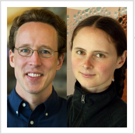
NOVEMBER 8, 2015 POSTS COMMENTS

FREEDOM TO TINKER

research and expert commentary on digital technologies in public life



How is NSA breaking so much crypto?

OCTOBER 14, 2015 BY ALEX HALDERMAN AND NADIA HENINGER 92 COMMENTS

There have been rumors for years that the NSA can decrypt a significant fraction of encrypted Internet traffic. In 2012, James Bamford published **an article** quoting anonymous former NSA officials stating that the agency had achieved a “computing breakthrough” that gave them “the ability to crack current public encryption.” The Snowden documents also hint at some extraordinary capabilities: they show that NSA has built extensive infrastructure to intercept and decrypt VPN traffic and suggest that the agency can decrypt at least some HTTPS and SSH connections on demand.

However, the documents do not explain *how* these breakthroughs work, and speculation about possible backdoors or broken algorithms has been rampant in the technical community. Yesterday at ACM CCS, one of the leading security research venues, we and twelve coauthors presented **a paper that we think solves this technical mystery**.

The key is, somewhat ironically, Diffie-Hellman key exchange, an algorithm that we and many others have advocated as a defense against mass surveillance. Diffie-Hellman is a cornerstone of modern cryptography used for VPNs, HTTPS websites, email, and many other protocols. Our paper shows that, through a confluence of number theory and bad implementation choices, many real-world users of Diffie-Hellman are likely vulnerable to state-level attackers.

For the nerds in the audience, here’s what’s wrong: If a client and server are speaking Diffie-Hellman, they first need to agree on a large prime number with a particular form. There seemed to be no reason why everyone couldn’t just use the same prime, and, in fact, many applications tend to use standardized or hard-coded primes. But there was a very important detail that got lost in translation between the mathematicians and the practitioners: an adversary can perform a single enormous computation to “crack” a particular prime, then easily break any individual connection that uses that prime.

How enormous a computation, you ask? Possibly a technical feat on a scale (relative to the state of computing at the time) not seen since the Enigma cryptanalysis during World War II. Even estimating the difficulty is tricky, due to the complexity of the algorithm involved, but our paper gives some conservative estimates. For the most common strength of Diffie-Hellman (1024 bits), it would cost a few hundred million dollars to build a machine, based on special purpose hardware, that would be able to crack one Diffie-Hellman prime every year.

Would this be worth it for an intelligence agency? Since a handful of primes are so widely reused, the payoff, in terms of connections they could decrypt, would be enormous. Breaking a single, common 1024-bit prime would allow NSA to passively decrypt connections to two-thirds of VPNs and a quarter of all SSH servers globally. Breaking a second 1024-bit prime would allow passive eavesdropping on connections to nearly 20% of the top million HTTPS websites. In other words, a one-time investment in massive computation would make it possible to eavesdrop on trillions of encrypted connections.

NSA could afford such an investment. The 2013 “**black budget**” request, leaked as part of the Snowden cache, states that NSA has prioritized “investing in groundbreaking cryptanalytic capabilities to defeat adversarial cryptography and exploit internet traffic.” It shows that the agency’s budget is on the order of $10 billion a year, with over $1 billion dedicated to computer network exploitation, and several subprograms in the hundreds of millions a year.

Based on the evidence we have, we can’t prove for certain that NSA is doing this. However, our proposed Diffie-Hellman break fits the known technical details about their large-scale decryption capabilities better than any competing explanation. For instance, the Snowden documents show that NSA’s **VPN decryption infrastructure** involves intercepting encrypted connections and passing certain data to supercomputers, which return the key. The design of the system goes to great lengths to collect particular data that would be necessary for an attack on Diffie-Hellman but not for alternative explanations, like a break in AES or other symmetric crypto. While the documents make it clear that NSA uses other attack techniques, like software and hardware “implants,” to break crypto on *specific* targets, these don’t explain the ability to passively eavesdrop on VPN traffic at a large scale.

Since weak use of Diffie-Hellman is widespread in standards and implementations, it will be many years before the problems go away, even given existing security recommendations and our new findings. In the meantime, other large governments potentially can implement similar attacks, if they haven’t already.

Our findings illuminate the tension between NSA’s two missions, gathering intelligence and defending U.S. computer security. If our hypothesis is correct, the agency has been vigorously exploiting weak Diffie-Hellman, while taking only small steps to help fix the problem. On the defensive side, NSA has recommended that implementors should transition to elliptic curve cryptography, which isn’t known to suffer from this loophole, but such recommendations tend to go unheeded absent explicit justifications or demonstrations. This problem is compounded because the security community is hesitant to take NSA recommendations at face value, following apparent efforts to **backdoor cryptographic standards**.

This state of affairs puts everyone’s security at risk. Vulnerability on this scale is indiscriminate—it impacts everybody’s security, including American citizens and companies—but we hope that a clearer technical understanding of the cryptanalytic machinery behind government surveillance will be an important step towards better security for everyone.

Freedom to Tinker is hosted by Princeton's Center for Information Technology Policy, a research center that studies digital technologies in public life. Here you'll find comment and analysis from the digital frontier, written by the Center's faculty, students, and friends.

CITP
CENTER FOR INFORMATION TECHNOLOGY POLICY

Search this website… Search

What We Discuss

AACS bitcoin Broadband CD Copy Protection censorship CITP Competition Computing in the Cloud Copyright Cross-Border Issues cybersecurity policy DMCA DRM Education Events Facebook FCC Government Government transparency Grokster Case Humor Innovation Policy Law Managing the Internet Media Misleading Terms NSA Online Communities Patents Peer-to-Peer Predictions Princeton Privacy Publishing Recommended Reading Secrecy Security Spam Super-DMCA surveillance Tech/Law/Policy Blogs Technology and Freedom Virtual Worlds Voting Wiretapping

Contributors

Select Author...

Archives by Month

- **2015:** J F M A M J J A S O N D
- **2014:** J F M A M J J A S O N D
- **2013:** J F M A M J J A S O N D
- **2012:** J F M A M J J A S O N D
- **2011:** J F M A M J J A S O N D
- **2010:** J F M A M J J A S O N D
- **2009:** J F M A M J J A S O N D
- **2008:** J F M A M J J A S O N D
- **2007:** J F M A M J J A S O N D
- **2006:** J F M A M J J A S O N D
- **2005:** J F M A M J J A S O N D
- **2004:** J F M A M J J A S O N D
- **2003:** J F M A M J J A S O N D
- **2002:** J F M A M J J A S O N D

For more details, see our research paper: **Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice**. (*Update:* We just received the Best Paper Award at CCS 2015!)

J. Alex Halderman *is an associate professor of Computer Science and Engineering at the University of Michigan and director of Michigan's Center for Computer Security and Society.*

Nadia Heninger *is an assistant professor of Computer and Information Science at the University of Pennsylvania.*

author log in

FILED UNDER: UNCATEGORIZED TAGGED WITH: CRYPTO, NSA, SURVEILLANCE

Comments

pete.d says:
October 14, 2015 at 9:25 pm

For the less-informed of us in the audience, what does it mean to "crack" or "break" a prime? As the article a bove describes the situation, it seems that these primes aren't secret. That is, they are in use in publicly available software and often the same prime is used by multiple people. And of course, primes aren't factorable. So what else is there?

And what's the solution? Does each user of the software have to pick their own 1024 bit prime?

Reply

informatimago says:
October 14, 2015 at 10:10 pm

Actually, there are two primes g and p, and the secret shared between the two end-points is is g^(a*b) modulo p, a and b being random numbers choosen each by one end-point, and they are never transmitted. The Diffie-Hellman protocol and mathematics allow the end-points to find this shared secret by exchanging g^a modulop and g^b modulo p. It is assumed that computing g^(a*b) modulo p is very hard when you only know g, p, g^a modulo p and g^b modulo p.

What this paper says, if I've understood correctly, is that it possible to pre-compute a function from g and p (even if it takes a lot of time and money), that will let you find g^(a*b) modulo p easily from g^a modulo p and g^b modulo p.

http://mathworld.wolfram.com/Diffie-HellmanProtocol.html

Reply

Mario says:
October 15, 2015 at 6:52 am

More exactly: p is a prime, g doesn't have to. To "break" the prime p means: to build some kind of database such that given some value y, it is possible to find the 'a' (secret) value such that y=g^a (mod p). From y=g^a and z=g^b generated by (TLS) client and server, it is then possible to get (1) the secret 'a' value, and then (2) to compute K=z^a=g^{a.b}, which is the TLS main secret key.

Reply

Anonymous says:
October 15, 2015 at 8:24 am

just get help from someone else please i dont have a ansewr

Reply

merrinen says:
October 16, 2015 at 5:04 pm

Lets say you have a lock. You give a key to one place so that only they can open your lock. This would be equivalent of an encrypted connection.

NSA's system has generated a lot of keys. They try a lot and lot of keys against the lock, and at some point they find out what keys work with a particular lock.

And the unfortunate part is that there are a lot of similar locks everyone are using. So with the generated key or set of keys they know will work can open the lock and NSA gets to the data and can see it.

Dan says:
October 15, 2015 at 12:46 am

The solution is to increase the size from 1024 bits to 2048. It is believed to be currently infeasible to crack such a long prime, even with a 10 billion dollar budget.

Reply

Wyatt says:
October 15, 2015 at 12:59 pm

May as well just jump to 4096, then. Why do it in half-measure?

Reply

David Jao says:
October 15, 2015 at 1:15 pm

I think the most likely source of the next break for Diffie-Hellman will be quantum computers, and quantum computers apply just as well to 4096-bit keys as 2048-bit keys. The way (sub)-exponential functions work, breaking a 2048-bit key is not just a little harder than breaking a 1024-bit key, it's much much MUCH harder. Most experts believe that 2048-bit security is adequate.

Reply

Mogo R. says:
October 15, 2015 at 3:31 pm

"Most experts believe that 2048-bit security is adequate." until it's not.

David Jao says:
October 15, 2015 at 4:07 pm

OK, but seriously, where do you stop? Why not 8192-bit keys? 16384? 65536? By the way, 65536-bit RSA keys take five hours to generate with OpenSSL on a typical PC.

Niko D. says:
October 15, 2015 at 11:27 pm

Was this not likely the thinking when 1024-bit was chosen? We must not have realized that computation was also growing exponentially.

Adnan says:
October 16, 2015 at 8:05 am

"Most experts believe that 2048-bit security is adequate." until it's not.

Who cares about distant future? 2048 is pretty safe for next 20 years

Bill F. says:
October 17, 2015 at 3:02 am

I also remember when AES128 was considered impossible to break and that was the end all beat all.

While I don't offer a solution to the problem, I never feel totally comfortable saying that any amount of security will be penetrable given a couple of years to technological advances.

Robert Claypool says:
October 18, 2015 at 5:44 pm

You stop when it takes too long to generate the secret on current computers, and I don't know why no one's pointed it out. As computers get faster, not only can they break encryption faster, but they can encrypt faster, too.

MegaZone says:
October 15, 2015 at 5:13 pm

Real-world performance is big issue. On some typical HW I'm familiar with, moving from 1024-bit DHE to 2048-bit DHE is a 5x performance cost, at minimum. So at best you now get 20% of the TLS transactions-per-second that you got at 1024-bit. And that's dedicated crypto HW, optimized for this task. Going to 4096-bit from 2048-bit is even worse.

The other issue is that there is currently no standard for negotiating Finite Field DHE key sizes, and 1024-bit is the default presumed size. If you have a client that only does 1024-bit, and the server negotiates DHE and tries to do 2048-bit – the connection will fail. There is currently a draft working its way through the IETF to add negotiation of DHE key sizes to TLS: https://tools.ietf.org/html/draft-ietf-tls-negotiated-ff-dhe-10

Since 1024-bit is basically universal it is the 'safe' value to use when DHE is negotiated. Until the Negotiated FF-DHE extension is widely supported using any other bit size is a crapshoot. If the server picks a size not supported by the client, then the connection fails.

Note that ECDHE doesn't have this problem because negotiation was built in from the start. So using ECDHE instead of DHE is a preferred real-world solution.

Reply

Anonymous says:
October 16, 2015 at 5:39 am

Damn, does that mean our passwords are getting longer ?

Reply

Anonymous says:
October 16, 2015 at 8:21 am

If that is what you got from that comment, then I'm afraid you need to revisit some basic concepts in cryptography. No, it doesn't mean that passwords will have a changed minimum length. It means that the way computers negotiate their way of ascertaining that they're talking to whomever they're INTENDING to talk to is changed in a specific way.

daedlanth says:
October 15, 2015 at 1:21 am

It would be best if each user picked their secret key and switched it often. When you give crackers enough payload generated from a given key you give them a sharper analysis.The Diffie-Hellman.encryption method uses modulus 13 residuals BTW.

Reply

Anonymous says:
October 15, 2015 at 12:00 pm

ha ha… "crackers"…

Reply

David Jao says:
October 15, 2015 at 12:59 pm

Changing keys frequently doesn't help, since the problem here is that all the keys belong to a single group, and the hypothetical NSA pre-computation affects all keys in that group. What is needed is to change to a different, larger group.

Reply

Phil Karn says:
October 16, 2015 at 4:03 am

Or to change DH groups on a regular basis. Unfortunately it takes a fair amount of crunching to find a large prime, as you have to do it on a trial-and-error basis.

You start with a large random odd number of the size you want and generate some number of sequential integers (well, every other integer) starting at that point. Sieve them by some number of small primes to weed out most of the losers, then test the rest for primality with Miller-Rabin. That test is only probabilistic, so you have to run it some number of times on each candidate to reduce the chances of the number actually being composite down to, say, the chance of an arithmetic error in your computer. That can take a lot of work.

Furthermore (I'm not up on this), a Diffie-Hellman prime modulus should be a "strong" prime, i.e., p-1 must have at least one large prime factor. That whittles the list down even further, so the cost is greater than generating, e.g., an RSA key pair with prime factors of the same size.

Reply

Matt Nordhoff says:
October 16, 2015 at 11:09 am

It's less of an issue than you think. Any x86 PC from the last five years can generate a 1024-bit strong prime in (on average) a few seconds. A server could switch primes once a minute for little cost. A multi-core box could even dedicate one CPU core to spitting out new primes constantly. Of course, it's still be a problem for phones or embedded devices.

David Jao says:
October 16, 2015 at 2:40 pm

It takes such enormous effort to change DH groups at runtime that in almost all cases you're far better off just using a larger (albeit fixed) group. The issue is that if you allow changing primes, then the prime must be included in the public key. This requirement DOUBLES the key size. But if you're willing to pay the cost of doubling the key size, then you might as well just use keys twice as large! There is no universe in which variable 1024-bit keys will win in security over fixed 2048-bit keys. A 2048-bit key is much much MUCH harder to break than even several billion 1024-bit keys.

mike.l says:
October 15, 2015 at 2:43 am

pete.d, I had the same question. (and I didn't know enough math to grasp informatimago's answer!) But I found this article really helpful, especially the "Two Keys are Better than One" section.
http://blogs.discovermagazine.com/crux/2013/07/31/how-to-create-codes-that-even-the-nsa-cant-break/#.Vh9JHPlVhBc
I might be totally wrong on this, but I think that what is being "cracked/broken" is not actually a prime, but rather the *product* of two primes.

Reply

Anonymous says:
October 15, 2015 at 1:35 pm

"I might be totally wrong on this, but I think that what is being "cracked/broken" is not actually a prime, but rather the *product* of two primes."

Yes, essentially. Start with the product of two primes, then derive what those primes are to get to the product. That is "cracking" the prime.

Reply

Harry Johnston says:
October 15, 2015 at 6:12 pm

Nope. You're thinking of RSA.

In this context, "cracking" a prime means building up a huge database of information about certain properties of the prime field, i.e., about the behaviour of arithmetic when performed modulo that prime.

Reply

whoby? says:
October 15, 2015 at 12:56 am

of course, the wikipedia article also tells us this: "RSA claims that 1024-bit keys are likely to become crackable some time between 2006 and 2010″ …
https://en.wikipedia.org/wiki/Key_size#Asymmetric_algorithm_key_lengths . Yeah they're different algorithms (and

different classes of one too), but I think its kinda clear that 1024bit primes are not providing enough entropy.

Reply

Anonymous says:
October 15, 2015 at 5:19 am

No, it's not clear at all. RSA security does not apply to DH, and vice versa.

Reply

Alan Braggins says:
October 15, 2015 at 7:30 am

From the same Wikipedia paragraph: "The Finite Field Diffie-Hellman algorithm has roughly the same key strength as RSA for the same key sizes."

Yes, it's a simplification, but it shows that there has been a general awareness for a while that 1024 bits isn't really enough. This paper doesn't really change that; it does demonstrate that it might well have been not merely possible but worthwhile for the NSA to break a 1024 bit DH key.

Reply

David Jao says:
October 15, 2015 at 1:04 pm

I remember attending a talk by Adi Shamir in 2003 where he mentioned that 1024-bit factoring would be within the reach of well-funded adversaries. One can only imagine how much easier it is now. And yes, the techniques used for factoring are extremely similar to the techniques used for DH/DLOG. Moreover, factoring is unlike DH in that there are no shared parameters, so there would be no way to do some sort of massive precomputation that allows you to factor integers in rapid-fashion. It's been clear for some time that 1024-bit keys, especially DH/DLOG keys, are unsuitable for use in the modern world. Yet, as is typical in crypto, the world has not moved to abandon 1024-bit keys even after decades of advance warning, because changing crypto in deployed software is hard.

Reply

erikbjare says:
October 15, 2015 at 7:51 am

Actually, if you read the article: "The Finite Field Diffie-Hellman algorithm has roughly the same key strength as RSA for the same key sizes. The work factor for breaking Diffie-Hellman is based on the discrete logarithm problem, which is related to the integer factorization problem on which RSA's strength is based. Thus, a 3072-bit Diffie-Hellman key has about the same strength as a 3072-bit RSA key."

Reply

Stephan says:
October 15, 2015 at 2:09 am

Can't we use multiple primes? So, exchange multiple keys using different primes over DiffieHellman and then combine them into the actual key? Thus you need to break all primes that are used to decrypt the key exchange. Thats still doable given enough time though :(

Reply

Anon says:
October 15, 2015 at 7:43 am

This would most likely cause too much overhead in the normal use of DH.

Reply

Max says:
October 15, 2015 at 10:52 am

Another thought is that Alice and Bob first agree on which prime to use out of a relatively large box of primes. That would make it a harder task to have all possible primes cracked while keeping roughly the same calculatory

Additionally, use bigger a, b and primes. Prime based crypto is a race with computer capability.

Reply

Mark Gritter says:
October 15, 2015 at 2:17 pm

Negotiating a prime makes you vulnerable to the same man-in-the-middle attacks you were presumably trying to avoid. It does help stop purely eavesdropping-based attacks, but not "cause both sides to pick the prime the NSA has already broken" attacks.

Reply

David Jao says:
October 15, 2015 at 1:05 pm

You're better off just using a single much larger prime.

Reply

Liam says:
October 16, 2015 at 6:16 am

It's an interesting point though; if you encrypted using two primes, one after another, obviously that would near double the overhead, however, from a cracker's pov, wouldn't it be much harder to tell if the first prime has been cracked?

Even after using the correct step 1 prime, there wouldn't be anyway of knowing it was successful until the second correct prime was used.

Is that an over simplification?

Reply

David Jao says:
October 16, 2015 at 2:42 pm

No, in public key cryptography the keys satisfy a mathematical relationship, and it is very easy to tell whether or not a key has been cracked. You don't even need any ciphertext. You just need the keys.

In any situation where you're willing to double the overhead, the best security strategy BY FAR is to use 2048-bit keys instead of 1024-bit keys.

Reply

Liam says:
October 19, 2015 at 5:26 pm

Thanks for taking the time to explain that. It's all very interesting to me; wish I had more time to read up about cryptography

bazar says:
October 15, 2015 at 3:24 am

Thanks for the post. Since this is a passive attack, I assume the servers do not support +1k DHE keys. Can you publish a list of the top million www-sites that use 1024-bit DHE. It would make it easier to vote with one's feet and it would also put some pressure for the companies to upgrade.

Reply

anonymous (not that it matters anymore anyway) says:
October 15, 2015 at 4:36 am

Given the dollar amount suggested to build such a computational engine, this attack is not limited to nation state agencies. Very large companies such as Apple, Google, Amazon, Facebook, etc have plenty of money and also spare compute resources to likely carry out the same type of cryptanalysis if so inclined.

Reply

stefancaunter says:
October 15, 2015 at 11:35 am

indeed, that compute capacity may be available at a negotiated price as part of a nation state's budget

Reply

Martin Anderson says:
October 15, 2015 at 7:08 am

What does this post add to the coverage in May?

http://it.slashdot.org/story/15/05/20/1258251/logjam-vulnerability-threatens-encrypted-connections

The PDF seems to be unchanged.

Reply

Chuck says:
October 15, 2015 at 11:36 am

It gets greater exposure.

One article is lost in the churn of the internet. Two articles (or a repost of the same article) may get notice. Ten thousand copies of the same article ensures people will see it.

Reply

Richard says:
October 15, 2015 at 7:41 am

So I will admit I am not a cryptography expert and only have an interest in it, but I still have a question I really want to know the answer to.

My question is this: why. Why. WHY ON GOD'S GREEN EARTH would you use *hard freaking coded* prime numbers for a *cryptography* application? What the *christ* were they thinking???

Reply

Anonymous says:
October 15, 2015 at 8:05 am

because the software isn't written by true cryptographers, either, unfortunately

Reply

Anonymous says:
October 15, 2015 at 8:56 am

Because finding these massive prime numbers are difficult and time-consuming.

Reply

Anonymous says:
October 15, 2015 at 12:32 pm

prime finding is really not the hardest thing out there.a bin 1800 digit prime has roughly 600 decimal digits. Getting a list of all primes in the 500-700 digit range would be awful, but finding a subset of numbers that are likely primes is fast, then one can attempt to verify primality from there (obviously, this is the long step)

Reply

Harry Johnston says:
October 15, 2015 at 6:16 pm

Prime finding is easy, but for DH to be secure you can't use any old prime – it has to be carefully chosen.

Reply

Adnan says:
October 16, 2015 at 8:11 am

Even a phone can find 2048 bit random prime within a second with very high probability that it is a prime. Finding 2048 bit prime (secure enough for crypto) is a piece of cake, even for a phone. There are algorithms for that in all standard libraries.

David Jao says:
October 16, 2015 at 2:44 pm

The main issue is not generating the prime. The main issue is that you then have to transmit the prime. The prime is as large as the size of the original key. Thus, transmitting the prime along with the key requires double the key size. But if you're willing to double the key size, you should have just use a double-length key (with a fixed prime) rather than an original-length key with a variable prime. The double-length key is MUCH more secure.

David Jao says:
October 15, 2015 at 1:10 pm

Hard-coded primes are fine as long as the prime is large enough. What happened here is that the prime was large enough by 1990 standards, but is not large enough now.

I agree that if you're going to use hard-coded primes then you need to plan for future advances in cryptanalysis. At the time these standards were created, people had a 25-year time horizon. Guess what? 1990 + 25 = 2015. The good news is that we won't have this problem ever again with respect to RSA or DH keys, because the next 25-year time horizon will put us into the quantum computing era, invalidating all of RSA and DH. The bad news is that we're just as slow as ever when it comes to updating crypto software deployments.

Reply

Mr. Raymond Kenneth Petry says:
October 16, 2015 at 4:42 pm

Quantum Computering (sic) looks suspicious from the beginning: It takes 4.5 Gyr for a U-238 nucleus to crack its own internal code with 50% likelihood, twice that for 75%… QM may say the probability of an outcome is uniformly distributed but hasn't publicly claimed that time-to-outcome is also uniform; isochronous computing proceeds near sequentially: you'd probably need a gram-atom of qubits to approximate parallel processing—for 4.5 Gyr….

Reply

Folkert Wiekmeijer says:
October 15, 2015 at 7:52 am

The most pragmatic solution is adding new pairs (g, p) to your TLS implementation. Precomputing tables for ALL 1024 bit primes is still beyond the NSA's capabilities. This solution does, for most TLS implementations, not require recompiling any code. And it works even when your TLS impl or that of your peer only handles 1024 bit security. Don't use primes from the net though, better dust of your algebra books.

Reply

Anonymous says:
October 15, 2015 at 10:08 am

agreed

Reply

David Jao says:
October 15, 2015 at 1:11 pm

Honestly, if you're going to do this, you might as well just update to 2048-bit primes. Changing an implementation to support multiple 1024-bit primes is usually harder than changing it to support a single 2048-bit prime, especially if your original implementation assumes a hard-coded prime.

Reply

Anonymous says:
October 16, 2015 at 5:59 pm

“Precomputing tables for ALL 1024 bit primes is still beyond the NSA’s capabilities.”

And it will be for a very, very long time.

Letting the number of primes less than N be Np = N/ln(N) [approximately]. If N=2^b then Np = (2^b)/b*ln(2), so for N=2^1024 Np = (2^1014)/ln(2). If we eliminate the primes less than 2^1013 [i.e. 1024 bit numbers where the msb is zero] the number of primes is 2^1023/ln(2).

In decimal notation 2^1023/ln(2) = 1.29676 * 10^308.

After computing these primes you still have to store them. Each 1024 bit prime needs 128 Bytes of storage — this works out to 1.66 * 10^310 Bytes.

I recently bought a 1 TB drive for about $50 — you would need 1.66 * 10^298 such drives at a cost of $83 * 10^298 — less any quantity discounts :). Way more than enough to make our national debt look like chump change.

And if you got those drives where would you store them?

Reply

Anonymous says:
October 18, 2015 at 2:24 am

Utah

Reply

Phill Hallam-Baker says:
October 15, 2015 at 9:55 am

It isn’t just the choice of prime that is the problem here, it is the construction used to derive keys in TLS.

At the time those 1024 and 512 bit primes were chosen, it was widely believed that DH keys only needed to be half the size of an equivalent RSA key. People wanted short keys because they wanted key setup to fit in a packet.

The way TLS works when doing ephemeral keying is that the RSA keys are used to generate a pre-master secret, call it P and then P is used to authenticate the DH parameters which are used to derive the master secret, call it M. In the current design, M is a feature of the DH parameters alone (M=DH). So the systems do a strong key negotiation, use it to authenticate a weak one and then use the weak one.

This can be fixed using the construction M= SHA2 (DH + P) so the ephemeral is used as a mix in to salt the pre-master secret rather than throwing away that work completely.

When the Internet starts moving to the new CFRG curves for ECC, I expect most people will end up using the 448 bit curve because it offers a huge speed improvement over RSA2048 (for servers) and a comfortable security margin over any public key algorithm that is widely used today (NIST defines a 521 bit curve but I am not aware of anyone issuing certs for it). Using a 448 bit curve in certificates makes perfect sense. But for perfect forward secrecy, it is overkill. I would much rather use the 255 bit curve and rekey more frequently.

Incidentally, BULLRUN tells us that the NSA spends $250 million a year sabotaging efforts to develop security standards. Is this one of the things they bought with that money? I did suggest the salted construction back in the day and was told that it was ‘unnecessary’ and I was ‘troublemaking’. Would it be overly paranoid of me to suggest that maybe this was the work of an NSA mole? After all wouldn’t the way someone like that would ingratiate themselves be to be always willing to volunteer to do lots of scut work?

Reply

Furby says:
October 15, 2015 at 11:06 am

Shouldn’t a state, that has become aware of such a big hole, warn its citizens about it so they can plug the hole rather than leave it open for enemy states to snoop in exactly the same way?

Yes, they get to hear all the juicy gossip about Janice in accounting’s affair with Doug in personnel, but what about all the company secrets that are being taken by nations that could do anything from hurting us financially to losing our technological edge.

Reply

Bill Cheswick says:
October 16, 2015 at 8:14 am

The NSA has two missions: to read other people’s mail, and to protect us from the same.

I have been told on several occasions over the years by highly placed NSA people that the first mission trumps the second.

ches

Reply

John Laprise says:
October 15, 2015 at 11:14 am

Let me suggest an alternative explanation of human parallel processing. For decades, the NSA has been the largest employer of theoretical mathematicians and promptly has them sign security clearance agreements. They have essentially built a very large mathematics college which is mostly knowledge permeable in a single direction in that it's mathematicians make use of the discoveries of the world at large but do not quickly share their own.

Given this state of affairs, let's consider mathematics as a field. Basically there's known and unknown math given the constraints of today's information society. However, the knowledge and work of the NSA's mathematicians represent a body of known math, but only to the NSA. Meanwhile NSA mathematicians are enjoying free rider access to the work of mathematicians around the world.

In this scenario, it is likely that the NSA's understanding of mathematics as a whole is greater than that of the world at large and that given their focused interest, they may well be able to employ developed knowledge to tackle problems that are unsolvable or at the very least require far more onerous approximations or solutions. The NSA's encryption and decryption prowess is likely based upon the unique mathematics knowledge advantage it possesses.

Reply

Craig Lewis says:
October 15, 2015 at 12:06 pm

It is assumed that the NSA is farther ahead in crypto math than the public, but we don't have a lot of data points to figure out how far they're ahead.

I'm a casual follower of crypto, so I personally could have missed a lot of data points here. I'm only aware of one data point to estimate how far ahead the NSA is. It took public mathematicians about 10 years to understand the NSA's suggestion that changed SHA into SHA1. So we can guess that the NSA is 10 years ahead of the public mathematicians.

This attack gives us another data point. How long has this program been going on? The linked document says this slide is from 2009. So it's been running at least 6 years. So they're still 5 to 10 years ahead of the public.

Reply

Jan Steinman says:
October 15, 2015 at 1:40 pm

It would seem that identifying information as encrypted makes you a target, no?

What about steganography? Combining encryption with obfuscation might do the trick for those who wish to avoid NSA scrutiny. My bet is that they don't spend a lot of effort looking at cute kitten photos. Another obfuscation might be to send your encrypted message in a "spam" payload. Is the NSA really opening all the DHL shipping notices for $1 million money orders, sent as .EXE files?

It seems to me that what the NSA is left with is "routine" encrypted traffic, and that through obfuscation, it should still be possible for moderately sophisticated terrorist groups, etc. to communicate without scrutiny.

Reply

Marcel Waldvogel says:
October 16, 2015 at 1:25 am

The problem identified here is limited to a few of the currently known DH primes. Keys created without DH or with different, maybe even stronger, primes, are not affected by this problem.

So using e.g. RSA (as in PGP or S/MIME) or changing the prime or switching to Elliptic Curves avoids this. No need to switch to Steganography, which makes it hard to talk privately with more than a handful of peers. I guess the latter should be the goal.

Reply

Phil Karn says:
October 16, 2015 at 5:02 am

Steganography is cute, but has limited applicability because of the enormous overhead it usually entails. And it's harder to do than you think.

I think it's much better to strongly encrypt as much of our traffic as we possibly can, including all the unimportant stuff. The bigger the haystack, the harder it is to find the needles. And we already know the NSA simply can't resist collecting bigger haystacks.

Reply

graham says:
October 16, 2015 at 6:23 am

I'd like to see a 1K effectively random blob attached to all emails, as a tag in all pictures etc. It may be random data (if you have nothing to say) or it may be encrypted data. No explanation as to the type of encryption or any other protocols embedded within it (that should be a pre-shared secret via completely other means). That's plenty of noise for the NSA to deal with.

Reply

Red says:
October 17, 2015 at 8:11 am

good for you, graham. one can imagine software that uses random noise (from a leaky diode, perhaps in the sound card, or even a spark-plug) etc, to automatically create genuine random content email, etc, and that simply sends it in many directions. assuming that this strategy became ubiquitous, given that creating noise is always going to be cheaper than "decrypting" noise…a considerable "delta T", the signal to noise avalanche would, logically, force the creation of discrimination within the sample, eg decrypt time assigned to less than 100% of traffic. eavesdropping would be forced to decide to not attempt decrypt on some subset of traffic. Of course wrinkles also naturally come to mind – encryption of randomness, etc. but lets recall that the essence of concealment is to conceal the fact that a thing is secret, ie what's secret is the fact that something is secret… by way of deception – all that – or, if you like, recall the wizard of oz's dictum: pay no attention to the man behind the green curtain… meantime OTP can sit in plain sight…and key to open hard-copy stuff, letter-to-editor, library books, etc. for genuine spooks it is all trivial. this reality suggests, in turn, that the primary result, and probably intent, of massive eavesdropping and decrypt has little or nothing to do with finding criminal actors… I'll leave it there…

Reply

Miko says:
October 15, 2015 at 1:52 pm

My vpn uses blowfish CBC encryption cipher and SHA1 hash algorithm, how would i tell if this affects me?

Reply

Anonymous says:
October 15, 2015 at 3:30 pm

SHA1 is thought to be likely to have purposeful collisions within the year.

Reply

Marcel Waldvogel says:
October 16, 2015 at 1:29 am

These are the symmetric ciphers used to encrypt and authenticate the data stream. How does your VPN determine the ephemeral keys used for these ciphers? If it uses a pre-shared key (same long and hard password used on both sides), then you might be safe, depending on how it generates the ephemeral keys from the PSK. Otherwise, you are likely using DH for the key agreement, and possibly even with one of the few primes the NSA has already made insecure.

Reply

Matunos says:
October 15, 2015 at 5:44 pm

I thought generating large prime (or pseudoprime) numbers was relatively easy, computationally-speaking. Why are so many implementations choosing from a fixed set of them?

Reply

Harry Johnston says:

October 15, 2015 at 6:19 pm

Because you can't use an arbitrary prime, it has to have certain properties in order for the key exchange to be secure.

Reply

Phil Karn says:

October 16, 2015 at 5:04 am

More specifically, it should be a "strong" prime (see Wikipedia).

It's relatively easy to generate large primes when you only do it occasionally to produce a new RSA key pair. It's a lot harder when you have to do it on every new SSL connection.

Reply

anon says:

October 15, 2015 at 5:52 pm

Um it's pretty obvious guys, they go in the back door and grab the keys and copy them. If you ever wondered why you can't view the global hook chain it's because they don't want you to see that they can get in and out. You need to encyrpt something – sure – you need the key either typed or in a file, good luck stopping them seeing and copying that.

Reply

anon says:

October 15, 2015 at 5:55 pm

or they make the company sign a silencing agreement and force them to supply them the keys to copy. Here's a deal you can't disagree with – it's the law, here's some money for your time, and never speak about it or we will execute you for treason.

Reply

Paul Bennett says:

October 15, 2015 at 6:14 pm

Is this a joke? You mention backdoors like they were rumors. Who needs to crack anything when all commercial VPN and firewall software is Federally mandated to have built-in backdoors? The "black hat" industry is simply an outgrowth of the revolving door between the backdoor writers/enforcers and the "Internet Security" industry.

Reply

Mark says:

October 16, 2015 at 5:21 am

You seem to have forgotten that the rest of the world exists. There are plenty of VPN providers outside of the USA, and there are also plenty of encryption products written by non-Americans.

Reply

Anonymous says:

October 16, 2015 at 3:56 pm

What about India

Reply

Scott says:

October 15, 2015 at 7:47 pm

The use of hard-coded primes has been known for over a decade. It was a bad idea then and now it that has propagated for all this time.

Reply

Andrew Sutherland says:
October 16, 2015 at 5:37 am

As suggested in the last sentence in the introduction to the paper, the best solution is not to switch to a larger modulus for finite field Diffie Hellman, it is to instead use elliptic curve Diffie Hellman. The group of points on an elliptic curve over a 256-bit finite field provides roughly the same level of security as a 3072-bit RSA key or 3072-bit finite field modulus (assuming the group has prime or near-prime order, which is what one will do in practice). More importantly, the only known algorithms for breaking Diffie Hellman on an elliptic curve actually do have exponential running times versus the subexponential time algorithms that are known for factoring integers and for computing discrete logarithms in the multiplicative group of a finite field. In practical terms: doubling the key size for an elliptic curve from 256 to 512 bits has roughly the same effect as increasing an RSA or Diffie-Hellman finite field modulus from 3072 bits to 15360 bits.

In fact, Google switched to using ECEDH (elliptic curve ephemeral Diffie Hellman) as their default for SSL session key exchange several years ago for precisely this reason (Facebook, Amazon, and several others followed suit shortly thereafter).

Reply

David Jao says:
October 16, 2015 at 2:51 pm

Yes to all these points, and in addition elliptic curve DH has lower computational costs and smaller key sizes than RSA/DH, with the cost and size advantage of elliptic curves increasing as key sizes increase. The performance advantage was one of the main motivations for Google to switch to ECEDH. It is important to mention that Google controls a large portion of both the servers (Google services) and the browsers (Google Chrome) on the internet, and for this reason Google is one of the very few companies that is in a position to drive change in deployed crypto unilaterally.

Reply

lilbahr says:
October 17, 2015 at 7:47 am

This is why you should use One-Time-Pad if you really want absolute secrecy. OTP, the only encryption technique, that can not be broken, because there is no algorithm there to break.

Sure, with OTP there is the problem of having the need to meet at least once but given that you can change 1 TB key that time and then keep telling the other side in secret where to find more, it really doesn't matter. 1 TB key you can talk in the phone with the other person for the rest of your life and no one else can ever know what it is about; no matter how powerful computers they have.

People championing modern encryption over OTP are either working for the man or their pets, fools.

Reply

David Jao says:
November 4, 2015 at 2:16 am

You think Amazon is really going to meet with every single customer to exchange a 1TB OTP key? You think every single internet user is going to meet with every single web site owner to exchange a 1TB OTP key? You do realize that none of these trillions of 1TB OTP keys can be shared or re-used?

You do realize how much simpler it is to have one private key which is kept secret from everyone, compared with the nightmarish scenario of thousands of keys for thousands of websites where each key must be shared with exactly one web site but kept secret from all others?

Good luck.

Reply

lilbahr says:
November 6, 2015 at 10:32 pm

Who said anything about using OTP everywhere? Obviously there is some level where the application turns from plain consumer level to operations of critical secrecy.

Naturally absolute secrecy (OTP) is more expensive a service than the vanilla secrecy. Amazon most likely could do it, for instance, although I am not really talking about that kind of need for secrecy, really. Where you need absolute secrecy you can not really use a subscriber like amazon, or pretty much anyone else, because they can simply leak your infrastructure because of legal coercion anyway.

People always complain about the Complexity of dealing with OTP. All kinds of excuses. Well, that is exactly why we have software: To deal with complexity and turn complex things into every day utilities.

There is a level of communication that NSA, or any other equivalent org wants to follow. Be it Mary sending her titties to her boyfriend across the world, your neighbourly cartel leader talking to his compadre, or business talking about their next big move that will disrupt the stock market, those bits of communication would immensely benefit from using OTP and most often they really would be able to switch OTP keys that one time. Later they could even tell in their secret communication where to go dig further keys, in total secrecy.

Absolute secrecy vs practical secrecy. There is no reason to just use one of them and the other can not replace the other. As said: People championing modern encryption _over_ OTP are either working for the man or their pets, fools.

Reply

James Kratzer says:
October 17, 2015 at 1:51 pm

The problem here is not so much one of encryption level, or type, or of mathematics, but of TRUST. Public trust in government, TRUSTWORTHINESS of NSA goals and agendae, and exactly WHAT the Director of Central Intelligence (the person that the Director of the NSA is Supposed to report to, below POTUS) and the Secretary of the Department of Homeland Security actually DO with all that information and Intelligence they obtain from cracking all the surveillance and Internet communications they monitor.
I, personally, haven't transmitted or received anything (other than my banking and bill-paying) that I object to N0 Such Agency having access to over the internet. But then again, I'm 63 years old and happily married. YOUR mileage may vary. The point is that NSA is Supposed to PROTECT and DEFEND the USA, and it's CITIZENS, from electronic and cybernetic attacks, as part of it's mission; the rest of it being providing secure communications FOR the U.S. Government in the form of codes and ciphers, and executing electronic espionage services TO the U.S. Government Intelligence community. But There's where the rub comes in – electronic espionage services against WHOM? Foreign, or domestic subjects? Domestic is supposed to be only with a Federal Court Order, not willy-nilly, blanket eavesdropping. Which is what they can, and do, perform, with the kind of decryption services we've been discussing here.

Reply

Anonymous says:
October 17, 2015 at 11:47 pm

The key to protecting your data in an age when encryption is useless is to physically split your datagrams into pieces and anonymize them.

Reply

anon says:
October 18, 2015 at 2:45 am

Intent to Deprecate: SHA-1 certificates

https://groups.google.com/a/chromium.org/forum/#!topic/security-dev/2-R4XziFc7A%5B1-25%5D

Reply

Red says:
October 18, 2015 at 9:43 am

Just a few thoughts

– most US people "educated" in recent years cannot write or read in cursive style… This has implications for personal record keeping. Get a book on copperplate…and an ink pen.

– morse in sidelobes as a steganographic strategy.

-noise as psudocrypto

Reply

David Gilkinson says:
October 21, 2015 at 12:12 pm

Limit privileges of connections utilising keys lower than 2048-bit. As security demands increase, 2048-bit can also be relegated to legacy connections that are able to run on outdated or antiquated hardware

Reply

David Gilkinson says:
October 21, 2015 at 12:23 pm

Limit privileges of connections utilising keys lower than 2048-bit. As security demands increase, 2048-bit can also be relegated to legacy connections that are able to run on outdated or antiquated hardware

The only problem is the NSA can ask for your lock, even if stored elsewhere else, steal your lock without notifying anyone, and also abuse access to your lock by using it to gain access to your property or that of any unknown persons living with you.

Reply

list of vpn providers says:
October 27, 2015 at 7:46 am

Second, you may notice that you aren't being targeted with certain types of
advertising. You just cannot resist preferring SSH VPNover
the others since it opens the door for a plethora of benefits that prove to
be fruitful for you in the long run. SSTP
is a Microsoft VPN with very good encryption but is only available for
Windows 7 an Vista, and is therefore harder to find.

Reply

Speak Your Mind

Name

Email

Post Comment

Return to top of page