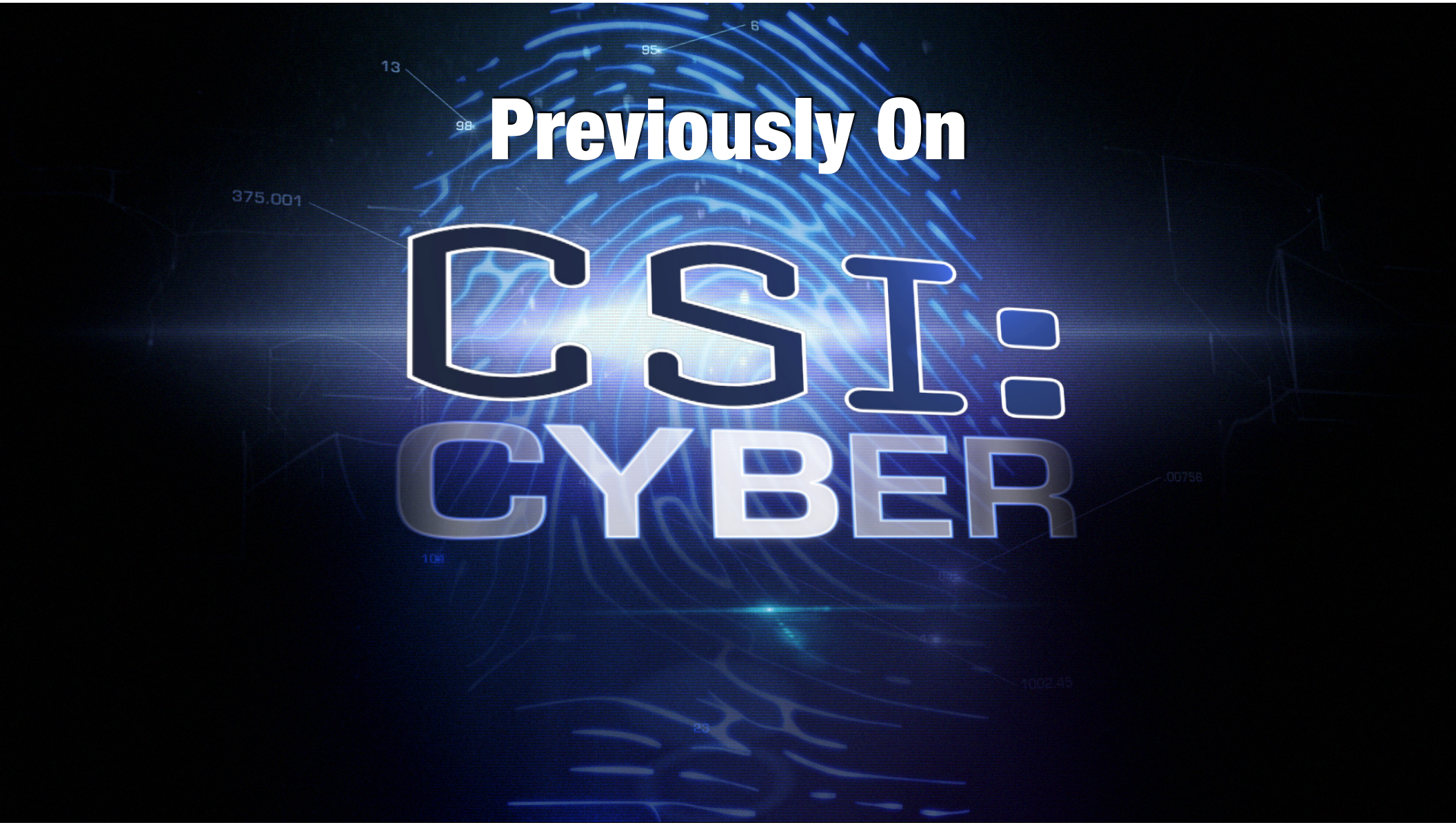
Cyber CSI II: Apple *vs.* FBI Encryption, Privacy, and Policy

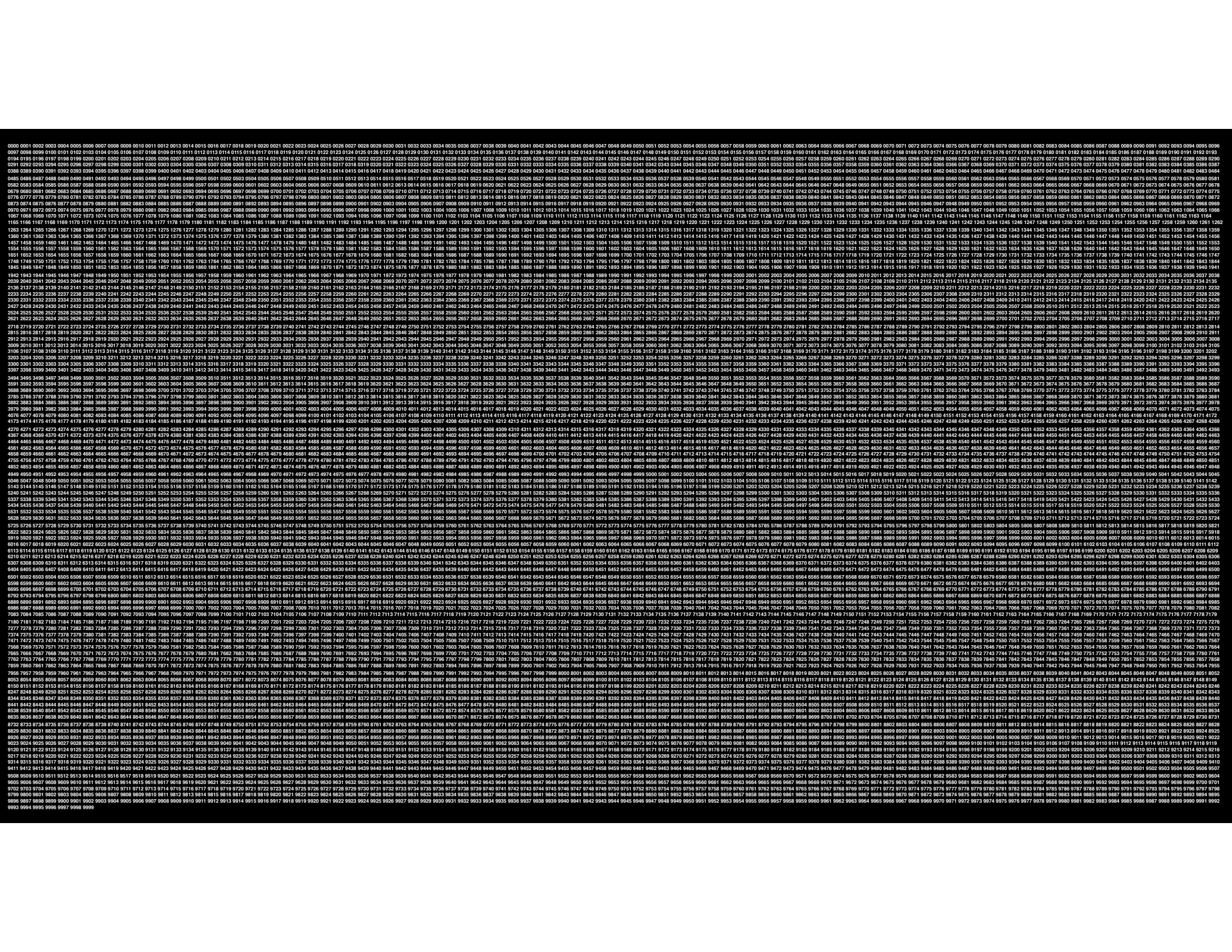
Zachary N J Peterson
Assistant Professor of Computer Science

REK Data Studio, Spring 2016


Previously On
CSI:
CYBER


DEPARTMENT OF JUSTICE
FEDERAL BUREAU OF INVESTIGATION
FIDELITY
BRAVERY
INTEGRITY


Enter Passcod
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
9 WXYZ
8

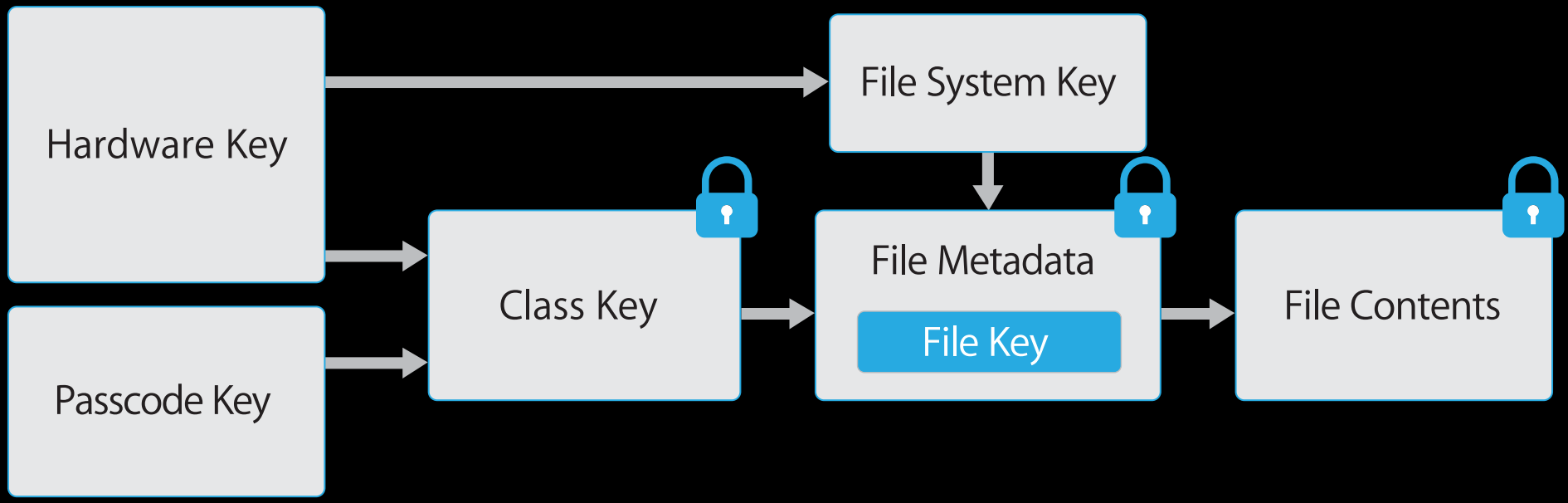



Enter Passcod
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
9 WXYZ
8

1. Apple will **bypass** or **disable** the **auto-erase feature**
2. Apple will **enable** the FBI to **submit passcodes electronically**
3. Apple will **disable** any **purposeful delay** between attempts

$$sign(\text{iOS}, k_{pri}) = sig$$

$$verify(\text{iOS}, sig, k_{pub}) = \text{true or false}$$


Hardware Key
Passcode Key
File System Key
Class Key
File Metadata
File Key
File Contents

Attempts	Delay Enforced
1-4	None
5	1 minute
6	5 minutes
7-8	15 minutes
9	1 hour


iPhone
Designed by Apple in California Assembled in China Model A1456
FCC ID:BCG-E2644A IC:579C-E2644A IMEI: 013300008603789
9:41 AM
100%
Tuesday
10
Messages
Calendar
Photos
Camera
Weather
Clock
Maps
Videos
Notes
Reminders
Stocks
Game Center
Newsstand
iTunes Store
App Store
Passbook
Compass
Settings
Phone
Mail
Safari
Music


DEPARTMENT OF JUSTICE
FEDERAL BUREAU OF INVESTIGATION
FIDELITY
BRAVERY
INTEGRITY

Subverting Cryptography

Ban it altogether	**ludicrous**
Escrowed keys (known backdoors)	**we tried this**
Undermining standards (unknown backdoors)	**we tried this, too**
Mandating standards	**you guessed it, tried it**
Legal pressures	**let’s see if this works**

$$n = pq$$

$$e = 2^{16} + 1$$

$$d = e^{-1} \mod \phi(n)$$

$$c = m^e \mod n$$

$$m = c^d \mod n$$

$$n = pq$$

$$e = 2^{16} + 1$$

$$d = e^{-1} \mod \phi(n)$$

$$c = m^e \mod n$$

$$m = c^d \mod n$$

(math is not a munition)

Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice

David Adrian[¶] Karthikeyan Bhargavan[*] Zakir Durumeric[¶] Pierrick Gaudry[†] Matthew Green[§]
J. Alex Halderman[¶] Nadia Heninger[‡] Drew Springall[¶] Emmanuel Thomé[†] Luke Valenta[‡]
Benjamin VanderSloot[¶] Eric Wustrow[¶] Santiago Zanella-Béguelin[‖] Paul Zimmermann[†]
[*]INRIA Paris-Rocquencourt [†]INRIA Nancy-Grand Est, CNRS, and Université de Lorraine
[‖]Microsoft Research [‡]University of Pennsylvania [§]Johns Hopkins [¶]University of Michigan

For additional materials and contact information, visit WeakDH.org.

ABSTRACT

We investigate the security of Diffie-Hellman key exchange as used in popular Internet protocols and find it to be less secure than widely believed. First, we present Logjam, a novel flaw in TLS that lets a man-in-the-middle downgrade connections to "export-grade" Diffie-Hellman. To carry out this attack, we implement the number field sieve discrete log algorithm. After a week-long precomputation for a specified 512-bit group, we can compute arbitrary discrete logs in that group in about a minute. We find that 82% of vulnerable servers use a single 512-bit group, allowing us to compromise connections to 7% of Alexa Top Million HTTPS sites. In response, major browsers are being changed to reject short groups.

We go on to consider Diffie-Hellman with 768- and 1024-bit groups. We estimate that even in the 1024-bit case, the computations are plausible given nation-state resources. A small number of fixed or standardized groups are used by millions of servers; performing precomputation for a single 1024-bit group would allow passive eavesdropping on 18% of popular HTTPS sites, and a second group would allow decryption of traffic to 66% of IPsec VPNs and 26% of SSH servers. A close reading of published NSA leaks shows that the agency's attacks on VPNs are consistent with having achi[illegible] a break. We conclude that moving to [illegible] methods should be a pri[illegible]

coded, or widely shared Diffie-Hellman parameters has the effect of dramatically reducing the cost of large-scale attacks, bringing some within range of feasibility today.

The current best technique for attacking Diffie-Hellman relies on compromising one of the private exponents (a, b) by computing the discrete log of the corresponding public value $(g^a \bmod p,\ g^b \bmod p)$. With state-of-the-art number field sieve algorithms, computing a single discrete log is more difficult than factoring an RSA modulus of the same size. However, an adversary who performs a large precomputation for a prime p can then quickly calculate arbitrary discrete logs in that group, amortizing the cost over all targets that share this parameter. Although this fact is well known among mathematical cryptographers, it seems to have been lost among practitioners deploying cryptosystems. We exploit it to obtain the following results:

Active attacks on export ciphers in TLS. We introduce Logjam, a new attack on TLS by which a man-in-th[illegible] attacker can downgrade a connection [illegible] tography. This attack is [illegible] but applie[illegible]

Protocol Failure in the Escrowed Encryption Standard

Matt Blaze
AT&T Bell Laboratories
mab@research.att.com

August 20, 1994

Abstract

The Escrowed Encryption Standard (EES) defines a US Government family of cryptographic processors, popularly known as "Clipper" chips, intended to protect unclassified government and private-sector communications and data. A basic feature of key setup between pairs of EES processors involves the exchange of a "Law Enforcement Access Field" (LEAF) that contains an encrypted copy of the current session key. The LEAF is intended to facilitate government access to the cleartext of data encrypted under the system. Several aspects of the design of the EES, which employs a classified cipher algorithm and tamper-resistant hardware, attempt to make it infeasible to deploy the system without transmitting the LEAF. We evaluated the publicly released aspects of the EES protocols as well as a prototype version of a PCMCIA-based EES device. This paper outlines various techniques that enable cryptographic communication among EES processors without transmission of the valid LEAF. We identify two classes of techniques. The simplest allow communication only between pairs of "re-
applicati
ties. The second, more com-

The proposal, called the Escrowed Encryption Standard (EES) [NIST94], includes several unusual features that have been the subject of considerable debate and controversy. The EES cipher algorithm, called "Skipjack", is itself classified, and implementations of the cipher are available to the private sector only within tamper-resistant modules supplied by government-approved vendors. Software implementations of the cipher will not be possible. Although Skipjack, which was designed by the US National Security Agency (NSA), was reviewed by a small panel of civilian experts who were granted access to the algorithm, the cipher cannot be subjected to the degree of civilian scrutiny ordinarily given to new encryption systems.

By far the most controversial aspect of the EES system, however, is *key escrow*. As part of the cryptosynchronization process, EES devices generate and exchange a "Law Enforcement Access Field" (LEAF). This field contains a copy of the current session key and is intended to enable a government
to recover the cleartext
sion k

Forward Secrecy

On the Practical Exploitability of Dual EC in TLS Implementations

Stephen Checkoway,[1] Matthew Fredrikson,[2] Ruben Niederhagen,[3] Adam Everspaugh,[2]
Matthew Green,[1] Tanja Lange,[3] Thomas Ristenpart,[2]
Daniel J. Bernstein,[3,4] Jake Maskiewicz,[5] and Hovav Shacham[5]
[1] *Johns Hopkins University,* [2] *University of Wisconsin,* [3] *Technische Universiteit Eindhoven,*
[4] *University of Illinois at Chicago,* [5] *UC San Diego*

Abstract

This paper analyzes the actual cost of attacking TLS implementations that use NIST's Dual EC pseudorandom number generator, assuming that the attacker generated the constants used in Dual EC. It has been known for several years that an attacker generating these constants and seeing a long enough stretch of Dual EC output bits can predict all future outputs; but TLS does not naturally provide a long enough stretch of output bits, and the cost of an attack turns out to depend heavily on choices made in implementing the RNG and on choices made in implementing other parts of TLS.

Specifically, this paper investigates OpenSSL-FIPS, Windows' SChannel, and the C/C++ and Java versions of the RSA BSAFE library. This paper shows that Dual EC exploitability is fragile, and in particular is stopped by an outright bug in the certified Dual EC implementation in OpenSSL. On the other

The documents also make specific reference to a set of pseudorandom number generator (PRNG) algorithms adopted as part of the National Institute of Standards and Technology (NIST) Special Publication 800-90 [21] in 2006, and also standardized as part of ISO 18031 [15]. These standards include an algorithm called the Dual Elliptic Curve Deterministic Random Bit Generator (Dual EC). As a result of these revelations, NIST reopened the public comment period for SP 800-90.

Known weaknesses in Dual EC. Long before 2013, Dual EC had been identified by the security community as biased [8, 27], extremely slow, and backdoorable.

SP 800-90 had already noted that "elliptic curve arithmetic" makes Dual EC generate "pseudorandom bits ... slowly than the other DRB

A Few Thoughts on Cryptographic Engineering

Some random thoughts about crypto. Notes from a course I teach. Pictures of my dachshunds.

Tuesday, December 22, 2015

On the Juniper backdoor



You might have heard that a few days ago, Juniper Systems announced the discovery of "unauthorized code" in the ScreenOS software that underlies the NetScreen line of devices. As a result of this discovery, the company announced a pair of separate vulnerabilities, CVE-2015-7755 and CVE-2015-7756 and urged their customers to patch immediately.

The first of these CVEs (#7755) was an authentication vulnerability, caused by a malicious hardcoded password in SSH and Telnet. Rapid7 has an excellent writeup of the issue. This is a pretty fantastic vulnerability, if you measure by the impact on security of NetScreen users. But on the *technological awesomeness* scale it rates about a two out of ten, maybe a step above 'hit the guy with a wrench'.

The *second* vulnerability is a whole different animal. The advisory notes that CVE-7756 -- which is independent of the first issue -- "may allow a knowledgeable attacker who can monitor VPN traffic to decrypt that traffic." This is the kind of vulnerability that makes cryptographers cry tears of joy. It certainly did that for me:

Follow

Contributors

- Christina Garman
- Matthew Green
- Ian Miers

Popular Posts



On the NSA
Let me tell you the story of my tiny brush with the biggest crypto story of the year . A few weeks ago I received a call from a reporter a...



What's the matter with PGP?
Image source: @bcrypt . Last Thursday, Yahoo announced their plans to support end-to-end encryption using a fork of Google's end-to-...



Attack of the week: FREAK (or 'factoring the NSA for fun and profit')
Cryptography used to be considered 'munitions'. This is the story of how a handful of cryptographers 'hacked' the NSA....



Truecrypt report
A few weeks back I wrote an update on the Truecrypt audit promising that we'd have some concrete results


GoldKey
GOLDKEY
SECURITY
shift

Technology | Wed Apr 6, 2016 5:13pm EDT

White House declines to support encryption legislation: sources

WASHINGTON | BY MARK HOSENBALL AND DUSTIN VOLZ

Related: ELECTION 2016, POLITICS, TECH



A worker checks an iPhone in a repair store in New York, in this file photo taken February 17, 2016.
REUTERS/EDUARDO MUNOZ/FILES

The White House is declining to offer public sup
give federal judges clearer author
enforcement crack

PHOTOS OF THE DAY



Our top photos from the last 24 hours. Slideshow »

Separatist clashes in Azerbaijan
How to build a bomb
Sleeping volcano awakens

BUILD YOUR PERFECT CAN

What’s Right?

- **Technologies** just **don’t exist**
- **US law** only **works in the US**
- **Software** is **speech**
- **Slippery slope**
- Can we **enhance** other aspects of **investigation**?
- What are the **implications** on **human rights**?
- The US’s role as a **world leader**

Your thoughts?

Thank you.