openly accessible public keys as possible from the web, while avoiding activities that our system administrators may have frowned upon. The resulting set of 11.7 million public keys contains 6.4 million distinct RSA moduli. The remainder is almost evenly split between ElGamal keys (cf. [11]) and DSA keys (cf. [25]), plus a single ECDSA key (cf. [25]). The frequency of keys blacklisted due to the Debian OpenSSL vulnerability (cf. [28]) is comparable to [12]. All keys were checked for consistency such as compositeness, primality, and (sub)group membership tests. As the sheer number of keys and their provenance precluded extensive cryptanalysis and the sensibility thereof, a modest search for obvious weaknesses per key was carried out as well. These efforts resulted in a small number of inconsistent or weak keys.

[3] Except for this sentence and Appendix A everything in this paper is based on our original dataset.

A tacit and crucial assumption underlying the security of the public key infrastructure is that during key setup previous random choices are not repeated. In [12,18] public key properties are considered but this issue is not addressed, with [18] nevertheless concluding that

> The entropy of the output distribution [of standardized RSA key generation] is always almost maximal, ... and the outputs are hard to factor if factoring in general is hard.

We do not question the validity of this conclusion, but found that it can only be valid if each output is considered in isolation. When combined some outputs are easy to factor because the above assumption sometimes fails. Among the ElGamal and DSA keys we found a few duplicates with unrelated owners. This is a concern because, if these owners find out, they may breach each other's security. It pales, however, compared to the situation with RSA. Of 6.6 million distinct X.509 certificates and PGP keys (cf. [1]) containing RSA moduli, 0.27 million (4%) share their RSA modulus, often involving unrelated parties. Of 6.4 million distinct RSA moduli, 71052 (1.1%) occur more than once, some of them thousands of times. Duplication of keys is thus more frequent in our collection than in the one from [12]. This leads to the same concern as for ElGamal and DSA, but on a wider scale.

99.8% Security. More seriously, we stumbled upon 12720 different 1024-bit RSA moduli that offer no security. Their secret keys are accessible to anyone who takes the trouble to redo our work. Assuming access to the public key collection, this is straightforward compared to more traditional ways to retrieve RSA secret keys (cf. [5,15]). Information on the affected X.509 certificates and PGP keys is given in the full version of this paper, cf. below. Overall, over the data we collected 1024-bit RSA provides 99.8% security at best (but see Appendix A).

Two's company, three's a crowd. Identifying primes with vertices, and (regular) RSA moduli[4] with edges connecting their prime factors, the graph of the moduli contained in c distinct certificates consists, in an ideal world, of c disjoint connected components each consisting of two vertices joined by a single edge, for a total of $2c$ vertices and c edges. The actual graph contains 1995 disjoint connected components on three vertices or more. Most of these components are depth one trees, rooted at a common factor and most with two or three leaves. These depth one trees may be caused by poor random number generator seeding. We also encountered seven components that we find harder to explain, the most remarkable one being a K_9, a complete graph on nine vertices. Figure 1 depicts a simplified sketch of the situation and how it may evolve.

Academic research into the strength of cryptographic systems is not supposed to cross the line of proof-of-concept constructions or solving representative challenges (cf. [9], [13], [23]). The purpose is timely identification of developments that could affect current security solutions and to propose adequate upgrades. Publication of results undermining the security of live keys is uncommon and inappropriate, unless all affected parties have been notified. In the present case, observing the above phenomena on lab-generated test data would not be convincing and would not work either: tens of millions of thus generated RSA moduli turned out to behave as expected based on the above assumption. Therefore limited to live data, our intention was to confirm the assumption, expecting at worst a very small number of counterexamples and affected owners to be notified. The quagmire of vulnerabilities that we

[4] A regular RSA modulus is the product of two different prime numbers. We have not encountered any proper RSA modulus that is not regular.

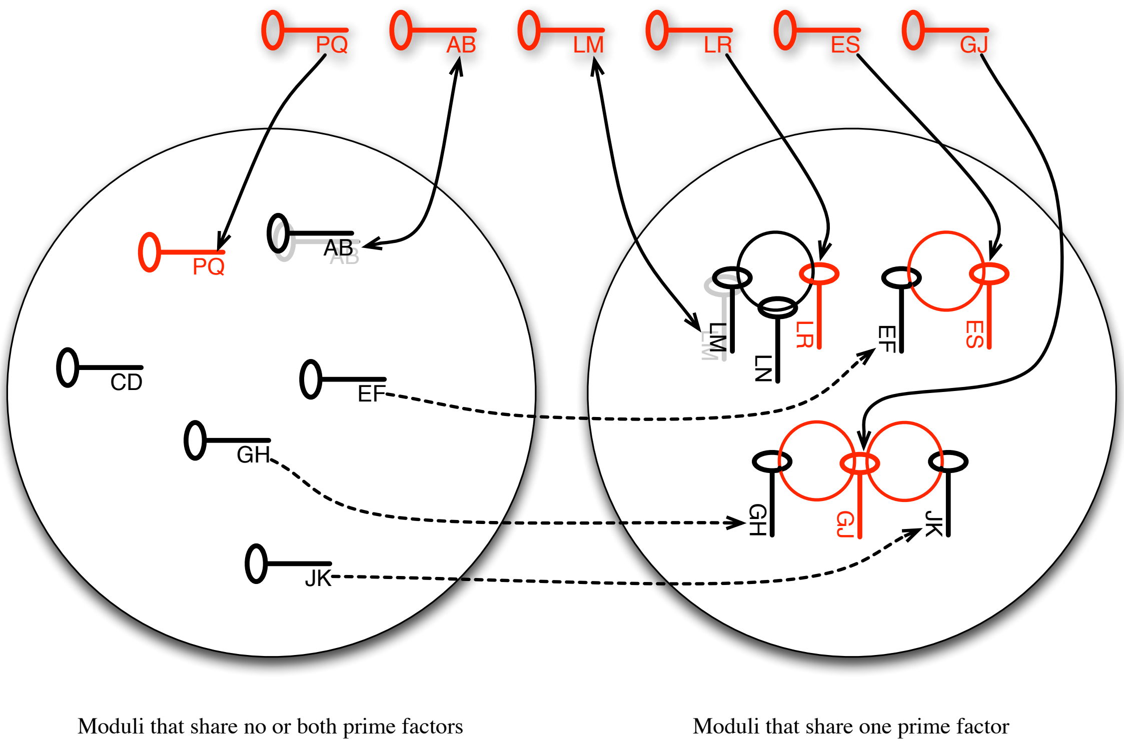

Fig. 1. An existing collection of seven (black) keys is extended with six (red) new keys, where capital letters play the role of (matching) large primes. Initially, keys AB, CD, EF, GH, and JK on the left are secure and keys LM and LN on the right are openly insecure in the same keyring due to the common factor L. New key PQ is secure and appended to the secure list on the left. New key AB duplicates key AB on the left, making both insecure to each other but not to anyone else. New key LM duplicates a key already known to be in the openly insecure group, while key LR results in a new openly insecure modulus on that keyring. Key ES removes known good key EF from the secure keys on the left, resulting in a new openly insecure group on the right consisting of keys EF and ES. Even if the owner of ES now knows that he is insecure and destroys the key, this information can be used by any owners involved to determine the factors of key EF. Key GJ removes two known good keys, GH and JK, from the list of secure keys on the left to form an insecure double keyring on the right (cf. Figure 5 in Section 3). All example keyrings, and many more, occur in the real world. Note that a key that has been dragged from left to right will never be able to return.

waded into, makes it infeasible to properly inform everyone involved, though we made a best effort to inform the larger parties and contacted all email addresses recommended (such as `ssl-survey@eff.org`[5]) or specified in valid affected certificates. The fact that most certificates do not contain adequate contact information limited our options. Our decision to make our findings public, despite our inability to directly notify everyone involved, was a judgment call. As was the case with the Debian OpenSSL vulnerability, we believe that publication is preferable to attempts to cover up weaknesses. There is a risk, but people desiring to exploit the current situation will have to redo our work, both the data collection and the calculation, as we have put our data under custody. We hope that people who know how to do the computation quickly will show good judgment. Combined with the EFF's forthcoming attitude, we

[5] The EFF initiated a new centralized SSL Observatory scan in January 2012, after which they will notify certification authorities of remaining vulnerable keys. See Appendix A for some results.

decided for publication. It will be appreciated, however, that no true evidence of our findings can be presented.

We have briefly considered creating a service allowing people to check if their RSA modulus is affected. The consideration, however, that a positive answer would expose at least one other party convinced us that this would not be a good idea – unless all parties in a component (cf. Section 3) simultaneously request information, in which case they can easily figure it out by themselves. For this same reason not to expose anyone to relatively straightforward searches and simpler calculations than we had to carry out, there will be two versions of this paper until the affected keys have been revoked: in this version some details are suppressed.

Cryptosystems such as RSA that require, during key-setup, multiple secrets are more affected by the apparent difficulty to generate proper random values than systems such as Diffie-Hellman (cf. [8]), ElGamal, and (EC)DSA that require a single secret. For either type of system identical keys can be exploited only by parties that can be identified by searching for the owner of the duplicate. But for the former partially identical choices allow any malcreant to commit fraud. For RSA multiple secrets can be avoided, for instance by using moduli pq for k-bit primes p chosen such that $q = \lceil (2^{2k-1} + p - (2^{2k-1} \bmod p))/p \rceil$ is prime (cf. [14]).

Section 2 presents our data collection efforts, Section 3 describes the counts and calculations that we performed for the RSA-related data. For completeness, Section 4 does the same for the ElGamal, DSA, and ECDSA data. Section 5 summarizes our findings.

2 Data collection

Before the data from [10] was generally available, we started collecting public keys from a wide variety of sources, assisted by colleagues and students. This resulted in almost 5.5 million PGP keys and fewer than 0.1 million X.509 certificates. The latter got a boost with [10] and, to a smaller extent, the data from [12]. We did not engage in web crawling, extensive `ssh`-session monitoring, or other data collection activities that may be perceived as intrusive or aggressive. Thus, far more data can be collected than we did (see also Appendix A).

Skipping a description of our attempts to agree on a sufficiently uniform and accessible representation of the data, by November 2011 the counts had settled as follows: 6 185 372 distinct X.509 certificates (most from the EFF SSL repository, 43 from other sources), and 5 481 332 PGP keys, for a total of at most 11 666 704 public keys. Of the X.509 certificates 6 185 230 are labeled to contain an RSA (modulus, exponent) pair with 141 DSA public keys and a single ECDSA point on the NIST standardized curve `secp384r1` (cf. [3, Section 2.8], [25]). Of the certificates 47.6% have an expiration date later than 2011. About 77.7% of the certifying signatures use SHA1 or better (5287×SHA256, 24×SHA384, 525×SHA512), 22.3% use MD5 (with 122×MD2, 30×GOST, 14×MD4, and 9×RIPEMD160). Both requirements, expiration later than 2011 and usage of SHA1-or-2, are met by 33.4% of the certificates.

Of the PGP keys 2 546 752 (46.5%) are labeled as ElGamal public keys, 2 536 959 (46.3%) as DSA public keys, the other 397 621 (7.3%) as RSA public keys. PGP keys have no expiration dates or hashes. All public keys were further analysed as described below.

Table 1. Most frequently occurring RSA public exponents.

X.509		PGP		Combined	
e	%	e	%	e	%
65537	98.4921	65537	48.8501	65537	95.4933
17	0.7633	17	39.5027	17	3.1035
3	0.3772	41	7.5727	41	0.4574
35	0.1410	19	2.4774	3	0.3578
5	0.1176	257	0.3872	19	0.1506
7	0.0631	23	0.2212	35	0.1339
11	0.0220	11	0.1755	5	0.1111
47	0.0101	3	0.0565	7	0.0596
13	0.0042	21	0.0512	11	0.0313
65535	0.0011	$2^{127}+3$	0.0248	257	0.0241
other	0.0083	other	0.6807	other	0.0774

3 RSA

In this section we present the results of various counts and tests that we conducted on the data labeled as RSA public keys. An RSA public key is a pair (n, e) of a supposedly hard to factor RSA modulus n and a public exponent e. The corresponding secret key is the integer d such that $de \equiv 1 \bmod \varphi(n)$ or, equivalently, the factorization of n.

Public exponents. Table 1 lists the ten most frequent public exponents along with their percentage of occurrence for the RSA keys in the X.509 certificates, the PGP keys, and when combined. Except for eight times $e = 1$ and two even e-values among the PGP RSA keys, there is no reason to suspect that the e-values are not functional. Two e-values were found that, due to their size and random appearance, may correspond to a short secret exponent (we have not investigated this). The public exponents are not further regarded below.

Debian moduli. Two of the n-values, a 1024 and a 2048-bit one each occurring once, were discarded because they could be fully factored using the data from [19] (cf. Debian OpenSSL vulnerability in [28]). A further 30097 n-values (0.48%, with 21459 distinct ones) were found to be blacklisted (cf. [24]), but as their factors were not easily available they were kept.

Shared moduli. We partition the set of 6 185 228 X.509 certificates into *clusters*, where certificates in the same cluster contain the same RSA modulus. There is a considerable number of clusters containing two or more certificates, each of which could be a security issue; clusters consisting of one certificate, on the other hand, are the good cases. As depicted in Figure 2, there is one cluster with 16489 certificates (the blue circle on the x-axis), followed by clusters of sizes 8366, 6351, 5055, 3586, 3538, 2645, for a total of 14 clusters with more than a thousand certificates (the red and blue circles on the x-axis; the 5055 share a blacklisted modulus, with no other blacklisted modulus occurring more than seven times). On the other side of the scale the number of good cases is 5 918 499 (the single green circle on the y-axis), with 58913 and 7108 clusters consisting of two and three certificates, respectively. It follows that $6\,185\,228 - 5\,918\,499 = 266\,729$ X.509 certificates (4.3%) contain an RSA modulus that is shared with another X.509 certificate. With 71024 clusters containing two or more certificates it follows that there are $5\,918\,499 + 71024 = 5\,989\,523$ different n-values.

Looking at the owners with shared n-values among the relevant set of 266 729 X.509 certificates, many of the duplications are re-certifications or other types of unsuspicious recycling

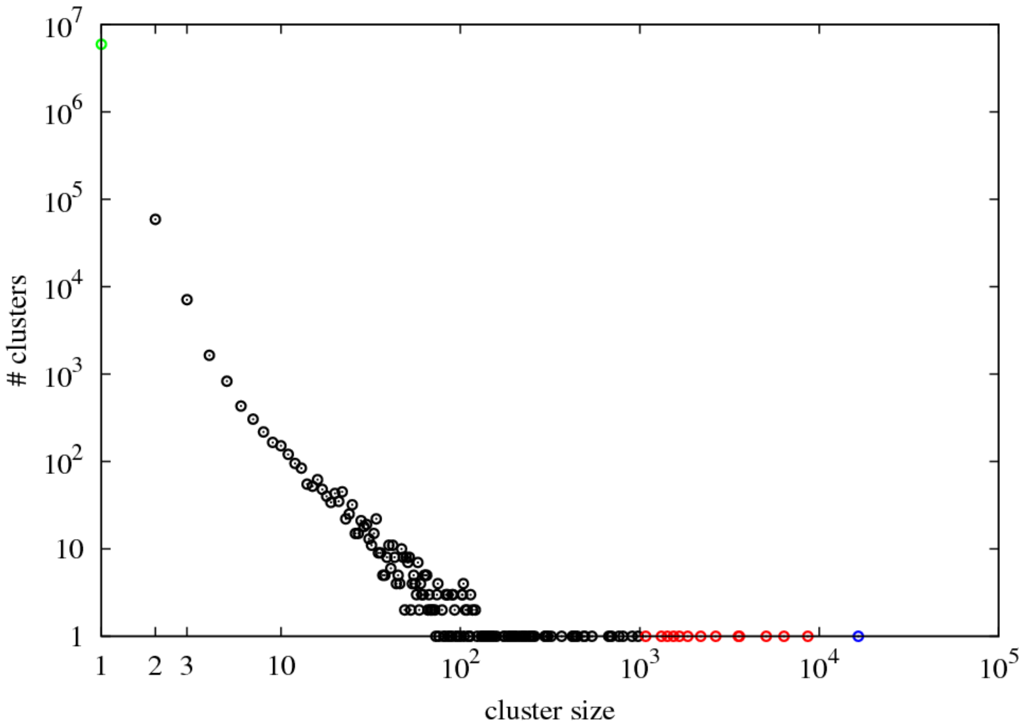

Fig. 2. Number of certificate clusters as a function of the cluster-size.

of the same key material by its supposedly legal owner. It also becomes clear that any single owner may come in many different guises. On the other hand, there are also many instances where an n-value is shared among seemingly unrelated owners. Distinguishing intentionally shared keys from other duplications (which are prone to fraud) is not straightforward, and is not facilitated by the volume of data we are dealing with (as 266 729 cases have to be considered). We leave it as a subject for further investigation into this “fuzzy” recognition problem to come up with good insights, useful information, and reliable counts.

The 397 621 PGP RSA keys share their moduli to a much smaller extent: one n-value occurs five times and 27 occur twice. Overall, 28 n-values occur more than once, for a total of 59 occurrences. The n-value that occurs in five PGP keys also occurs twice among the X.509 certificates, and all seven occurrences refer to the same owner. For some of the other 27 multiple occurrences of n-values unique ownership of the RSA keys was harder to assess.

Distinct moduli. As seen above, we extracted 5 989 523 different n-values from the X.509 certificates. Similarly, $397\,621 - 59 + 28 = 397\,590$ of the PGP n-values are unique. Joining the two sets resulted in 6 386 984 distinct values, with the 129 n-values contained in both sets occurring in 204 X.509 certificates and in 137 PGP keys: as mentioned, some PGP keys are X.509-certified as well (though we have not tried to establish unique or conflicting ownerships, as this already proved to be infeasible for keys shared just among X.509 certificates). In order not to make it easier to re-derive our results, most information below refers to the joined set of unique values, not distinguishing between X.509 and PGP ones.

Modulus sizes. The cumulative sizes of the moduli in the set of 6 386 984 n-values are depicted in Figure 3. Although 512-bit and 768-bit RSA moduli were factored in 1999 (cf. [2]) and 2009 (cf. [13]), respectively, 1.6% of the n-values have 512 bits (with 0.01% of size 384 and smallest size 374 occurring once) and 0.8% of size 768. Those moduli are weak, but still offer marginal security. A large number of the 512-bit ones were certified after the year 2000 and even until a few years ago. With 73.9% the most common size is 1024 bits, followed by 2048 bits with 21.7%. Sizes 3072, 4096, and 8192 contribute 0.04%, 1.5%, and 0.01%, respectively. The largest size is 16384 bits, of which there are 181 (0.003%).

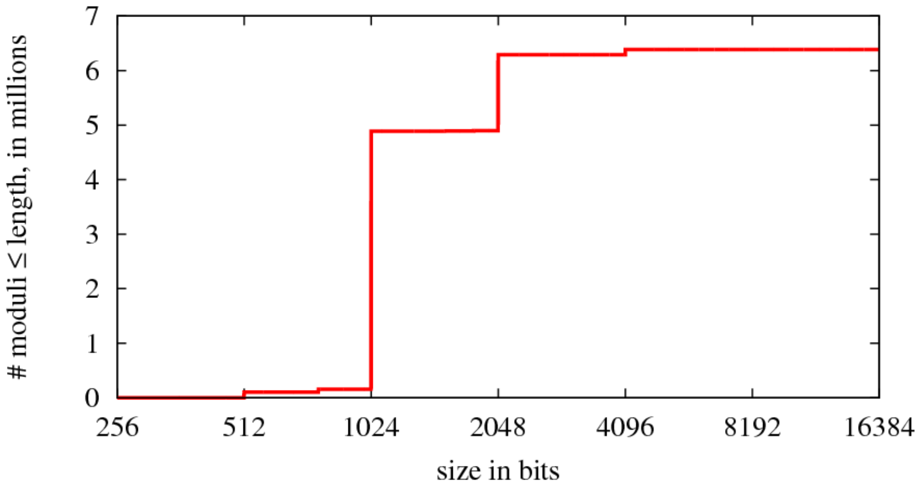

Fig. 3. Cumulative number of modulus sizes for RSA.

Primality, small factors, and other tests. Two of the unique n-values are prime, 171 have a factor $< 2^{24}$ (with 68 even n-values) after removal of which six cofactors are prime. About 25% of the remaining 165 composites were fully factored after a modest search for small factors using the implementation from [29] of the elliptic curve method (ECM, cf. [16]), some of the others may indeed be hard to factor and could, in principle, serve as RSA modulus. Nevertheless, these 173 n-values do not comply with the standards for the generation of RSA moduli (cf. [18]) and they were discarded. Nine cases are probably due to copy-and-paste errors, as eight proper moduli were found that differed from the wrong ones in a few hexadecimal positions (two distinct wrong moduli match up with the same correct one).

Fermat's factorization method, which works if two factors are close together, did not produce any factors. In particular we found no moduli as reported by Mike Wiener [27] ($n = pq$ with p prime and q the least prime greater than p, and thus with p the largest prime $\leq [\sqrt{n}]$).

Moduli with shared factors. Moduli that share one prime factor result in complete loss of security for all moduli involved. We discuss the results based on the graphs spawned by the moduli and shared factors. If different users make different choices during key setup, the graph associated to c distinct n-values (cf. Introduction) would consist of c connected components[6],

[6] Two distinct vertices are in the same connected component if and only if they are connected by a path consisting of edges in the graph.

Table 2. The T-column indicates the number of depth one trees with ℓ leaves. The bold entry corresponds to the depth one tree depicted in Figure 4.

ℓ	T	ℓ	T	ℓ	T	ℓ	T	ℓ	T	ℓ	T	ℓ	T	ℓ	T
2	1200	8	30	14	5	20	3	27	1	42	1	65	1	**4627**	**1**
3	345	9	15	15	5	21	3	32	1	44	2	92	1		
4	139	10	11	16	3	22	3	33	1	46	1	121	1		
5	75	11	12	17	3	23	1	35	2	59	1	151	1		
6	52	12	6	18	2	24	2	36	2	61	1	348	1		
7	39	13	6	19	3	26	1	37	1	63	3	379	1		

each consisting of a single edge connecting two unidentified – and supposedly unidentifiable – primes. This turned out not to be the case: it took a matter of hours on a single core to find 1995 connected components that each consists of at least two edges[7].

Of the 1995 connected components, 1988 are depth one trees[8]. Of those 1200 have two leaves (i.e., 1200 pairs of n-values, each with a distinct prime factor in common), 345 three leaves, etc., up to a single one with 4627 leaves (i.e., 4627 n-values all with the same prime factor in common). Table 2 lists for each number of leaves ℓ how often each tree-size occurs.

Six of the other seven connected components contain four vertices and three edges, but are not depth one trees. Each of these six components thus consists of a "central" n-value that has a factor in common with each of two other, co-prime n-values. The remaining connected component is the most intriguing – or suspicious – as it is a complete graph on nine vertices (K_9): nine primes, each of whose $\binom{9}{2} = 36$ pairwise products occurs as n-value.

Denoting the primes identified with the vertices of the graph by p_1, p_2, ... (using an ordering naturally implied by our representation), Figures 4 and 5 depict the largest depth one tree and one of the six four-vertex components (all six look alike), respectively.

Any two n-values associated to edges in the same depth one tree can be factored. Two n-values associated to other edges can be factored if the edges are adjacent (i.e., share a vertex), or one finds a path connecting them. For non-adjacent edges in the same connected component from Figure 5 that is the unique central edge, for edges in the K_9 many paths are possible. All required edges are in our set of n-values.

Affected RSA moduli and certificates. Overall, this led to the factorization of 12720 n-values of 1024 bits. More details, also about affected X.509 certificates and PGP keys, are given in the full version of this paper.

Discussion. Generation of a regular RSA modulus consists of finding two random prime numbers. This must be done in such a way that these primes were not selected by anyone else before. The probability not to regenerate a prime is commensurate with the security level if NIST's recommendation [25, page 53] is followed to use a random seed of bit-length twice the intended security level. Clearly, this recommendation is not always followed.

[7] Much larger datasets can be handled without trouble. We chose not to describe the details of our calculation or how well it scales – sapienti sat. On a 1.8GHz i7 processor the straightforward approach would require about ten core-years and would not scale well. Inclusion of the p and q-values from Section 4 and the primes from [19] related to the Debian OpenSSL vulnerability [28] did not produce additional results.

[8] A depth one tree has no cycles and contains one *root* vertex with edges leading to all other vertices, as in Figure 4.

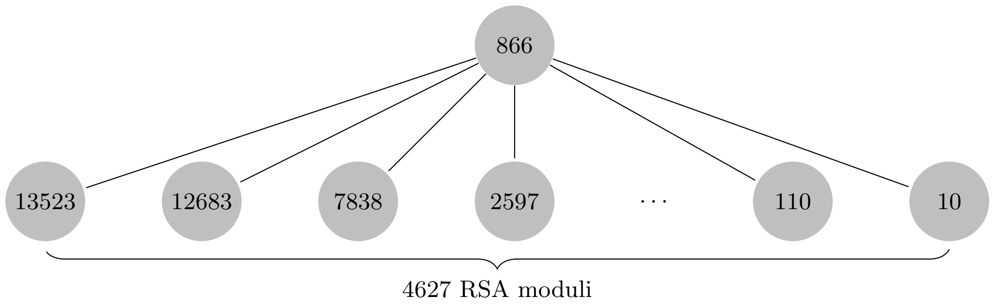

Fig. 4. The largest depth one tree found, on 4628 vertices, with the 512-bit prime p_{866} as root and leaves p_{13523}, p_{12683}, ..., p_{10}.

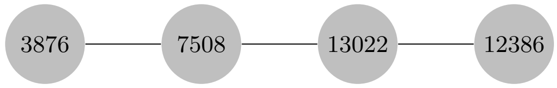

Fig. 5. One of the six connected components consisting of four vertices.

Irrespective of the way primes are selected (additive/sieving methods or methods using fresh random bits for each attempted prime selection), a variety of obvious scenarios is conceivable where poor initial seeding may lead to mishaps, with duplicate keys a consequence if no "fresh" local entropy is used at all. If the latter is used, the outcome may be worse: for instance, a not-properly-random first choice may be prime right away (the probability that this happens is inversely proportional to the length of the prime, and thus non-negligible) and miss its chance to profit from local entropy to become unique. But local entropy may lead to a second prime that is unique, and thus a vulnerable modulus.

The above may, to some extent, explain the occurrence of duplicate RSA moduli and depth one trees. But we cannot explain the relative frequencies and appearance of these mishaps yet. Neither do we understand how connected components in Figure 5 and the K_9 may be expected to appear. No correlation between certification time and vulnerability of keys was detected.

As mentioned in the introduction, avoiding two random choices during RSA modulus generation is straightforward. But the resulting moduli may have other, as yet unpublished weaknesses (we are not aware of serious ones). It is better to make sure that cryptographic keys are generated only after proper initialization of the source of randomness.

4 ElGamal, DSA, and ECDSA

In this section we present the results of various counts and tests that we conducted on the data labeled as ElGamal, DSA, or ECDSA public keys. In neither collection did we find any of the numbers from [19] (cf. Debian OpenSSL vulnerability [28]).

4.1 ElGamal

An ElGamal public key consists of a triple (p, g, y) where p is prime, g is a generator of the multiplicative group $(\mathbf{Z}/p\mathbf{Z})^*$ or a subgroup thereof of small index, and y is an element of $\langle g \rangle$. The secret key is an integer $x \in \{0, 1, \ldots, p-2\}$ with $g^x = y$.

Correct ElGamal keys. Among the PGP keys, 2 546 752 are labeled as ElGamal public keys. Three are incomplete and were discarded. Of the remaining triples 82 contain a composite p-value, resulting in 2 546 667 triples with correct p-values. Almost half (38) of the wrong p-values share a pattern with 65.6% of the p-values in the correct ElGamal keys, cf. below.

Restricting to the triples (p, g, y) with prime p-values, a triple is a correct ElGamal public key if $y \in \langle g \rangle$. To verify this the order of g, and thus the factorization of $p - 1$, is needed. This is easy for *safe primes* (i.e., primes p for which $(p-1)/2$ is prime), but may be hard otherwise. The order of g could be established for 70.8% of the triples (65.6% with safe primes, 5.2% with primes p for which $(p-1)/(2m)$ is prime and $m > 1$ has only small factors) and could reasonably be guessed for the other 29.2% (almost all with primes p for which $(p-1)/2$ is composite but has no small factors). For at least 16.4% of the ElGamal keys the g-values do not generate $(\mathbf{Z}/p\mathbf{Z})^*$. This led to 33 failed membership tests $y \in \langle g \rangle$, i.e., an insignificant 0.001% of the triples. Note that if $y \in \langle g \rangle$ a secret key exists; it does not follow that the owner knows it. A handful of triples were identified with peculiar y-values for which it is doubtful if a secret key is known to anyone.

Shared ElGamal keys. Six of the ElGamal keys occur twice: two keys with two unrelated owners each, and four keys occurring twice but with the same owner.

ElGamal key sizes. Figure 6 depicts the cumulative p-sizes in the set of 2 546 628 correct ElGamal keys. There are 1437 different p-sizes, ranging from thrice 256 bits to nine times 16384 and once 20000. Most frequent are 2048 bits (69.3%), 1024 (11.2%), 4096 (10.8%) and 3072 (5.8%) followed by 1536 (1.3%), 1792 (0.7%), 768 (0.4%), and 1025 (0.04%).

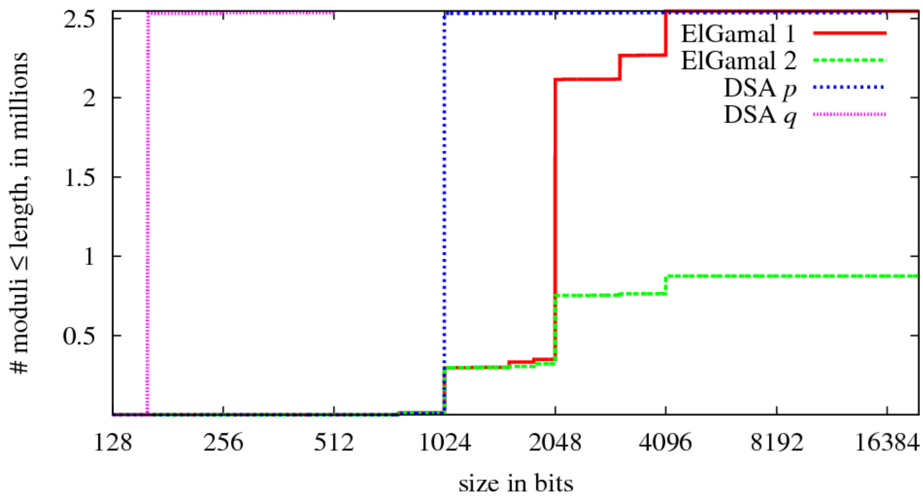

Fig. 6. Cumulative numbers of p and q-sizes in ElGamal and DSA public keys, indicated by "ElGamal 1", "DSA p", and "DSA q"; cumulative sizes of the distinct ElGamal p-values is indicated by "ElGamal 2".

Shared primes, generators. Primes and generators may be shared. Among the 2 546 628 distinct ElGamal keys 876 202 distinct p-values (and distinct (p, g)-pairs) occur. Despite this high duplication rate, only 93 distinct p-values occur more than once. The four most frequent p-values are "similar". Let $p(x, L)$ denote the least safe prime $\geq x \bmod 2^L$. There is an integer v such that $p(v, L)$ for L-values 2048, 4096, 3072, 1536 occurs as p-value in 52.4%, 6.5%, 5.6%, and 1% of the ElGamal keys, respectively ($p(v, 1024)$ occurs twice, $p(v+2^{510}, 512)$ once, and as noted above parts of v also occur in incorrect ElGamal keys). We suspect that these p-values, of different sizes, were generated using similar software and identical random seeding (if any), and from the least significant bit up to the most significant one.

All $p(., L)$-values use $g = 2$, which for $L = 2048$ generates $(\mathbf{Z}/p\mathbf{Z})^*$, but for the others an index two subgroup thereof. Overall, $g = 2$ occurs most frequently (70.8%), followed by $g = 5$ (19.5%), 6 (4.7%), 7 (1.9%), 11 (1.3%), and 13 (0.9%), with a total of 76 distinct g-values. No g-values were found that do not have a large order, but for at least 9.6% of the distinct (p, g)-pairs the g-values do not generate $(\mathbf{Z}/p\mathbf{Z})^*$. We can only give a lower bound because, as pointed out above, we failed to find *any* prime factor of 29.2% of the $(p-1)/2$-values in the ElGamal keys (which turns out to be 87.7% of the distinct $(p-1)/2$-values in the set of distinct p-values). Thus, we cannot be certain that the corresponding generators were properly chosen; consistent failure of all ECM factoring attempts of these numbers suggests, however, that they were well chosen.

Among the distinct ElGamal keys, all y-values are distinct, which is as expected because distinct (p, g)-pairs have negligible probability to lead to the same y-value (and identical (p, g)-pairs with identical y-values have already been identified and removed). The secret keys, however, may still be the same. But as there is no way to tell if that is the case (for distinct (p, g)-pairs) there is no serious security risk even if they are.

4.2 DSA

A DSA public key is a four-tuple (p, q, g, y) where p and q are primes with q dividing $p - 1$, the element g generates an order q subgroup of the multiplicative group $(\mathbf{Z}/p\mathbf{Z})^*$, and y is an element of $\langle g \rangle$. The secret key is the integer $x \in \{0, 1, \ldots, q-1\}$ with $g^x = y$.

Correct DSA keys. Among the PGP keys and X.509 certificates, 2 536 959 and 141 four-tuples, respectively, are labeled as DSA keys. All four-tuples were first checked for correctness, casting them aside at the first test they failed. The tests were conducted in the following order: T1: primality of p; T2: primality of q; T3: divisibility of $p-1$ by q; T4: order of g equals q; and T5: order of y equals q. An insignificant 0.002% (66) of the PGP four-tuples are incorrect with failures 12×T1, 2×T2, 10×T4, and 42×T5 (where T2 failed twice for the same q-value, as it occurred twice). The X.509 DSA four-tuples passed all tests. Some of the failures may be due to transcription errors, as they occur in four-tuples that differ from correct ones in a few hexadecimal positions.

Shared DSA keys. The remaining 2 536 893 PGP DSA keys contain very few duplicates: one key occurs thrice (with possibly double ownership) and two keys occur twice each (each with single ownership), resulting in a total of 2 536 889 distinct PGP DSA keys. Although all 141 X.509 DSA keys are distinct, 95 of them are also among the PGP DSA keys, resulting in a total of $2\,536\,889 + 141 - 95 = 2\,536\,935$ DSA keys. We have not checked ownerships of these 95 duplicate DSA keys.

DSA key sizes. The cumulative p and q-sizes in the set of 2 536 935 DSA keys are depicted in Figure 6. There are nine different q-sizes: all except 0.2% (5012) are 160, with 256, 224, and 232 the most frequent exceptions occurring 4016, 702, and 249 times, respectively. The smallest and largest q-sizes are 160 and 512, the latter with seven occurrences. With 78 different sizes the variation among p-sizes is larger, though nowhere close to the variation among ElGamal p-sizes. All except 0.6% (15457) of the p-sizes are 1024, with 768, 2048, 3072 and 512 the most frequent exceptions with 9733, 3529, 1468 and 519 occurrences, respectively. The smallest and largest p-sizes are 512 and 16384, the latter with a single occurrence.

Shared primes, generators. Distinct DSA keys may contain identical p, q, or g values. In total 2 535 074 distinct p-values occur, with 2 535 037 distinct primes occurring once, 22 occuring twice, five occurring thrice, etc., up to a prime occurring 969 times (note the difference with ElGamal). Not surprisingly (but not necessarily, as the same q-value may give rise to many different p-values), the overall counts and numbers of occurrences are the same for the distinct q-values and the distinct (p, q)-pairs. Although the generator also allows considerable variation, the number of distinct (p, q, g)-triples is the same too. For all except 265 of the unique (p, q, g)-triples, the generator g equals $2^{(p-1)/q}$. We have not been able to determine how the other 265 generators were chosen.

The y-values are all distinct among the distinct DSA keys – given that shared keys were already removed, identical y-values would have been odd indeed. The same remark as above applies concerning identical secret keys.

4.3 ECDSA

The only interesting fact we can report about ECDSA is the surprisingly small number of certificates encountered that contain an ECDSA key (namely, just one), and the small number of certificates signed by ECDSA (one self-signed and a handful of RSA keys). As long as one subscribes to the notion of a standardized curve over a finite field of prime cardinality of a special form, as opposed to a randomly but properly chosen curve over a non-special prime field (cf. [17]), there is nothing wrong with the curve parameters `secp384r1`. It offers (in "theory") about 192 bits of security which makes it, security-wise, comparable to 8000-bit RSA moduli (n) and ElGamal or DSA finite field sizes (p), and 384-bit DSA subgroup sizes (q).

4.4 ElGamal and (EC)DSA.

Not surprisingly, the intersection of the sets of p-values for ElGamal and for DSA is empty. We have not tried hard to retrieve any of the secret exponents, i.e., (for ElGamal and DSA) x-values such that $g^x = y$, but have checked that none is less than 2^{12} in absolute value.

Random nonces in ElGamal and (EC)DSA. Unlike RSA, during signature generation ElGamal and (EC)DSA require a random nonce that should be entirely unpredictable (cf. [11,20,21]). Given what we have witnessed for RSA (cf. Section 3) it is hoped that, by the time these nonces are generated, poor seeding is compensated for by enough local entropy. We are not aware of any studies that verify whether or not the nonces are properly chosen (with the notable exception of [6]).

Discussion. Both for ElGamal and DSA a small number of keys were identified that are shared among unrelated parties. This may be a security concern. Furthermore, there were some ill-formatted keys that cannot be expected to work and that should be of insignificant security concern. From the point of view of this paper, the main security concern using ElGamal and (EC)DSA is the generation of the random nonce. Although our data may allow us to look into this issue, we have no immediate plans to do so.

5 Conclusion

We checked the computational properties of millions of public keys that we collected on the web. The majority does not seem to suffer from obvious weaknesses and can be expected to provide the expected level of security. We found that on the order of 0.003% of public keys is incorrect, which does not seem to be unacceptable. We were surprised, however, by the extent to which public keys are shared among unrelated parties. For ElGamal and DSA sharing is rare, but for RSA the frequency of sharing may be a cause for concern. What surprised us most is that many thousands of 1024-bit RSA moduli, including thousands that are contained in still valid X.509 certificates, offer no security at all. This may indicate that proper seeding of random number generators is still a problematic issue (see also Appendix A).

The lack of sophistication of our methods and findings make it hard for us to believe that what we have presented is new, in particular to agencies and parties that are known for their curiosity in such matters. It may shed new light on NIST's 1991 decision to adopt DSA as digital signature standard as opposed to RSA, back then a "public controversy" (cf. [7]); but note the well-known nonce-randomness concerns for ElGamal and (EC)DSA (cf. Section 4.4) and what happens if the nonce is not properly used (cf. [6]).

Factoring one 1024-bit RSA modulus would be historic. Factoring 12720 such moduli is a statistic. The former is still out of reach for the academic community (but anticipated). The latter comes as an unwelcome warning that underscores the difficulty of key generation in the real world.

Acknowledgements

We thank Peter Eckersley from EFF for his invaluable assistance. We gratefully acknowledge key collection and other efforts by Benne de Weger, Philippe Joye, Paul Leyland, Yann Schoenberger, Deian Stefan, Fabien Willemin, Maryam Zargari and others that need to remain anonymous. The first author appreciates the legal advice from Mrs. Chardonnens. James P. Hughes participated in this project in his individual capacity without sponsorship of his employer or any other party. This work was supported by the Swiss National Science Foundation under grant number 200020-132160.

References

1. J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer. OpenPGP message format. RFC 4880, 2007.
2. S. Cavallar, B. Dodson, A. K. Lenstra, W. M. Lioen, P. L. Montgomery, B. Murphy, H. te Riele, K. Aardal, J. Gilchrist, G. Guillerm, P. C. Leyland, J. Marchand, F. Morain, A. Muffett, C. Putnam, C. Putnam, and P. Zimmermann. Factorization of a 512-bit RSA modulus. In B. Preneel, editor, *Eurocrypt 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 1–18. Springer, Heidelberg, 2000.

3. Certicom Research. Standards for efficient cryptography 2: Recommended elliptic curve domain parameters. Standard SEC2, Certicom, 2000.
4. D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280, 2008.
5. D. Coppersmith. Modifications to the number field sieve. *Journal of Cryptology*, 6(3):169–180, 1993.
6. Darkmirage. PS3 completely cracked, 2011. `http://www.darkmirage.com/2011/01/06/ps3-completely-cracked/`.
7. Y. Desmedt, P. Landrock, A. K. Lenstra, K. S. McCurley, A. M. Odlyzko, R. A. Rueppel, and M. E. Smid. The Eurocrypt '92 controversial issue: Trapdoor primes and moduli (panel). In R. A. Rueppel, editor, *Eurocrypt '92*, volume 658 of *Lecture Notes in Computer Science*, pages 194–199. Springer, 1993.
8. W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.
9. Electronic Frontier Foundation. *Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design*. O'Reilly Media, 1998.
10. Electronic Frontier Foundation. EFF SSL Observatory. `https://www.eff.org/observatory`, 2010.
11. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. Blakley and D. Chaum, editors, *Crypto 1984*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18. Springer, Heidelberg, 1985.
12. R. Holz, L. Braun, N. Kammenhuber, and G. Carle. The SSL landscape: a thorough analysis of the x.509 PKI using active and passive measurements. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, IMC '11, pages 427–444. ACM, 2011.
13. T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik, H. te Riele, A. Timofeev, and P. Zimmermann. Factorization of a 768-bit RSA modulus. In T. Rabin, editor, *Crypto 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 333–350. Springer, Heidelberg, 2010.
14. A. K. Lenstra. Generating RSA moduli with a predetermined portion. In K. Ohta and D. Pei, editors, *Asiacrypt '98*, volume 1514 of *Lecture Notes in Computer Science*, pages 1–10. Springer, 1998.
15. A. K. Lenstra and J. Hendrik W. Lenstra, editors. *The development of the number field sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1993.
16. H. W. Lenstra Jr. Factoring integers with elliptic curves. *Annals of Mathematics*, 126(3):649–673, 1987.
17. M. Lochter and J. Merkle. Elliptic curve cryptography (ECC) brainpool standard curves and curve generation. RFC 5639, 2010.
18. D. Loebenberger and M. Nüsken. Analyzing standards for RSA integers. In A. Nitaj and D. Pointcheval, editors, *Africacrypt '11*, volume 6737 of *Lecture Notes in Computer Science*, pages 260–277. Springer, 2011.
19. H. D. Moore. Debian OpenSSL Predictable PRNG Toys. See `http://digitaloffense.net/tools/debian-openssl/`, 2008.
20. P. Q. Nguyen and I. Shparlinski. The insecurity of the digital signature algorithm with partially known nonces. *Journal of Cryptology*, 15(3):151–176, 2002.
21. P. Q. Nguyen and I. Shparlinski. The insecurity of the elliptic curve digital signature algorithm with partially known nonces. *Design, Codes Cryptography*, 30(2):201–217, 2003.
22. R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.
23. M. Stevens, A. Sotirov, J. Appelbaum, A. Lenstra, D. Molnar, D. A. Osvik, and B. de Weger. Short chosen-prefix collisions for MD5 and the creation of a rogue CA certificate. In S. Halevi, editor, *Crypto 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 55–69. Springer, Heidelberg, 2009.
24. J. Tomášek et al. Blacklisted moduli. See `http://mirror.switch.ch/ftp/mirror/debian/pool/main/o/openssl-blacklist/` and `http://pocitace.tomasek.cz/debian-randomness/index.html`.
25. U.S. Department of Commerce/National Institute of Standards and Technology. Digital Signature Standard (DSS). FIPS-186-3, 2009. `http://csrc.nist.gov/publications/fips/fips186-3/fips_186-3.pdf`.
26. N. Vratonjic, J. Freudiger, V. Bindschaedler, and J.-P. Hubaux. The inconvenient truth about web certificates. In *The Workshop on Economics of Information Security (WEIS)*, 2011.
27. M. J. Wiener. Personal communication, 1992.
28. S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage. When private keys are public: results from the 2008 debian OpenSSL vulnerability. In A. Feldmann and L. Mathy, editors, *Internet Measurement Conference*, pages 15–27. ACM, 2009.
29. P. Zimmermann et al. GMP-ECM (elliptic curve method for integer factorization). `https://gforge.inria.fr/projects/ecm/`, 2012.

A Analysis of the data of the new EFF SSL Observatory scan

During the weekend of February 12, 2012, the EFF kindly shared with us the results of their new scan containing 7.2 million distinct X.509 certificates (up from 6.2 million). We report on our brief analysis of the new data.

In the new collection the number of unique 1024-bit RSA moduli was reduced, compared to the original dataset, from 4.7M to 3.7M. More than five thousand of the affected 1024-bit moduli that we had identified is no longer present. On the other hand, 13019 of the new 1024-bit moduli turned out to be affected: the new collection by itself contains 20251 distinct vulnerable 1024-bit moduli, involving 23130 prime numbers (with more than six thousand old primes gone, but 14836 new primes found). As depicted by the green graph in Figure 7, compared to the original dataset the number of 2048-bit RSA moduli increased from 1.4 to 3.2 million. Ten of the 2048-bit RSA moduli offer no security, two of which occur in (new) X.509 certificates that have not expired and use SHA1.

There is no reason, however, to just consider the new dataset, as more vulnerable moduli may be found by merging the original and new datasets. This led to 11.4 million distinct RSA moduli, of which 7.1 million have at most 1024 bits (as depicted by the blue graph in Figure 7). Analysis of the combined set is underway. Looking at the 7.1 million "short" ones, 26444 vulnerable 1024-bit RSA moduli were detected, 509 of 512 bits, and two 768-bit ones (one in the original, one in the new dataset), involving 31150 distinct primes. With the previous paragraph it follows that hundreds of moduli in the new dataset are vulnerable due to moduli that do not belong to it.

None of the newly found affected moduli are blacklisted (cf. [24]). Based on this brief analysis, we conclude that the problem reported in the main body of this paper persists and that the 99.8% security level of 1024-bit RSA moduli is further reduced. We are not planning further investigations, have put our data under custody, and informed the EFF and some of the 2048-bit RSA modulus owners of our findings.

The new data suggests that, as the body of keys grows, so does the percentage of keys affected and that the simple step of abandoning 1024-bit keys for 2048-bit ones may be less effective than desired. RSA's feature that allows bad number generators to affect other keys in such a public way is problematic.

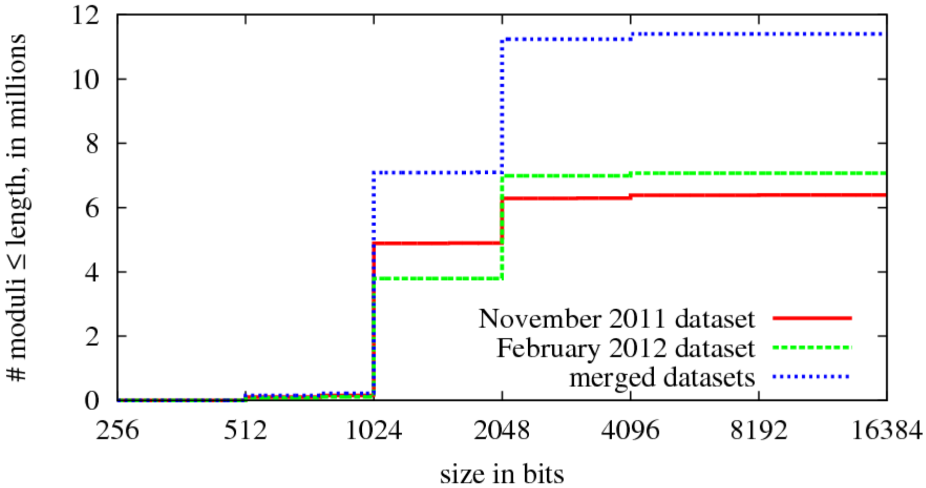

Fig. 7. Cumulative numbers of modulus sizes, with the original collection of distinct RSA moduli in red (cf. Figure 3), the distinct RSA moduli from the February, 2012, EFF SSL Observatory scan in green, and the merged (and uniqued) collection in blue.