Fundamentals of Cryptography

NALINI ELKINS

INDUSTRY NETWORK TECHNOLOGY COUNCIL

PRESIDENT@INDUSTRYNETCOUNCIL.ORG

A few words about me

- President: Industry Network Technology Council
- Founder & CEO: Inside Products, Inc.
- Advisory Board: India Internet Engineering Society
- RFCs: RFC8250 (Embedded performance and diagnostics for IPv6) and others
- Product developer (OEMed by IBM and others)
- Working with IPv6 for over 20 years
- Working with network management, diagnostic, cryptography, performance issues at large brick-and-mortar enterprises for over 30 years

Fundamentals of Cryptography

Details:

- DES
- 3DES
- Asymmetric encryption / symmetric encryption
- Elliptic curve cryptography
- Certificate authority
- Diffie-Hellman key exchange
- Diffie-Hellman groups
- Hashed message authentication code (HMAC)
- HMAC MD5
- HMAC_SHA
- Message authentication code (MAC)
- Message digest algorithm 5 (MD5)
- Rivest Shamir Adleman (RSA)
- Secure hash algorithm 1 (SHA1)
- X.500 distinguished name
- X.509 digital certificate

Concepts

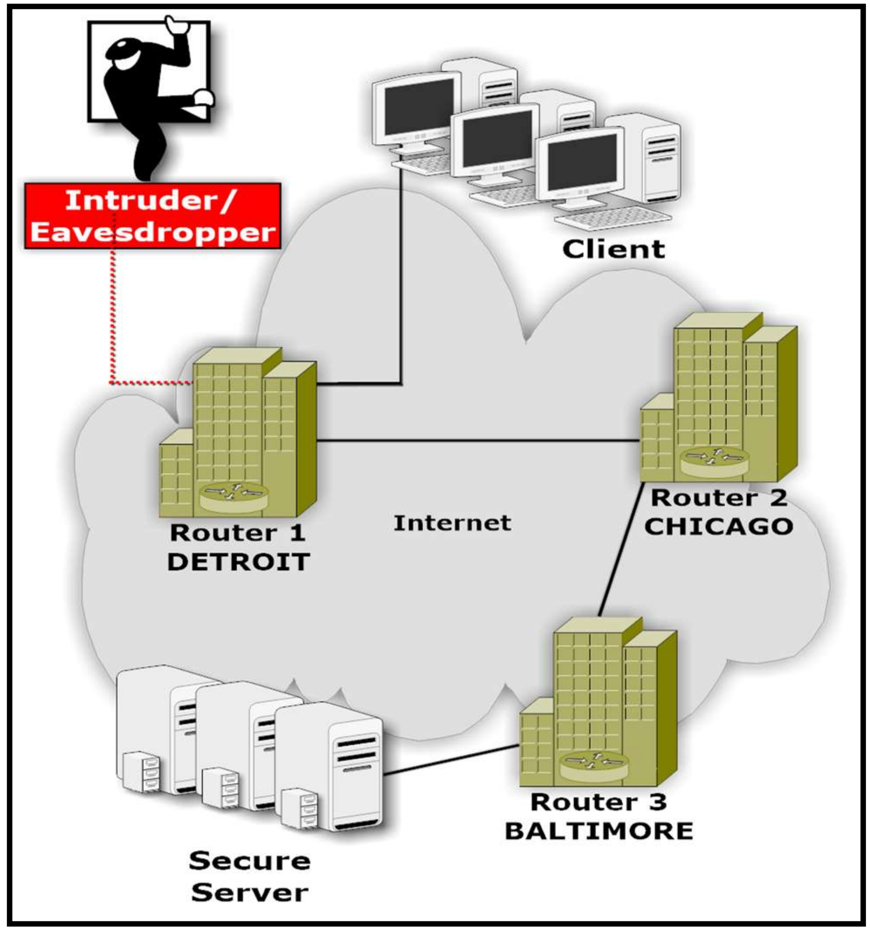
- Block cipher
- Encryption
- Hash
- Keys
- Public / private keys
- Tags

Issues

- Key sizes
- Choice of protocol
- End-to-end security

Why Cryptography

- Today, we have vast communications networks (internet, cell phones, Automatic Teller Machines (ATM)) offering instant ‘secure’ communication.
- The future of Electronic Commerce and, in fact, the electronic world, rests on secure digital communication.
- Unfortunately, so does the success of terrorists, drug rings, people smugglers, child porn, organized crime, spy rings, and ‘cyber crime’.
- Security is why we need to understand cryptography!



Where is cryptography done?

```
✓ Transport Layer Security
  ˅ TLSv1.3 Record Layer: Handshake Protocol: Client Hello
      Content Type: Handshake (22)
      Version: TLS 1.0 (0x0301)
      Length: 512
    ˅ Handshake Protocol: Client Hello
        Handshake Type: Client Hello (1)
        Length: 508
        Version: TLS 1.2 (0x0303)
        Random: aefacb70b6f9d80095dc6a85548065de97668bee7d6917135306b2bd7595bbb2
        Session ID Length: 32
        Session ID: bcbe336580edbde24106dd25f0660683a4be8b25c99f25594aa4ff412bef9ba1
        Cipher Suites Length: 32
      ˅ Cipher Suites (16 suites)
          Cipher Suite: Reserved (GREASE) (0x2a2a)
          Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
          Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
          Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
          Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
          Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
          Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
          Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
          Cipher Suite: TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca9)
          Cipher Suite: TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8)
          Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
          Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
          Cipher Suite: TLS_RSA_WITH_AES_128_GCM_SHA256 (0x009c)
          Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
          Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
          Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
        Compression Methods Length: 1
      > Compression Methods (1 method)
        Extensions Length: 403
```

Cipher Suites are embedded in Protocols

This is a TLS Client Hello packet (Starts TLS session)

Historical Encryption Techniques

- One of the earliest ways to encrypt was using a substitution alphabet.

- This consists of substitution over a single letter (simple substitution). It can be demonstrated by writing out the alphabet in some order to represent the substitution.

Simple Substitution alphabet:

A = H

B = 2

C = y …

Substitution Alphabets

- For example, you may have the following substitution alphabet
- Let's look at the encoded message:

  TQXXA

- What might this be? If we decode it, it becomes:

  HELLO

A = M	N = Z
B = N	O = A
C = O	P = B
D = P	Q = C
E = Q	R = D
F = R	S = E
G = S	T = F
H = T	U = G
I = U	V = H
J = V	W = I
K = W	X = J
L = X	Y = K
M = Y	Z = L

Frequency Analysis

- Substitution alphabets are easy to break – certain letters in a language usually occur more frequently than others.

- This is called: Frequency Analysis

Mary, Queen of Scots: Codes

https://www.nationalarchives.gov.uk/education/resources/elizabeth-monarchy/ciphers-used-by-mary-queen-of-scots/

[Lost and found: Codebreakers decipher 50+ letters of Mary, Queen of Scots | Ars Technica].

English Letter Frequency (based on a sample of 40,000 words)

Letter	Count		Letter	Frequency
E	21912		E	12.02
T	16587		T	9.10
A	14810		A	8.12
O	14003		O	7.68
I	13318		I	7.31
N	12666		N	6.95
S	11450		S	6.28
R	10977		R	6.02
H	10795		H	5.92
D	7874		D	4.32

https://pi.math.cornell.edu/~mec/2003-2004/cryptography/subs/frequencies.html

Mixed Alphabets: Adding a Key

- Mixed alphabets are created by first writing out a keyword, then all the remaining letters.
- For example, you may have the following mixed alphabet

```
Plaintext  alphabet: abcdefghijklmnopqrstuvwxyz
Ciphertext alphabet: zebrascdfghijklmnopqsuvwxy
```

- Let's look at the encoded message:

SIAA ZQ LKBA. VA ZOA RFPBLUAOAR!

- What might this be?

Mixed Alphabets: Answer

```
Plaintext  alphabet:abcdefghijklmnopqrstuvwxyz
Ciphertext alphabet:ZEBRASCDFGHIJKLMNOPQTUVWXY
```

The message :

Encrypted:SIAA ZQ LKBA. VA ZOA RFPBLUAOAR!
Decrypted:FLEE AT ONCE. WE ARE DISCOVERED

Symmetric Key Based Algorithms

- Today, we use key-based algorithms.
- Such algorithms use an encryption key to encrypt the message. So, you need both the message and a 'key'.
- The receiver can then use a decryption key to decrypt the message. So, it needs the encrypted message and also the SAME 'key' used to encrypt.
- These are called "Symmetric Algorithms"

Encryption Key: **TIGER**

Encrypted Message:

Scr qul wdjj gmkr mus smkmppmw

***Decryption Key*: TIGER**

Decrypted Message:

The sun will come out tomorrow

Encryption With Same Key

Here is a simple example. Let's suppose we're transmitting only numerical characters. For example:

```
1 2 3 4 5 6 5 4 3 2 1
```

Let’s choose a key to encrypt the message : "4232". To encrypt, we'll repeat the key as many times as necessary to 'cover' the whole message:

```
1 2 3 4 5 6 5 4 3 2 1
4 2 3 2 4 2 3 2 4 2 3
```

Now, we arrive at the encrypted message by adding both numbers:

```
1 2 3 4 5 6 5 4 3 2 1 +
4 2 3 2 4 2 3 2 4 2 3
---------------------
5 4 6 6 9 8 8 6 7 4 4
```

The resulting (54669886744) is the encrypted message. We decrypt by repeating the key as many time as necessary to cover the message, and then subtract the key character by character:

```
5 4 6 6 9 8 8 6 7 4 4 -
4 2 3 2 4 2 3 2 4 2 3
---------------------
1 2 3 4 5 6 5 4 3 2 1
```

Key is Crucial

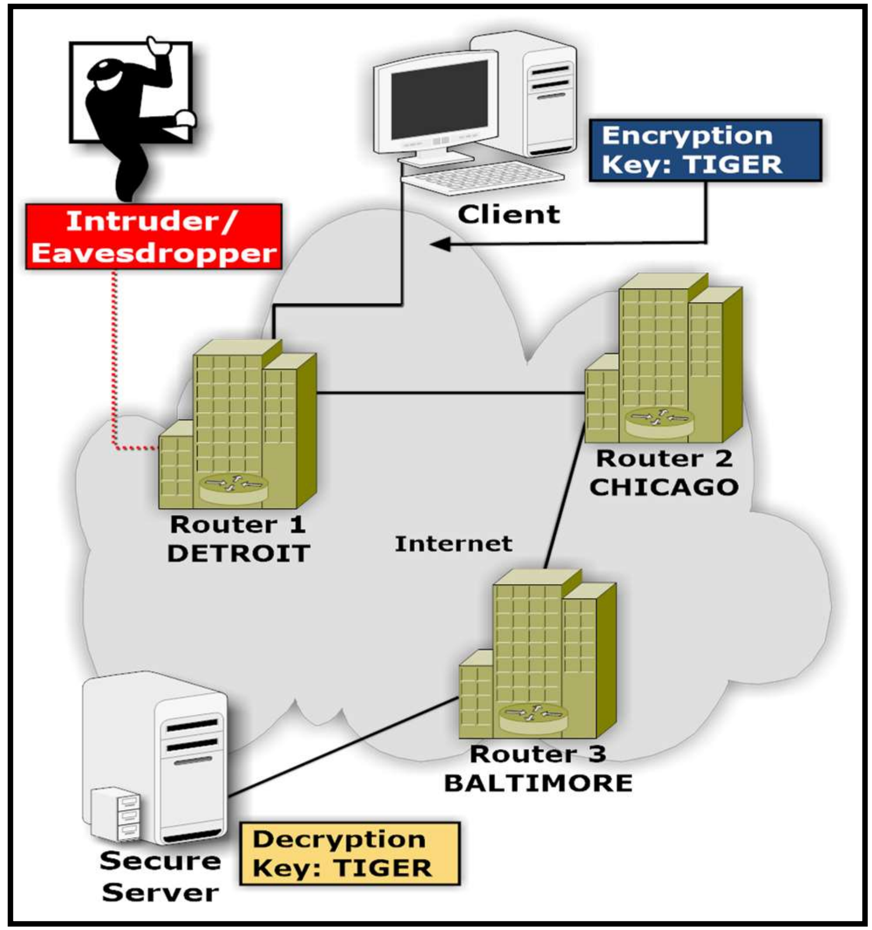
- Notice how it is absolutely necessary to have the decryption key (in this case, the same as the encryption key) to be able to decrypt the message.
- This means that a malicious user would need both the message and the key to eavesdrop on our conversation.
- This is a very trivial example.
- Current key-based algorithms are much more sophisticated (for starters, keys are much longer, and the encryption process is not as simple as 'adding the message and the key').
- However, these complex algorithms are based on the same basic principle shown in our example: a key is needed to encrypt/decrypt message.

Key Lengths:

56 bits
80 bits
128 bits
168 bits
256 bits
768 bits
1,024 bits
2,048 bits
3,072 bits
15,360 bits

Symmetric Algorithms : Problem!

- Symmetric algorithms are generally very fast and simple to implement, they also have several drawbacks.
- Both the sender and the receiver need to agree on the key they will use throughout the secure conversation! This is not a trivial problem.
- In the past, banks used to send out couriers to physically hand the keys to each user!



https://www.iiesoc.in/

https://industrynetcouncil.org/

History of Symmetric Algorithms

- In 1972 the US NATIONAL BUREAU OF STANDARDS (NIST) began the search for an encryption algorithm that could be tested and certified.
- In 1974, IBM offered the US government an algorithm which was based on the early 1970's LUCIFER algorithm.
- The algorithm was tested and 'adjusted' by the NSA and eventually released as a federal standard in 1976.
- This algorithm was: DES. DES is a SYMMETRIC BLOCK cipher based on a 64 bit block. The user feeds in a 64 bit block of plaintext and is returned 64 bits of ciphertext.

Key : Tiger

Algorithm: Add

64 Bits PlainText

Key : Tiger

Algorithm : Add

64 Bits CipherText

```
81 ADCD     PACKET    00000004 12:49:53.215285 Packet Trace
 To Interface        : ETH1              Device: LCS Ethernet     Full=108

  Sequence #         : 0                 Flags: Pkt Out
 IpHeader: Version   : 4                 Header Length: 20
  Tos                : 00                QOS: Routine Normal Service
  Packet Length      : 108               ID Number: 0D8B
  Fragment           :                   Offset: 0
  TTL                : 64                Protocol: UDP    CheckSum: E7D3 FFFF
  Source             : 192.168.1.232
  Destination        : 192.168.1.234
 UDP
  Source Port        : 500   (isakmp)    Destination Port: 500   (isakmp)
  Datagram Length    : 88                CheckSum: 5254 FFFF
 IsaKmp message      : 80
  IsaKmp Header

[ some lines omitted ]

Transform Payload
  Next Payload       : 0(None)           Payload length: 0x20(32) Offset: 0030
  Transform Number   : 0x1(1)            TransformID: 1(KEY_IKE)
  Attribute Type     : 1(Encr Alg)       Value: 1(DES)
  Attribute Type     : 2(Hash Alg)       Value: 1(MD5)
  Attribute Type     : 3(Auth Method)    Value: 1(PresharedKey)
  Attribute Type     : 4(Group,Desc)     Value: 1(768 bit MODP)
  Attribute Type     : 11(Life Type)     Value: 1(seconds)
  Attribute Type     : 12(Life Duration) Value: 0x7080(28800)
    1 payload(s) found
```

Notice that in the IPSec negotiation, the encryption algorithm and key lengths are negotiated.

Block / Stream Ciphers

- A block cipher uses a key and an algorithm to a block of data at once as a group rather than to one bit at a time.
- The main alternative method, used much less frequently, is called the stream cipher.
- In a stream cipher, the key and algorithm are applied to each binary digit in a data stream, one bit at a time.

Sample block cipher

64 Bit Block

8 * 8 bytes

Sample stream cipher

Applied to each bit

0001 1111

IPSec Encoded Packet

IP Header

ESP Header : Block Cipher

TCP, UDP, ICMP, IP

ESP Trailer w/ Padding

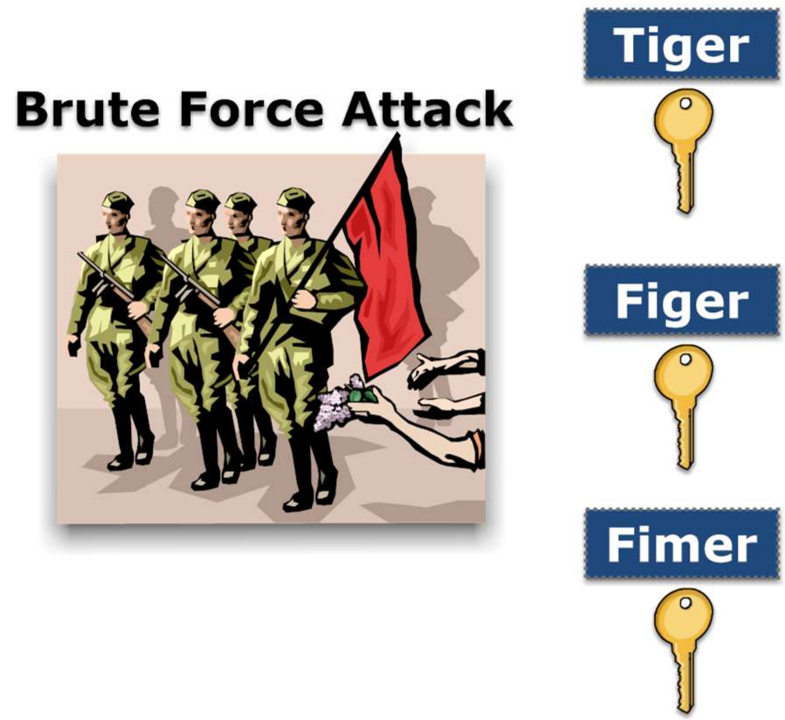
DES uses a 56bit Key

- If there are only 1 million keys, then a cryptanalyst with a powerful computer could use brute force attack and find the key in minutes.
- DES uses a 56-bit key. This is 72,057,594,037,927,936 keys
- When DES was approved as a federal standard in 1976, it was thought that a machine fast enough to test that many keys in a reasonable time would cost an unreasonable amount of money to build, or that a machine cheap enough to be reasonable could not test that many keys in a reasonable time.

1956 : IBM 650 (.0010 MIPs)
1962 : IBM 709 (.1780 MIPs)
1967 : IBM 360/50 (.6800 MIPs)
1972 : IBM 360/165 (1.89 MIPs)
1977 : Amdahl 470/V6 (2.2 or 3.95 MIPs)
1981 : IBM 3033 (4.7 MIPs)
1982 : IBM 4341(1.10 MIPS)
1983 : IBM 3081(10 MIPS)
1986 : 3090 (28 MIPS)
1990 : IBM 3090/600E (32 MIPs)

Brute Force Attacks

- For any cipher, the most basic method of attack is brute force — trying every possible key in turn.
- The length of the key determines the number of possible keys, and hence the feasibility of this approach.
- For DES, questions were raised about the adequacy of its key size early on, even before it was adopted as a standard.
- It was the small key size, rather than theoretical cryptanalysis, which dictated a need for a replacement algorithm.

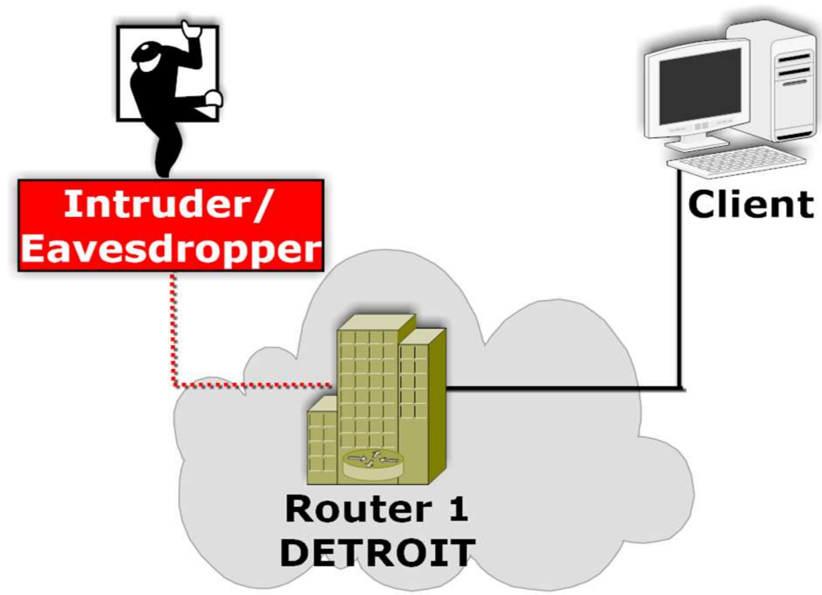


https://www.iiesoc.in/

https://industrynetcouncil.org/

DES → Cryptanalysis

- Data Encryption Standard (DES) was controversial because of “classified design elements”, a relatively short key length and suspicions about a National Security Agency (NSA) backdoor.
- DES came under intense academic scrutiny and motivated the modern understanding of block ciphers and their cryptanalysis. (How to break them!)
- Components of Cryptography:
  - Encryption,
  - Decryption,
  - Cryptanalysis



DES Key Length

56 bits

https://www.iiesoc.in/

https://industrynetcouncil.org/

DES Cracking

- In academia, various proposals for a DES-cracking machine were advanced.
- In 1977, Diffie and Hellman proposed a machine costing an estimated US$20 million which could find a DES key in a single day.
- By 1993, Wiener had proposed a key-search machine costing US$1 million which would find a key within 7 hours.
- The vulnerability of DES was practically demonstrated in the late 1990s. In 1997, RSA Security sponsored a series of contests, offering a $10,000 prize to the first team that broke a message encrypted with DES for the contest.
- That contest was won by the DESCHALL Project, led by Rocke Verser, Matt Curtin, and Justin Dolske, using idle cycles of thousands of computers across the Internet.

$20 million – 1 day

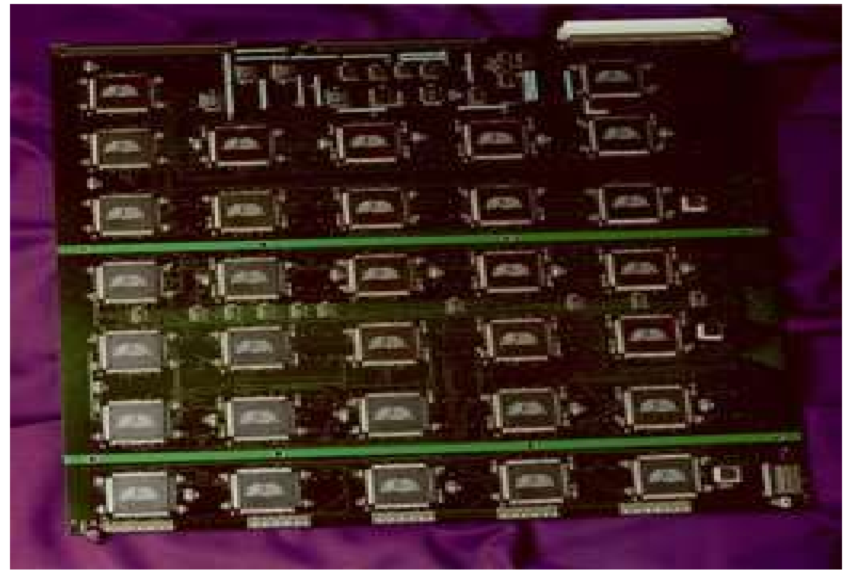
$1 million – 1 day

DESCHALL – Spare capacity (grid!)

At the time the winning key was reported to RSADSI, the DESCHALL effort had searched almost 25% of the total. At its peak, the DESCHALL effort was testing 7 billion keys per second.

The Final Blow

- RSA Security set up DES Challenge II-1, which was solved by Distributed.net in 41 days in January and February of 1998.
- In 1998, the Electronic Frontier Foundation built Deep Crack. It cost $250,000 to build. In response to DES Challenge II-2, on July 17, 1998, Deep Crack decrypted a DES-encrypted message after only 56 hours of work, winning $10,000.
- (At about the same time at least one attorney from the US Justice Department was announcing that DES was unbreakable.)
- Deep Crack was the final blow to DES. The brute force attack showed that cracking DES was actually a very practical proposition.
- For well-endowed governments or corporations, building a machine like Deep Crack would be no problem.



- **Deep Crack consisted of 1856 custom chips, housed on 29 circuit boards of 64 chips.**
- **The boards are then fitted in six cabinets.**
- **The search is coordinated by a single PC which assigns ranges of keys to the chips.**
- **The entire machine was capable of testing over 90 billion keys per second. It would take about 5 days to test every possible key.**

https://www.iiesoc.in/

Today, AES is approved as the Symmetric Key algorithm

But … changes are afoot!

Will discuss in later sections.

Asymmetric Algorithms

- What we have discussed so far are called symmetric algorithms.
- Secure systems also use asymmetric algorithms, where a different key is used to encrypt and decrypt the message.
- Public-key algorithms are the most commonly used type of asymmetric algorithms.
- In public-key cryptography, the two keys are called the private key and the public key
  - Private key: This key must be known only by its owner.
  - Public key: This key is known to everyone (it is public)
- Relation between both keys: What one key encrypts, the other one decrypts, and vice versa. That means that if the sender encrypts something with a public key, the receiver needs his private key to decrypt the message.

***Encryption Key:* TIGER**

Encrypted Message:

Scr qul wdjj gmkr mus smkmppmw

***Decryption Key*: TIGER**

Decrypted Message:

The sun will come out tomorrow

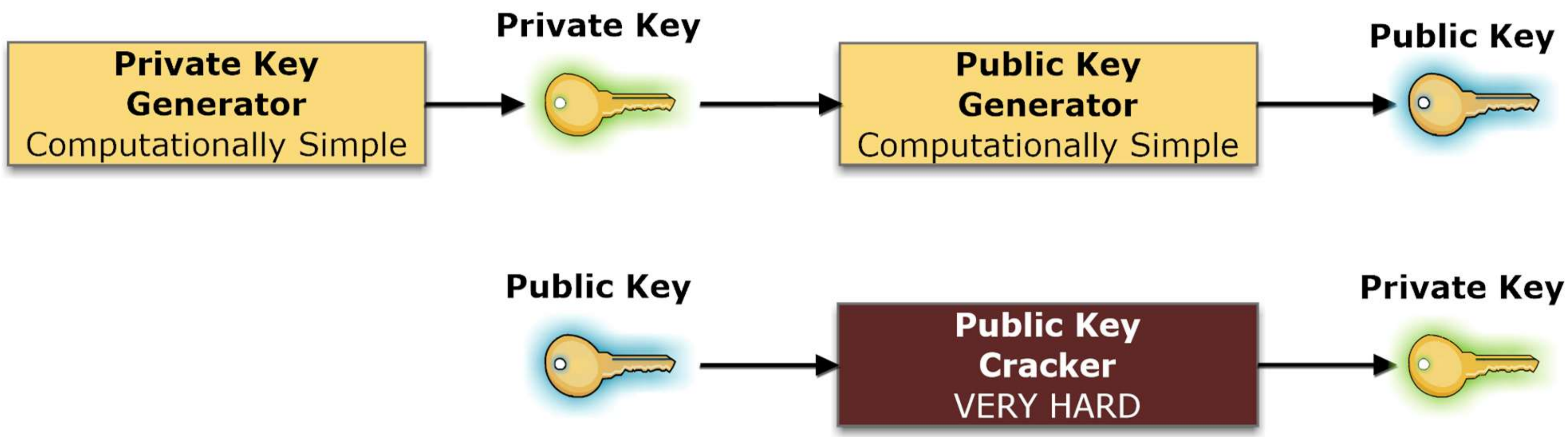
?????

***Public Key:* FLOWER**

***Private Key*:TIGERS**

https://www.iiesoc.in/ https://industrynetcouncil.org/

Key Generation



- In public-key systems, it is relatively easy to compute the public key from the private key, but very hard to compute the private key from the public key (which is the one everyone knows).
- In fact, some algorithms need several months (and even years) of constant computation to obtain the private key from the public key.
- The public key algorithm most often used is called the RSA algorithm which was invented in 1978 by Ron Rivest, Adi Shamir, and Leonard Adleman.
- The algorithms rely on the hard to invert function of factoring prime numbers.

Hard to Invert Functions

- What you want is a function which is hard to undo.
- That is, if you give me a number, I can compute the result easily. But if I give you just the result, you can't tell me the original number very easily.

Easy to Invert	Hard to Invert
Function 1: **You give me 2** **I give you 4.** **You give me 5.** **I give you 10.** **You give me 9.** **I give you 18.** **Now, to invert it, if I give you 22, what is the answer?**	**Function 2:** **You give me 2** **I give you 1.** **You give me 5.** **I give you 2.** **You give me 56.** **I give you 35.** **Now, if I give you 3, what is the answer? It is not so easy to see the pattern.**

Hard to Invert Functions

- What you want is a function which is hard to undo.
- That is, if you give me a number, I can compute the result easily. But if I give you just the result, you can't tell me the original number very easily.

Easy to Invert

Function 1:

You give me 2
I give you 4.
You give me 5.
I give you 10.
You give me 9.
I give you 18.

Now, to invert it, if I give you 22, you can quite easily tell me the answer. (11)!

Hard to Invert

Function 2:

You give me 2
I give you 1.
You give me 5.
I give you 2.
You give me 56.
I give you 35.

Now, if I give you 3, it is not so easy to see the pattern. I am using the modulus or remainder function.

Answers to above:

1. Start with 707 / 2 remainder =1
2. Start with 707 / 5 remainder = 2

Many possibilities for a number which will give a remainder of 3 when divided into 707.

https://www.iiesoc.in/

https://industrynetcouncil.org/

Factoring Prime Numbers

- A prime number (or a prime) has exactly two distinct divisors: 1 and itself.
- The smallest twenty-five prime numbers (all the prime numbers under 100) are:

  2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83,89, 97
- Prime factorization is a list of all the prime-number factors of a given number.
- The prime factorization does not include 1, but does include every copy of every prime factor. For instance, the prime factorization of 8 is 2×2×2, not just "2". Yes, 2 is the only factor, but you need three copies of it to multiply back to 8, so the prime factorization includes all three copies

Example of RSA

- P = 61 <- first prime number (destroy this after computing E and D)
- Q = 53 <- second prime number (destroy this after computing E and D)
- PQ = 3233 <- modulus (give this to others)
- E = 17 <- public exponent (give this to others)
- D = 2753 <- private exponent (keep this secret!)

- Your public key is (E,PQ).
- Your private key is D.

- The encryption function is: encrypt(T) = (T^E) mod PQ = (T^17) mod 3233
- The decryption function is: decrypt(C) = (C^D) mod PQ = (C^2753) mod 3233

- To encrypt the plaintext value 123, do this: encrypt(123) = (123^17) mod 3233 = 337587917446653715596592958817679803 mod 3233 = 855
- To decrypt the ciphertext value 855, do this: decrypt(855) = (855^2753) mod 3233 = 123

Cracking RSA?

- A challenge was placed in Martin Gardner's column in 1977 in Scientific American in which the readers were invited to factor
- C=114,381,625,757,888,867,669,235,779,976,146,612,010,218,296,721,242,362,562,561,842,935,706,935,245,733,897,830,597,123,563,958,705,058,989,075,147,599,290,026,879,543,541
- into its two prime number factors.
- The first solver was to win one hundred dollars.
- This was solved 17 years later in April 26, 1994, cracked by an international effort via the internet with the use of 600 volunteers, workstations, mainframes, and supercomputers. They attacked the number above for eight months before finding its factors.
- Today, the numbers used for c are much larger.

A one line change …

Transport Layer Security (TLS) Protocol Version 1.3: draft-02 : Remove support for static RSA and DH key exchange.

Why remove RSA?

RSA depends on prime number factorization and other mathematics which people thought could be broken only with brute force and other difficult attacks.

↓

We now have:

ROBOT (Return Of Bleichenbacher's Oracle Threat)	Theft of private keys	Quantum computing

ROBOT
Return Of Bleichenbacher's Oracle Threat

ROBOT is the return of a 19-year-old vulnerability that allows performing RSA decryption and signing operations with the private key of a TLS server.

We discovered that by using some slight variations, this vulnerability can still be used against many HTTPS hosts in today's Internet.

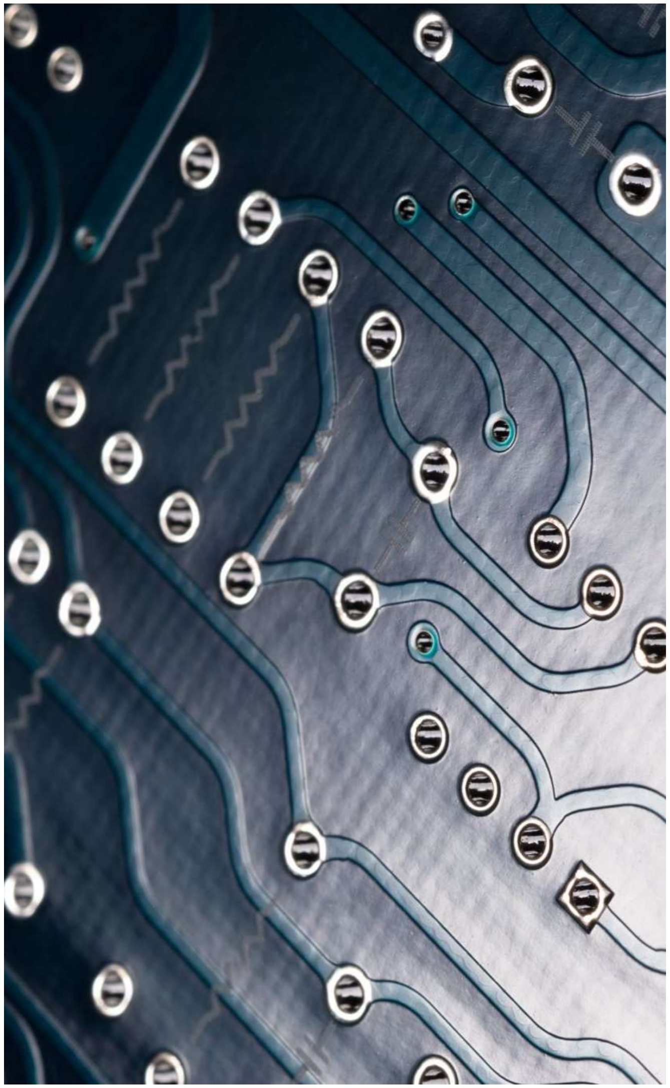
https://robotattack.org/

Theft of RSA private keys

- Why invent when you can steal?
- The Snowden revelations showed the compromise of chips used for encryption.
- Remember, in Public Key Encryption systems, keeping the private key secret is crucial.

Diagram from GCHQ slide

https://theintercept.com/2015/02/19/great-sim-heist/



Quantum Computing: What is the problem?

- A quantum computer will be developed in the next 5 – 10 years.
- This poses a threat to current modes of encryption.
- Protocols including ciphers for post-quantum crypto are being developed.

NIST: Post-Quantum Cryptography

Table 1 - Impact of Quantum Computing on Common Cryptographic Algorithms

Cryptographic Algorithm	Type	Purpose	Impact from large-scale quantum computer
AES	Symmetric key	Encryption	Larger key sizes needed
SHA-2, SHA-3	---------------	Hash functions	Larger output needed
RSA	Public key	Signatures, key establishment	No longer secure
ECDSA, ECDH (Elliptic Curve Cryptography)	Public key	Signatures, key exchange	No longer secure
DSA (Finite Field Cryptography)	Public key	Signatures, key exchange	No longer secure

NIST: **Report on Post-Quantum Cryptography (http://dx.doi.org/10.6028/NIST.IR.8105)**

What exactly is the threat?

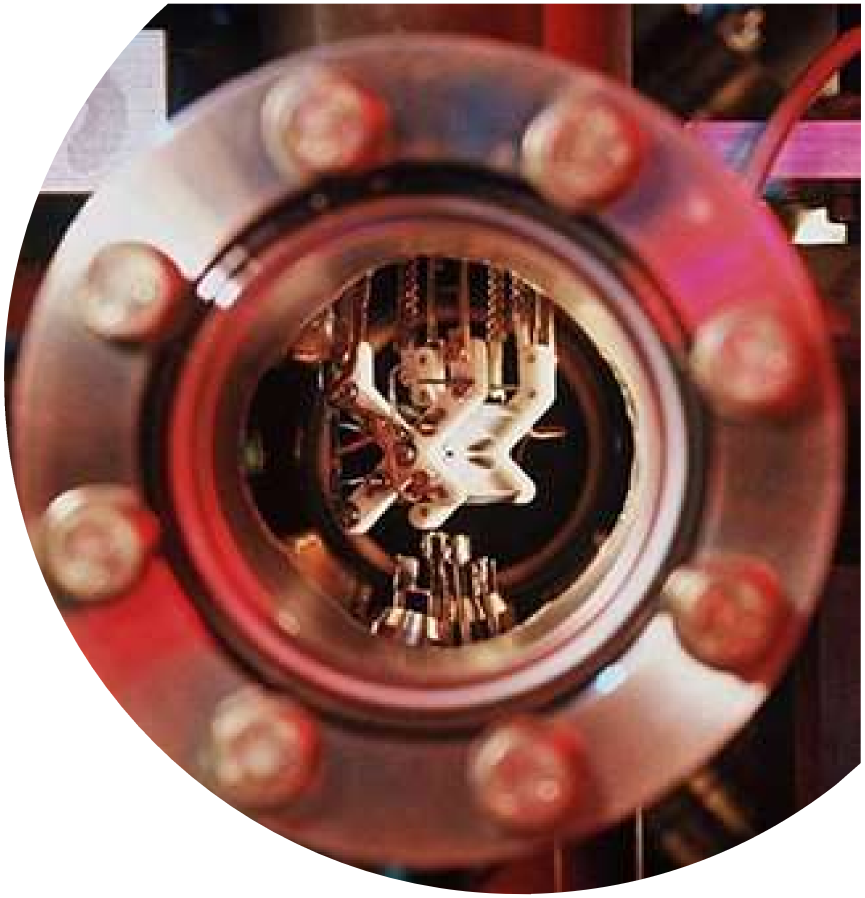
Adversary captures your network traffic and stores it

Adversary gets access to a powerful quantum computer.

Encryption is broken. You are toast.

Quantum Computing

A classical computer uses bits of information, 1s and 0s. A quantum computer uses what are called qubits, which can be a mix of both 1 and 0 simultaneously and which exist in a delicate quantum state called superposition.



http://physicsworld.com/cws/article/news/2016/mar/04/shors-algorithm-is-implemented-using-five-trapped-ions

http://spectrum.ieee.org/tech-talk/computing/hardware/encryptionbusting-quantum-computer-practices-factoring-in-scalable-fiveatom-experiment

Shor’s Algorithm

Peter Shor, an MIT math professor, developed an algorithm to factor large numbers with a quantum computer in 1994 but had no way to test it.

In 2001,Isaac Chuang, an MIT physicist and electrical engineer, managed to use this algorithm to factor the number 15, but the quantum system he used could not be scaled up to factor anything more complicated.

http://spectrum.ieee.org/tech-talk/computing/hardware/encryptionbusting-quantum-computer-practices-factoring-in-scalable-fiveatom-experiment

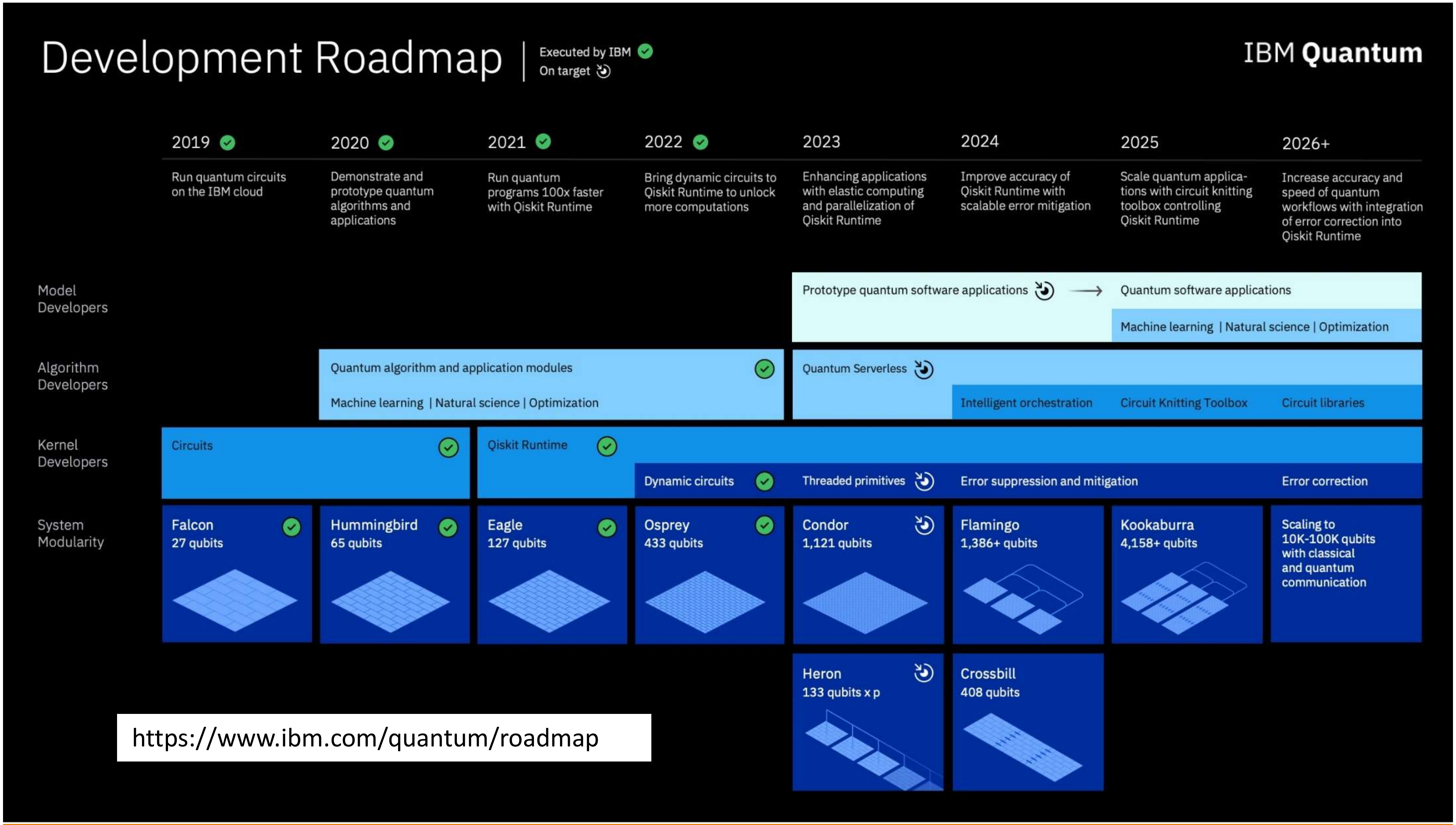
Quantum Factorization: Initial Progress

Table 5: *Quantum factorization records*

Number	# of factors	# of qubits needed	Algorithm	Year implemented	Implemented without prior knowledge of solution
15	2	8	Shor	2001 [2]	×
	2	8	Shor	2007 [3]	×
	2	8	Shor	2007 [3]	×
	2	8	Shor	2009 [5]	×
	2	8	Shor	2012 [6]	×
21	2	10	Shor	2012 [7]	×
143	2	4	minimization	2012 [1]	✓
56153	2	4	minimization	2012 [1]	✓
291311	2	6	minimization	not yet	✓
175	3	3	minimization	not yet	✓

Credit: Dattani and Bryans : More at: http://phys.org/news/2014-11-largest-factored-quantum-device.html

IBM Quantum Roadmap



https://www.ibm.com/quantum/roadmap

Google and other efforts

Google plans to spend several billion dollars to build a quantum computer by 2029 that can perform large-scale business and scientific calculations without errors, said Hartmut Neven, a distinguished scientist at Google who oversees the company's Quantum AI program. The company recently opened an expanded California-based campus focused on the effort, he said.

https://www.wsj.com/articles/google-aims-for-commercial-grade-quantum-computer-by-2029-11621359156

NIST: Symmetric Algorithms

Table 1 - Impact of Quantum Computing on Common Cryptographic Algorithms

Cryptographic Algorithm	Type	Purpose	Impact from large-scale quantum computer
AES	Symmetric key	Encryption	Larger key sizes needed
SHA-2, SHA-3	---------------	Hash functions	Larger output needed
RSA	Public key	Signatures, key establishment	No longer secure
ECDSA, ECDH (Elliptic Curve Cryptography)	Public key	Signatures, key exchange	No longer secure
DSA (Finite Field Cryptography)	Public key	Signatures, key exchange	No longer secure

Remember Brute Force Attacks!

- For any cipher, the most basic method of attack is brute force — trying every possible key in turn.
- The length of the key determines the number of possible keys.
- For post-quantum crypto, increase key size. (Will discuss in upcoming sessions)

Upcoming Sessions

Fundamentals of Cryptography: April 20, 11am Eastern, 8:30pm India

Fundamentals of Cryptography: May 18, 11am Eastern, 8:30pm India

Very next session! IPv6!

Introduction to Segment Routing and SRv6: March 9, 11 am Eastern 9:30 pm India

Questions?

Contact:

info@iiesoc.in

president@industrynetcouncil.org