Security on msnbc.com

Hidden flaw jeopardizes millions of online transactions

Mathematicians discover weakness in commonly used encryption schemes

Jump to discuss comments below

Recommend 38 Tweet 9 +1 3 Share 20

Below: Discuss Related

By Paul Wagenseil

updated 2/15/2012 6:01:37 PM ET

Mathematicians based in Switzerland and the United States have discovered a small but the most commonly used digital encryption schemes, a flaw that could undermine the secu communication.

COMPUTERWORLD

Topics | News | In Depth | Reviews | Blogs | Opin

Blogs

IT Blogwatch

A Daily Digest of IT Blogs from Richi Jennings

More posts | Read bio

February 15, 2012 - 6:00 A.M.

RSA crypto: 'flawed', 'risky', 'quagmire of vulnerabilities'

2 Comments Like 5 +1 1

TAGS: certificate, crypto, cryptography, Diffie-Hellman, encryption, key management, RSA, SSL

IT TOPICS: Applications, Cybercrime & Hacking, Financial IT, Government & Regulation, Internet, Privacy, Security, Security Hardware & Software

A group of six academic researchers have concluded that real-world RSA encryption keys are riskier than Diffie-Hellman-based ones. It seems that some of the random numbers used to generate them weren't, errm, random. In IT Blogwatch, bloggers wonder if the sky is falling.

The Register®

Hardware | Software | Music & Media | Networks | Security | Cloud | Public Sector | Business | Science

Crime | Malware | Enterprise Security | Spam | ID

Print | Comment | Tweet | Like 1 | Alert

'Predictably random' public keys can be cracked - crypto boffins

Battling researchers argue over whether you should panic

By **John Leyden** · **Get more from this author**

Posted in Enterprise Security, 16th February 2012 13:38 GMT

Analysis Cryptography researchers have discovered flaws in the key generation that underpins the security of important cryptography protocols, including SSL.

Secure Chat

Alice wants to send a secret message to Bob?

- Sometime in the past, they exchange a one-time pad
- Alice uses the pad to encrypt the message
- Bob uses the same pad to decrypt the message

Key point Without the pad, Eve cannot understand the message

15

Encryption Machine

Goal Design a machine to encrypt and decrypt data

S	E	N	D	M	O	N	E	Y

encrypt

g	X	7	6	W	3	v	7	K

decrypt

S	E	N	D	M	O	N	E	Y

16

Encryption Machine

Goal Design a machine to encrypt and decrypt data

S	E	N	D	M	O	N	E	Y

encrypt

g	X	7	6	W	3	v	7	K

decrypt

S	E	N	D	M	O	N	E	Y

Enigma encryption machine

- "Unbreakable" German code during WWII
- Broken by Turing bombe
- One of first uses of computers
- Helped win Battle of Atlantic by locating U-boats

17

A Digital World

Data is a sequence of bits [bit = 0 or 1]

- Text
- Programs, executables
- Documents, pictures, sounds, movies, …

File formats txt, pdf, java, exe, docx, pptx, jpeg, mp3, divx, …

computer with a lens

computer with earbuds

computer with a radio

A Digital World

Data is a sequence of bits [bit = 0 or 1]

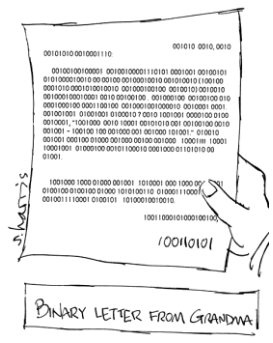
- Text
- Programs, executables
- Documents, pictures, sounds, movies, …

File formats txt, pdf, java, exe, docx, pptx, jpeg, mp3, divx, …

computer with a cash dispenser

computer with a ballot box

computer with a heating element



Copyright 2004, Sidney Harris, http://www.sciencecartoonsplus.com

A Digital World

Data is a sequence of bits [bit = 0 or 1]

- Text
- Programs, executables
- Documents, pictures, sounds, movies, …

Base64 encoding Use 6 bits to represent each alphanumeric symbol.

very weak type of encryption

Binary	Char	Binary	Char	Binary	Char	Binary	Char	Binary	Char	Binary	Char
000000	A	001011	L	010110	W	100001	h	101100	s	110111	3
000001	B	001100	M	010111	X	100010	i	101101	t	111000	4
000010	C	001101	N	011000	Y	100011	j	101110	u	111001	5
000011	D	001110	O	011001	Z	100100	k	101111	v	111010	6
000100	E	001111	P	011010	a	100101	l	110000	w	111011	7
000101	F	010000	Q	011011	b	100110	m	110001	x	111100	8
000110	G	010001	R	011100	c	100111	n	110010	y	111101	9
000111	H	010010	S	011101	d	101000	o	110011	z	111110	+
001000	I	010011	T	011110	e	101001	p	110100	0	111111	/
001001	J	010100	U	011111	f	101010	q	110101	1		
001010	K	010101	V	100000	g	101011	r	110110	2		

One-Time Pad Encryption

Encryption

- Convert text message to N bits

Base64 Encoding

char	dec	binary
A	0	000000
B	1	000001
…	…	…
M	12	001100
…	…	…

S	E	N	D	M	O	N	E	Y	
010010	000100	001101	000011	001100	001110	001101	000100	011000	base64

message

One-Time Pad Encryption

Encryption

- Convert text message to N bits
- Generate N random bits (one-time pad)

S	E	N	D	M	O	N	E	Y	
010010	000100	001101	000011	001100	001110	001101	000100	011000	base64
110010	010011	110110	111001	011010	111001	100010	111111	010010	random bits

message

One-Time Pad Encryption

Encryption

- Convert text message to N bits
- Generate N random bits (one-time pad)
- Take bitwise XOR of two bitstrings

sum corresponding pair of bits: 1 if sum is odd, 0 if even

XOR Truth Table

x	y	x ^ y
0	0	0
0	1	1
1	0	1
1	1	0

S	E	N	D	M	O	N	E	Y	
010010	000100	001101	000011	001100	001110	001101	000100	011000	base64
110010	010011	110110	111001	011010	111001	100010	111111	010010	random bits
100000	010111	111011	111010	010110	110111	101111	111011	001010	XOR

0 ^ 1 = 1

24

One-Time Pad Encryption

Encryption

- Convert text message to N bits
- Generate N random bits (one-time pad)
- Take bitwise XOR of two bitstrings
- Convert binary back into text

Base64 Encoding

char	dec	binary
A	0	000000
B	1	000001
...	...	...
W	22	010110
...	...	...

S	E	N	D	M	O	N	E	Y	
010010	000100	001101	000011	001100	001110	001101	000100	011000	base64
110010	010011	110110	111001	011010	111001	100010	111111	010010	random bits
100000	010111	111011	111010	010110	110111	101111	111011	001010	XOR
g	X	7	6	W	3	v	7	K	encrypted

25

Secure Chat (review)

Alice wants to send a secret message to Bob?

- Sometime in the past, they exchange a one-time pad
- Alice uses the pad to encrypt the message
- Bob uses the same pad to decrypt the message

Key point Without the pad, Eve cannot understand the message

26

One-Time Pad Decryption

Decryption

- Convert encrypted message to binary

g	X	7	6	W	3	v	7	K	
100000	010111	111011	111010	010110	110111	101111	111011	001010	encrypted

27

One-Time Pad Decryption

Decryption

- Convert encrypted message to binary

Base64 Encoding

char	dec	binary
A	0	000000
B	1	000001
...	...	...
W	22	010110
...	...	...

g	X	7	6	W	3	v	7	K	
100000	010111	111011	111010	010110	110111	101111	111011	001010	base64

28

One-Time Pad Decryption

Decryption

- Convert encrypted message to binary
- Use same N random bits (one-time pad)

g	X	7	6	W	3	v	7	K	
100000	010111	111011	111010	010110	110111	101111	111011	001010	base64
110010	010011	110110	111001	011010	111001	100010	111111	010010	random bits

29

One-Time Pad Decryption

Decryption
- Convert encrypted message to binary
- Use same N random bits (one-time pad)
- Take bitwise XOR of two bitstrings

XOR Truth Table

x	y	x ^ y
0	0	0
0	1	1
1	0	1
1	1	0

g	X	7	6	W	3	v	7	K	
100000	010111	111011	111010	010110	110111	101111	111011	001010	encrypted
100000	010111	111011	111010	010110	110111	101111	111011	001010	base64
110010	010011	110110	111001	011010	111001	100010	111111	010010	random bits
010010	000100	001101	000011	001100	001110	001101	000100	011000	XOR

1 ^ 1 = 0

30

One-Time Pad Decryption

Decryption
- Convert encrypted message to binary
- Use same N random bits (one-time pad)
- Take bitwise XOR of two bitstrings
- Convert back into text

Base64 Encoding

char	dec	binary
A	0	000000
B	1	000001
...	...	...
M	12	001100
...	...	...

g	X	7	6	W	3	v	7	K	
100000	010111	111011	111010	010110	110111	101111	111011	001010	encrypted
100000	010111	111011	111010	010110	110111	101111	111011	001010	base64
110010	010011	110110	111001	011010	111001	100010	111111	010010	random bits
010010	000100	001101	000011	001100	001110	001101	000100	011000	XOR
S	E	N	D	M	O	N	E	Y	message

31

Why Does It Work?

Crucial property Decrypted message = original message

Notation	Meaning
a	original message bit
b	one-time pad bit
^	XOR operator
a ^ b	encrypted message bit
(a ^ b) ^ b	decrypted message bit

Why is crucial property true?
- Use properties of XOR.
- (a ^ b) ^ b = a ^ (b ^ b) = a ^ 0 = a

associativity of ^ | always 0 | identity

XOR Truth Table

x	y	x ^ y
0	0	0
0	1	1
1	0	1
1	1	0

32

One-Time Pad Decryption (with the wrong pad)

Decryption
- Convert encrypted message to binary

g	X	7	6	W	3	v	7	K	
100000	010111	111011	111010	010110	110111	101111	111011	001010	encrypted

33

One-Time Pad Decryption (with the wrong pad)

Decryption
- Convert encrypted message to binary

g	X	7	6	W	3	v	7	K	
100000	010111	111011	111010	010110	110111	101111	111011	001010	encrypted
100000	010111	111011	111010	010110	110111	101111	111011	001010	base64

34

One-Time Pad Decryption (with the wrong pad)

Decryption
- Convert encrypted message to binary
- Use wrong N bits (bogus one-time pad)

g	X	7	6	W	3	v	7	K	
100000	010111	111011	111010	010110	110111	101111	111011	001010	encrypted
100000	010111	111011	111010	010110	110111	101111	111011	001010	base64
101000	011100	110101	101111	010010	111001	100101	101010	001010	wrong bits

35

One-Time Pad Decryption (with the wrong pad)

Decryption
- Convert encrypted message to binary
- Use wrong N bits (bogus one-time pad)
- Take bitwise XOR of two bitstrings

g	X	7	6	W	3	v	7	K	encrypted
100000	010111	111011	111010	010110	110111	101111	111011	001010	base64
101000	011100	110101	101111	010010	111001	100101	101010	001010	wrong bits
001000	001011	001110	010101	000100	001110	001010	010001	000000	XOR

36

One-Time Pad Decryption (with the wrong pad)

Decryption
- Convert encrypted message to binary
- Use wrong N bits (bogus one-time pad)
- Take bitwise XOR of two bitstrings
- Convert back into text: Oops

g	X	7	6	W	3	v	7	K	encrypted
100000	010111	111011	111010	010110	110111	101111	111011	001010	base64
101000	011100	110101	101111	010010	111001	100101	101010	001010	wrong bits
001000	001011	001110	010101	000100	001110	001010	010001	000000	XOR
I	L	O	V	E	O	K	R	A	wrong message

37

38

Goods and Bads of One-Time Pads

Good
- Easily computed by hand
- Very simple encryption/decryption processes
- Provably unbreakable if bits are truly random [Shannon, 1940s]

eavesdropper Eve sees only random bits

"one time" means one time only

Bad
- Easily breakable if pad is re-used
- Pad must be as long as the message
- Truly random bits are very hard to come by
- Pad must be distributed securely

impractical for Web commerce

a Russian one-time pad

39

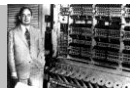
Pseudo-Random Bit Generator

Practical middle-ground
- Let's make a "random"-bit generator gadget
- Alice and Bob each get identical small gadgets

instead of identical large one-time pads

How to make small gadget that produces "random" bits
- Enigma machine
- Linear feedback shift register
- Linear congruential generator
- Blum-Blum-Shub generator
- ...

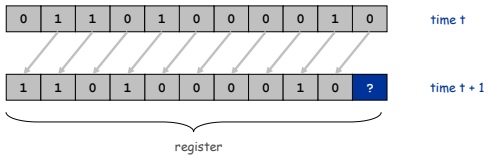
" Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin. "
– Jon von Neumann (left)
– ENIAC (right)



40

Shift Register

Shift register terminology.
- Bit: 0 or 1
- Cell: storage element that holds one bit
- Register: sequence of cells
- Seed: initial sequence of bits
- Shift register: when clock ticks, bits propagate one position to left



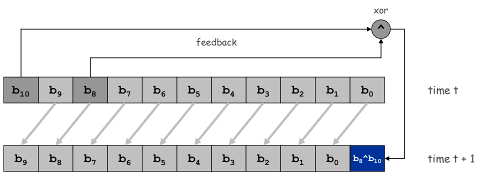
41

Linear Feedback Shift Register (LFSR)

{8, 10} linear feedback shift register

- Shift register with 11 cells
- Bit b_0 is is XOR of previous bits b_8 and b_{10}
- Pseudo-random bit = b_0

LFSR demo



Random Numbers

Q. Are these 2000 numbers random? If not, what is the pattern?

```
110010010011110110111001011010111001100010111111010010000100110100101111001100100111111
1011100000101011000100001110101001101000011110010011001110111111101010000010000100001010
0101010001100000101111000100100110101101111000110100110111001111010111100100010011110101
0111010000010100100010001101010101110000000101100000100111000101110110100101011001100000
11111110011000001111110001100001101111001110100111101001110010011101110111101010101010000
000000010000000001010000000100010000101010100100000000110100000111001000110111010111010100
010100001010001001000101011010100001100001001111001011100111001011110111001001010111011
000010101110010000101110100100101001101100011110111011001010101110000000100110000101111
1001001000111011010110101100011000111011110110101001011000011001110011111110111100001010
0110010001111110101100001000111001010110111000011010110011100011111011011000101101111010
011010100111100001110011001101111111111010000000010010000010110100010011001010111111100001
0000110010100111110001110001101101101110110110101011011000001101110001110101101101000111
0110010111011110010101001110000011101100011010111011100010101011010000000110010000111110
1001100010011111010111000100010110101010011000000111110000110001100111101111111001010000
1110001001101101011110110001001011101011001010001111000101100110100111111001110000011110
1100110010111111111001000000111010000110100100111001101110111101010100010000000101010000
1000001001010001011000101001110100011101001011010011001100111111111110000000001100000000
11110000001100110001111111101100000001011100001001011001011001111001111100111100011110011
0110011111011111100010100011010001011100101001011100011001011011111001101000111110010110
00111001110110111101011010010001100110101111111000100000110101000111000010110110010011 0
1111011110100101001001100011011111011101000101010010100000110001000111101010110010000001
1110100011001001011111011001000101111010100100100001101101001110110011101011111101000100
010010101010110000000001110000001101100001110111001101010111110000010001100010101111010
```

A. No. This is output of {8, 10} LFSR with seed 01101000010!

LFSR Encryption

LFSR encryption

- Convert text message to N bits
- Initialize LFSR with small seed
- Generate N bits with LFSR
- Take bitwise XOR of two bitstrings
- Convert binary back into text

Base64 Encoding

char	dec	binary
A	0	000000
B	1	000001
…	…	…
w	22	010110
…	…	…

S	E	N	D	M	O	N	E	Y	message
010010	000100	001101	000011	001100	001110	001101	000100	011000	base64
110010	010011	110110	111001	011010	111001	100010	111111	010010	LFSR bits
100000	010111	111011	111010	010110	110111	101111	111011	001010	XOR
g	X	7	6	w	3	v	7	K	encrypted

LFSR Decryption

LFSR Decryption

- Convert encrypted message to N bits
- Initialize identical LFSR with same seed
- Generate N bits with LFSR
- Take bitwise XOR of two bitstrings
- Convert binary back into text

Base64 Encoding

char	dec	binary
A	0	000000
B	1	000001
…	…	…
M	12	001100
…	…	…

g	X	7	6	w	3	v	7	K	encrypted
100000	010111	111011	111010	010110	110111	101111	111011	001010	base64
110010	010011	110110	111001	011010	111001	100010	111111	010010	LFSR bits
010010	000100	001101	000011	001100	001110	001101	000100	011000	XOR
S	E	N	D	M	O	N	E	Y	message

Goods and Bads of LFSR Encryption

Goods

- Easily computed with simple machine
- Very simple encryption/decryption process
- Scalable: 20 cells for 1 million bits; 30 cells for 1 billion bits
  [but need theory of finite groups to know where to put taps]

a commercially available LFSR

Bads

- Still need secure, independent way to distribute LFSR seed
- The bits are not truly random
  [bits in our 11-bit LFSR cycle after $2^{11} - 1 = 2047$ steps]
- Experts have cracked LFSR
  [more complicated machines needed]

Other LFSR Applications

What else can we do with a LFSR?

- DVD encryption with CSS
- DVD decryption with DeCSS!
- Subroutine in military cryptosystems

DVD Jon
(Norwegian hacker)

```
/*     efdtt.c     Author:  Charles M. Hannum <root@ihack.net>          */
/*     Usage is:  cat title-key scrambled.vob | efdtt >clear.vob        */

#define m(i)(x[i]^s[i+84])<<

                 unsigned char x[5]      ,y,s[2048];main(
                 n){for( read(0,x,5     );read(0,s ,n=2048
                         ); write(1    ,s,n)         )if(s
                [y=s       [13]%8+20]  /16%4  ==1       ){int
                i=m(       1)17  ^256 +m(0)   8,k       =m(2)
                0,j=       m(4)   17^ m(3)   9^k*      2-k%8
                ^8,a     =0,c    =26;for   (s[y]    -=16;
               --c;j  *=2)a=     a*2^i&    1,i=i /2^j&1
               <<24;for(j=        127;     ++j<n;c=c>
                                  y)
                                  c

                         +=y=i^i/8^i>>4^i>>12,
                i=i>>8^y<<17,a^=a>>14,y=a^a*8^a<<6,a=a
              >>8^y<<9,k=s[j],k         ="7Wo~'G_\216"[k
               &7]+2^"cr3sfw6v;*k+>/n."[k>>4]*2^k*257/
                     8,s[j]=k^(k&k*2&34)*6^c+~y
                                ;}}
```

http://www.cs.cmu.edu/~dst/DeCSS/Gallery

A Closing Profound Question

Q. What is a random number?

LFSR does not produce random numbers
- It is a very simple deterministic machine
- But not obvious how to distinguish the bits it produces from random

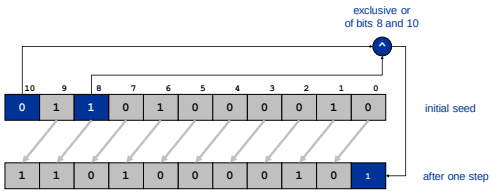
Q. Are truly random processes found in nature?
- Motion of cosmic rays or subatomic particles?
- Mutations in DNA?

Q. Or, is the natural world a (not-so-simple) deterministic machine?

> *" God does not play dice. "*
> *– Albert Einstein*

Extra Slides

Linear Feedback Shift Register



One step of an 11-bit LFSR with initial seed 01101000010 and tap at position 8