DEVELOPMENT OF CRYPTOGRAPHIC TOOLS AND THEIR CRYPTANALYSIS

Project Report submitted in partial fulfillment of the requirement

for the degree of

Bachelor of Technology

in

Computer Science Engineering

under the Supervision of

Brig. (Retd.) S.P. Ghrera

By

Shivi Gandhi	**(091204)**
AdityaSrivastava	**(091268)**
AshmitaLucktoo	**(091327)**
Kanika Gupta	**(091258)**

to

Jaypee University of Information and Technology

Waknaghat, Solan – 173234, Himachal Pradesh

CERTIFICATE

This is to certify that the work entitled **"Development of Cryptographic Tools and Their Cryptanalysis"** submitted by **Shivi Gandhi (091204), Aditya Srivastava (091268),Ashmita Lucktoo (091327) and Kanika Gupta (091258),** in partial fulfillment for the award of degree of **Bachelor of Technology** of Jaypee University of Information Technology, Waknaghat has been carried out under my supervision.

This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Guide:

Name of Guide:	**Brig. (Retd.) S.P. Ghrera**
Designation:	**Head of Department**
Date:	**May 16, 2013**

ACKNOWLEDGEMENT

This project could not have been at the stage it is right now had it not been for the cooperation of Brig. (Retd.) S.P. Ghrera, our project guide, who was always there to tell us how to go about our project in a systematic manner and who always took out time to help us with our technical and non-technical doubts at various stages of the project.

Equally important was the contribution of Dr. Nitin, our project supervisor, who kept faith in our ability to complete the project well and on time.

Last but not the least; it was the teachers and Ph.D. students of the Jaypee University of Information Technology, Waknaghat, particularly Mr. Amol Vasudeva, Mr. Shiv K. Gupta, Mr. Amit Srivastava, Mrs. Anshul Sood and Ms. Rashmi who came to our rescue whenever we got stuck in any piece of code or otherwise.

Names of Students:

Shivi Gandhi (091204)

Aditya Srivastava (091268)

Ashmita Lucktoo (091327)

Kanika Gupta (091258)

Date:

May 16, 2013

CONTENTS

LIST OF FIGURES

ABSTRACT

Today, security is one of the foremost concerns in the transmission of data over the internet or over any other network. We will thus develop applications to implement various security services that will help the individuals in the network to maintain their safety and privacy. Also, the strength of these security solutions will be cryptanalyzed through study of various networking attacks and developing countermeasures against them like,

- Brute Force Method implemented on the basis of limiting the number of attempts,
- Brute Force Method implemented on the basis of imposing a time limit,
- RSA, 3DES, KDC

The strength of the cryptographic tools implemented will thus be tested against these implemented attacks.

While the cryptographic techniques have to be implemented all at once, the cryptanalysis is done one algorithm at a time, as only one type of attack will be launched by an attacker at one time.

USE CASE DIAGRAM

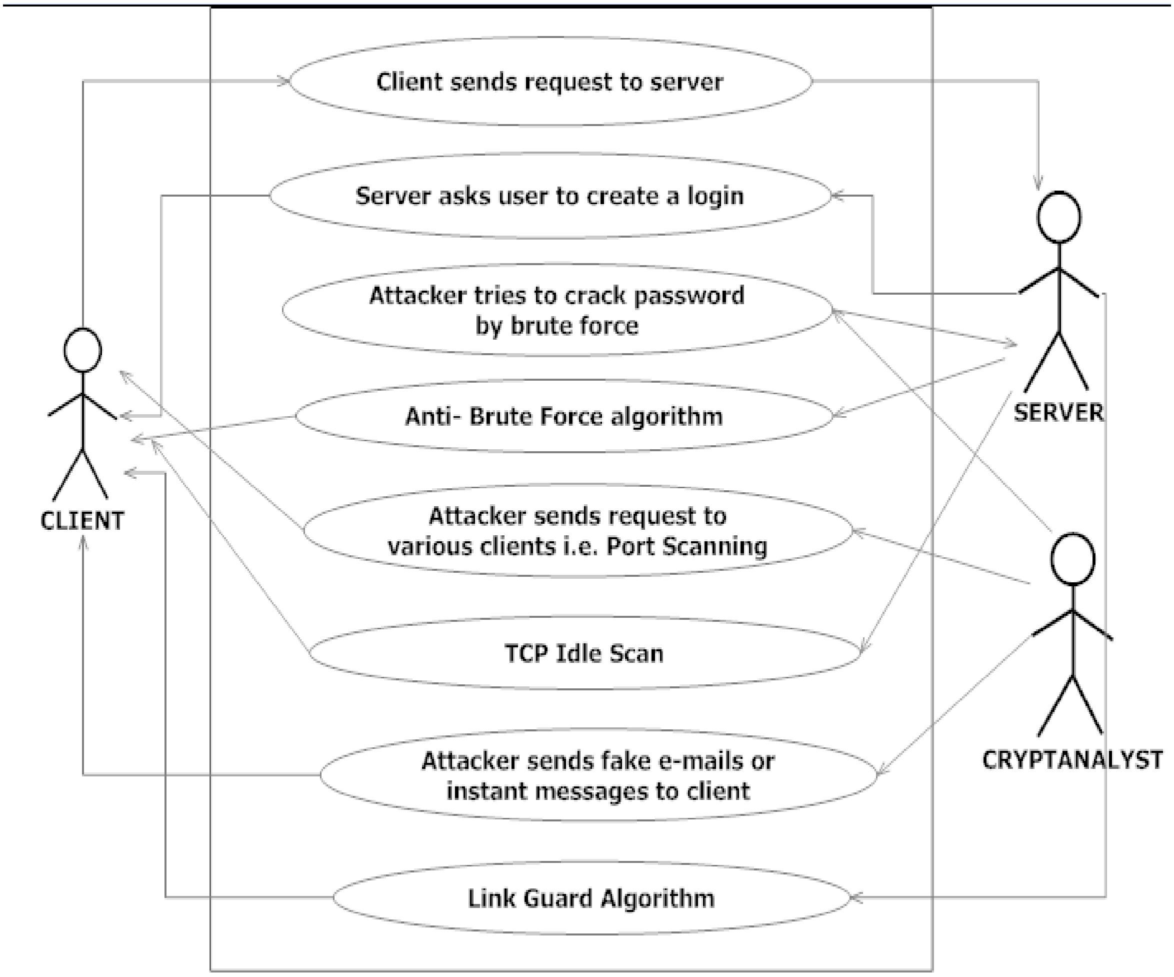

Client sends request to server
Server asks user to create a login
Attacker tries to crack password by brute force
Anti- Brute Force algorithm
Attacker sends request to various clients i.e. Port Scanning
TCP Idle Scan
Attacker sends fake e-mails or instant messages to client
Link Guard Algorithm
CLIENT
SERVER
CRYPTANALYST

Figure 1: Use case diagram

LITERATURE SURVEY

Phishing:

It is the act of sending an e-mail to a user falsely claiming to be an established legitimate enterprise, in order to scam the user into surrendering private information that can then be used for identity theft. The e-mail directs the user to visit a Web site where they are asked to update personal information, such as passwords and credit card, social security, and bank account numbers, that the legitimate organization already has.

List of phishing techniques:

- **Spear Phishing:** Phishing attempts directed at specific individuals or companies. Attackers may gather personal information about their target to increase their probability of success.

- **Clone Phishing:** A type of phishing attack whereby a legitimate, and previously delivered, email containing an attachment or link has had its content and recipient address(es) taken and used to create an almost identical or cloned email. The attachment or link within the email is replaced with a malicious version and then sent from an email address spoofed to appear to come from the original sender. It may claim to be a re-send of the original or an updated version to the original.

- **Whaling:**Several recent phishing attacks have been directed specifically at senior executives and other high profile targets within businesses, and the term whaling has been coined for these kinds of attacks.

LinkGuard Algorithm (Anti-phishing technique):

- In its main routine *LinkGuard, it first extracts the DNS names from the* actual and the visual links. It then compares the actual and visual DNS names, if these names are not the same, then it is of phishing category.

- If dotted decimal IP address is directly used in actual DNS, then a possible phishing attack.

- If the actual link or the visual link is encoded: first decode the links, then recursively call *LinkGuard to return a result.*

- *When there is* no destination information (DNS name or dotted IP address) in the visual link, LinkGuard calls *AnalyzeDNS to* analyze the actual DNS.

- LinkGuard therefore handles all the categories of phishing attacks.

Brute force attacks:

- **Manual login attempts**, they will try to type in a few usernames and passwords

- **Dictionary based attacks**, automated scripts and programs will try guessing thousands of usernames and passwords from a dictionary file, sometimes a file for usernames and another file for passwords

- **Generated logins**, a cracking program will generate random usernames set by the user. They could generate numbers only, a combination of numbers and letters or other combinations.

Few Main Ways to Stop a Brute Force Attack:

- restricting the amount of login attempts that a user can perform
- banning a user’s IP after multiple failed login attempts
- the most obvious way to block brute-force attacks is to simply lock out accounts after a defined number of incorrect password attempts
- Since the success of the attack is dependent on time, inject random pauses when checking a password. Adding even a few seconds' pause can greatly slow a brute-force attack but will not bother most legitimate users as they log in to their accounts
- to lock out an IP address with multiple failed logins
- after one or two failed login attempts, you may want to prompt the user not only for the username and password but also to answer a secret question
- for advanced users who want to protect their accounts from attack, give them the option to allow login only from certain IP addresses
- use a CAPTCHA to prevent automated attacks

Brute Force by limiting the number of attempts:

RealVNC Anti 3.x Brute Force Algorithm

```
if (strcmp (current ->machineName, Machine) == 0)

        {
        if (current-> blocked)
                return;

                current->lastRefTime.QuadPart  =  now.QuadPart + 10;

                current->failureCount++;

                if (current->failureCount> five)

                        current-> blocked = TRUE;

                return;

        }
```

Brute Force by imposing a time limit:

RealVNC 4.x Anti Brute Force algorithm

```
if ((*i).second.marks>=  THRESHOLD)

{
time_t  now = time (0);
if (now >= (* i). second.blockUntil)

        {
        (*i). second.blockUntil = now + (*i).second.blockTimeout;
        (*i).second.blockTimeout = (*i).second.blockTimeout * 2;
        return false;
}

        return True;

}

(*i).second.marks++;

return false;
```

Resource metering:

- Resource metering is a technique designed to restrict the repetition frequency of data submission to an application or host system. To be successful, a resource metering solution should enforce restrictions at the client-side and not consume additional resources at the server-side.
- The most practical method of implementing resource metering is through the use of cryptographic hashes. The use of a cryptographic hash in this fashion is sometimes referred to as requiring an “electronic payment” before processing the customer’s submission. In essence, the server-side application requires the customer’s client to compute a value that is computationally intensive, but easy to validate, before processing the submitted data.

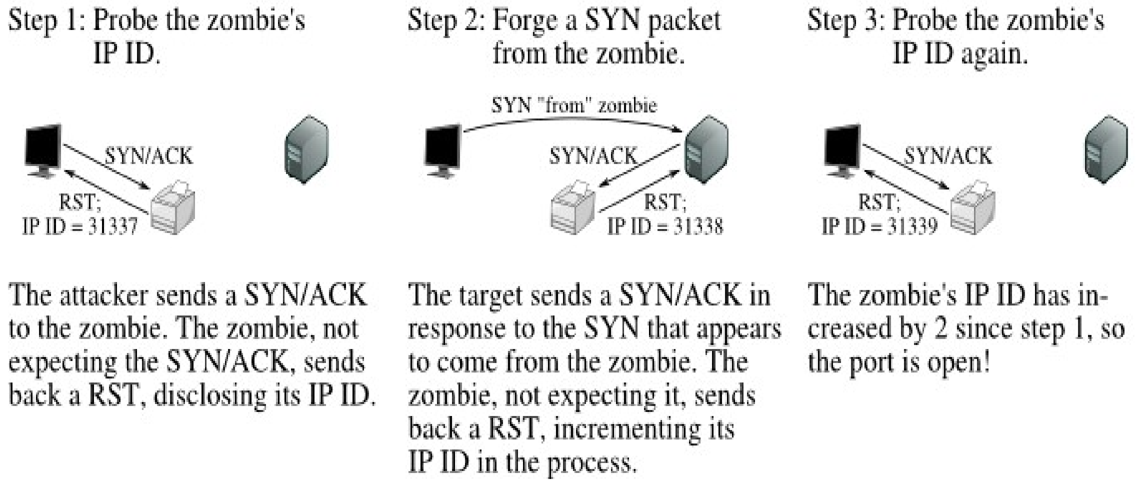
Port Scanning:

- The act of systematically scanning a computer's ports. Since a port is a place where information goes into and out of a computer, port scanning identifies open doors to a computer. Port scanning has legitimate uses in managing networks, but port scanning also can be malicious in nature if someone is looking for a weakened access point to break into your computer.

The following three diagrams show exactly what happens in the three cases of an open, closed, and filtered port.

the attacker, the zombie, and the target.

Idle scan of an open port:



Step 1: Probe the zombie's IP ID.

Step 2: Forge a SYN packet from the zombie.

Step 3: Probe the zombie's IP ID again.

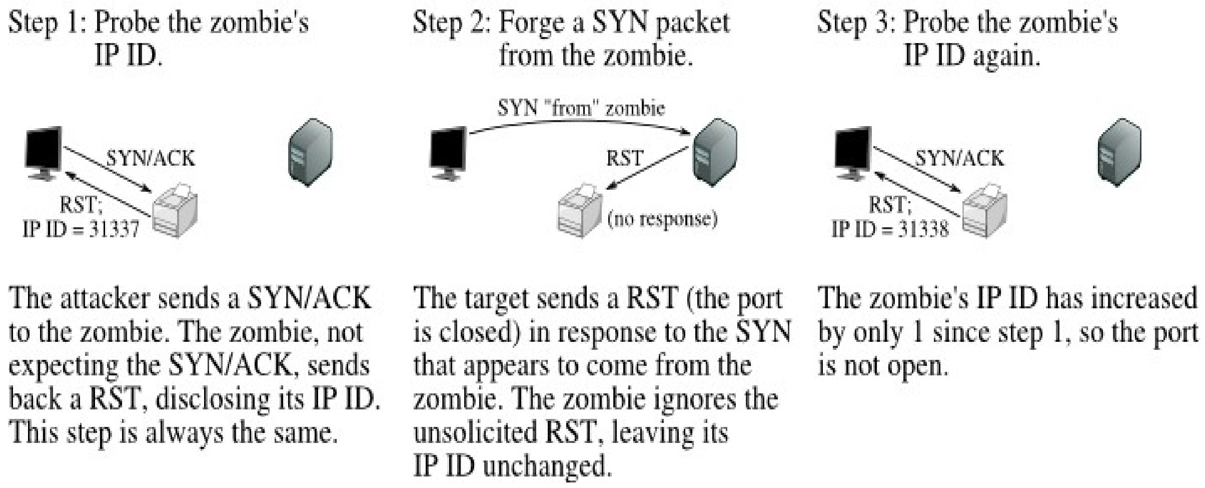
The attacker sends a SYN/ACK to the zombie. The zombie, not expecting the SYN/ACK, sends back a RST, disclosing its IP ID.

The target sends a SYN/ACK in response to the SYN that appears to come from the zombie. The zombie, not expecting it, sends back a RST, incrementing its IP ID in the process.

The zombie's IP ID has increased by 2 since step 1, so the port is open!

Figure 2: Idle scan of an open port

Idle scan of a closed port:



Step 1: Probe the zombie's IP ID.

Step 2: Forge a SYN packet from the zombie.

Step 3: Probe the zombie's IP ID again.

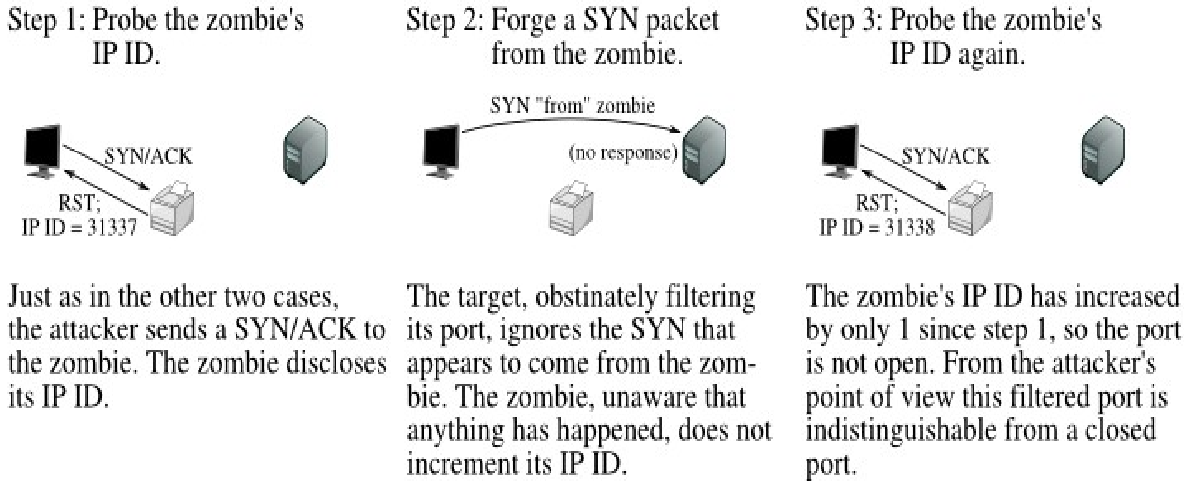
The attacker sends a SYN/ACK to the zombie. The zombie, not expecting the SYN/ACK, sends back a RST, disclosing its IP ID. This step is always the same.

The target sends a RST (the port is closed) in response to the SYN that appears to come from the zombie. The zombie ignores the unsolicited RST, leaving its IP ID unchanged.

The zombie's IP ID has increased by only 1 since step 1, so the port is not open.

Figure 3: Idle scan of a closed port

Idle scan of a filtered port:



Step 1: Probe the zombie's IP ID.

Step 2: Forge a SYN packet from the zombie.

Step 3: Probe the zombie's IP ID again.

Just as in the other two cases, the attacker sends a SYN/ACK to the zombie. The zombie discloses its IP ID.

The target, obstinately filtering its port, ignores the SYN that appears to come from the zombie. The zombie, unaware that anything has happened, does not increment its IP ID.

The zombie's IP ID has increased by only 1 since step 1, so the port is not open. From the attacker's point of view this filtered port is indistinguishable from a closed port.

Figure 4: Idle scan of a filtered port

Methods to avoid malicious use of Port Scanning:

- **TCP Idle Scan:** Idle scan allows for completely blind port scanning. Attackers can actually scan a target without sending a single packet to the target from their own IP address. Instead, a clever side-channel attack allows for the scan to be bounced off a dumb “zombie host”.

- **SYN Scan:** SYN scanning is a tactic that a malicious cracker can use to determine the state of a communications port without establishing a full connection. This approach, one of the oldest in the repertoire of crackers, is sometimes used to perform denial-of-service (DoS) attacks. SYN scanning is also known as half-open scanning.

- One way to determine whether a TCP port is open is to send a SYN (session establishment) packet to the port. The target machine will respond with a SYN/ACK (session request acknowledgment) packet if the port is open and RST (reset if the port is closed.

- A machine that receives an unsolicited SYN/ACK packet will respond with a RST. An unsolicited RST will be ignored.

- Every IP packet on the Internet has a fragment identification number (IP ID). Since many operating systems simply increment this number for each packet they send, probing for the IP ID can tell an attacker how many packets have been sent since the last probe.

Consumer Best Practices:

- **Automatically block malicious/fraudulent E-mail:** Spam detectors can help keep theconsumer from ever opening the suspicious E-mail, but they aren't fool proof.

- **Automatically detect and delete malicious software:** Spyware is often part of a phishingattack, but can be removed by many commercial programs.

- **Automatically block outgoing delivery of sensitive information to malicious parties:** Even if the consumer can't visually identify the true web site that will receive sensitive information, there are software products that can.

- **Be suspicious:** If you aren't sure if an E-mail is legitimate, call the apparent sending institution to verify the authenticity.

None of these remedies individually provides a complete answer to the problem. A combination of countermeasures is recommended that will:

- minimize the number of phishing attacks delivered to consumers;

- increase the likelihood that the consumer will recognize a phishing attack; and

- minimize the opportunities for the consumer to inadvertently release sensitive information.

Education remains critical so consumers are aware of both the phishing techniques and how legitimate entities will communicate with them via E-mail and the web.

Fundamentally, an idle scan consists of three steps that are repeated for each port:

- Probe the zombie's IP ID and record it.
- Forge a SYN packet from the zombie and send it to the desired port on the target. Depending on the port state, the target's reaction may or may not cause the zombie's IP ID to be incremented.
- Probe the zombie's IP ID again. The target port state is then determined by comparing this new IP ID with the one recorded in step 1.
- After this process, the zombie's IP ID should have increased by either one or two. An increase of one indicates that the zombie hasn't sent out any packets, except for its reply to the attacker's probe. This lack of sent packets means that the port is not open (the target must have sent the zombie either a RST packet, which was ignored, or nothing at all). An increase of two indicates that the zombie sent out a packet between the two probes. This extra packet usually means that the port is open (the target presumably sent the zombie a SYN/ACK packet in response to the forged SYN, which induced a RST packet from the zombie). Increases larger than two usually signify a bad zombie host. It might not have predictable IP ID numbers, or might be engaged in communication unrelated to the idle scan.

-

TRIPLE-DES:

The most widely used private key block cipher, is the Data Encryption Standard (DES). It was adopted in 1977 by the National Bureau of Standards as Federal Information Processing Standard 46 (FIPS PUB 46). DES encrypts data in 64-bit blocks using a 56-bit key. The DES enjoys widespread use. It has also been the subject of much controversy its security.

Symmetric key cryptography:

A symmetric encryption scheme has five ingredients:

- **Plaintext:** This is the original intelligible message or data that is fed into the algorithm as input.
- **Encryption algorithm:** The encryption algorithm performs various substitutions and transformations on the plaintext.
- **Secret key:** The secret key is also input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key being used at the time. The exact substitutions and transformations performed by the algorithm depend on the key.
- **Ciphertext:** This is the scrambled message produced as output. It depends on the plaintext and the secret key. For a given message, two different keys will produce two different ciphertexts. The ciphertext is an apparently random stream of data and, as it stands, is unintelligible.
- **Decryption algorithm:** This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the secret key and produces the original plaintext.

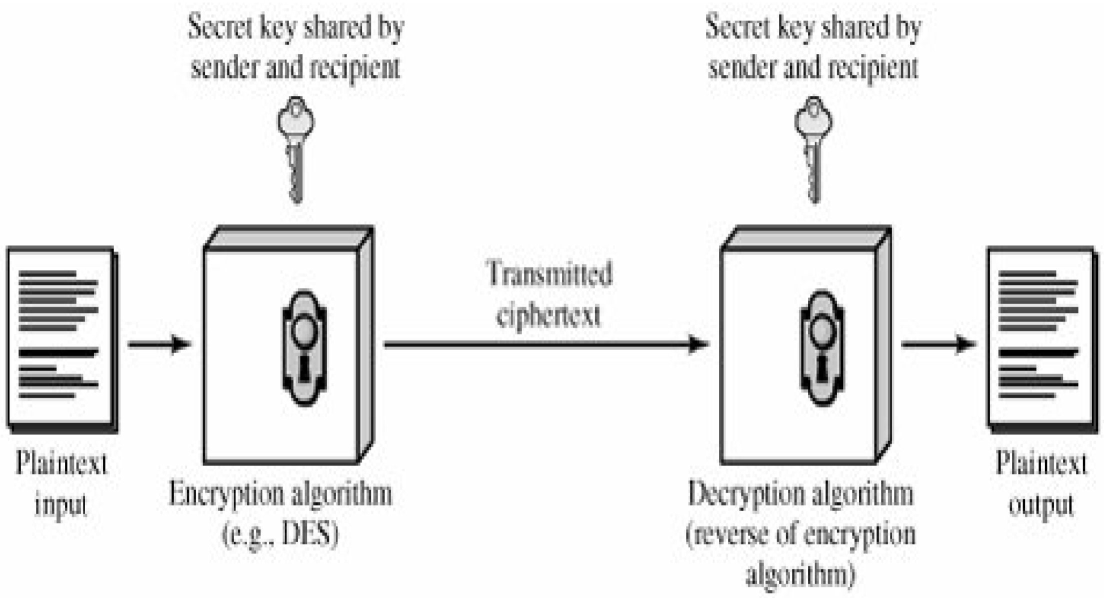

Secret key shared by sender and recipient
Secret key shared by sender and recipient
Transmitted ciphertext
Plaintext input
Encryption algorithm (e.g., DES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

Figure 5: Simplified Model of Conventional Encryption

DES Design Controversy:

Before its adoption as a standard, the proposed DES was subjected to intense & continuing criticism over the size of its key & the classified design criteria.

Recent analysis has shown despite this controversy, that DES is well designed. DES is theoretically broken using Differential or Linear Cryptanalysis but in practise is unlikely to be a problem yet. Also rapid advances in computing speed though have rendered the 56 bit key susceptible to exhaustive key search, as predicted by Diffie& Hellman.

DES has flourished and is widely used, especially in financial applications. It is still standardized for legacy systems, with either AES or triple DES for new applications.

Strength of DES – Key Size:

Since its adoption as a federal standard, there have been lingering concerns about the level of security provided by DES in two areas: key size and the nature of the algorithm.

With a key length of 56 bits, there are 2^56 possible keys, which is approximately 7.2*10^16 keys. Thus a brute-force attack appeared impractical.

However DES was finally and definitively proved insecure in July 1998, when the Electronic Frontier Foundation (EFF) announced that it had broken a DES encryption using a special-purpose "DES cracker" machine that was built for less than $250,000. The attack took less than three days. The EFF has published a detailed description of the machine, enabling others to build their own cracker [EFF98].

There have been other demonstrated breaks of the DES using both large networks of computers & dedicated h/w, including:

- 1997 on a large network of computers in a few months
- 1998 on dedicated h/w (EFF) in a few days
- 1999 above combined in 22hrs!

It is important to note that there is more to a key-search attack than simply running through all possible keys. Unless known plaintext is provided, the analyst must be able to recognize plaintext as plaintext.

Clearly must now consider alternatives to DES, the most important of which are AES and triple DES.

Possible analytic attacks:

Another concern is the possibility that cryptanalysis is possible by exploiting the characteristics of the DES algorithm.

The focus of concern has been on the eight substitution tables, or S-boxes, that are used in each iteration. These techniques utilise some deep structure of the cipher by gathering information about encryptions so that eventually you can recover some/all of the sub-key bits, and then exhaustively search for the rest if necessary. Generally these are statistical attacks which depend on the amount of information gathered for their likelihood of success.

Attacks of this form include the following

- Timing attack
- differential cryptanalysis,
- linear cryptanalysis and
- related key attacks.

Diagrammatic Overview:

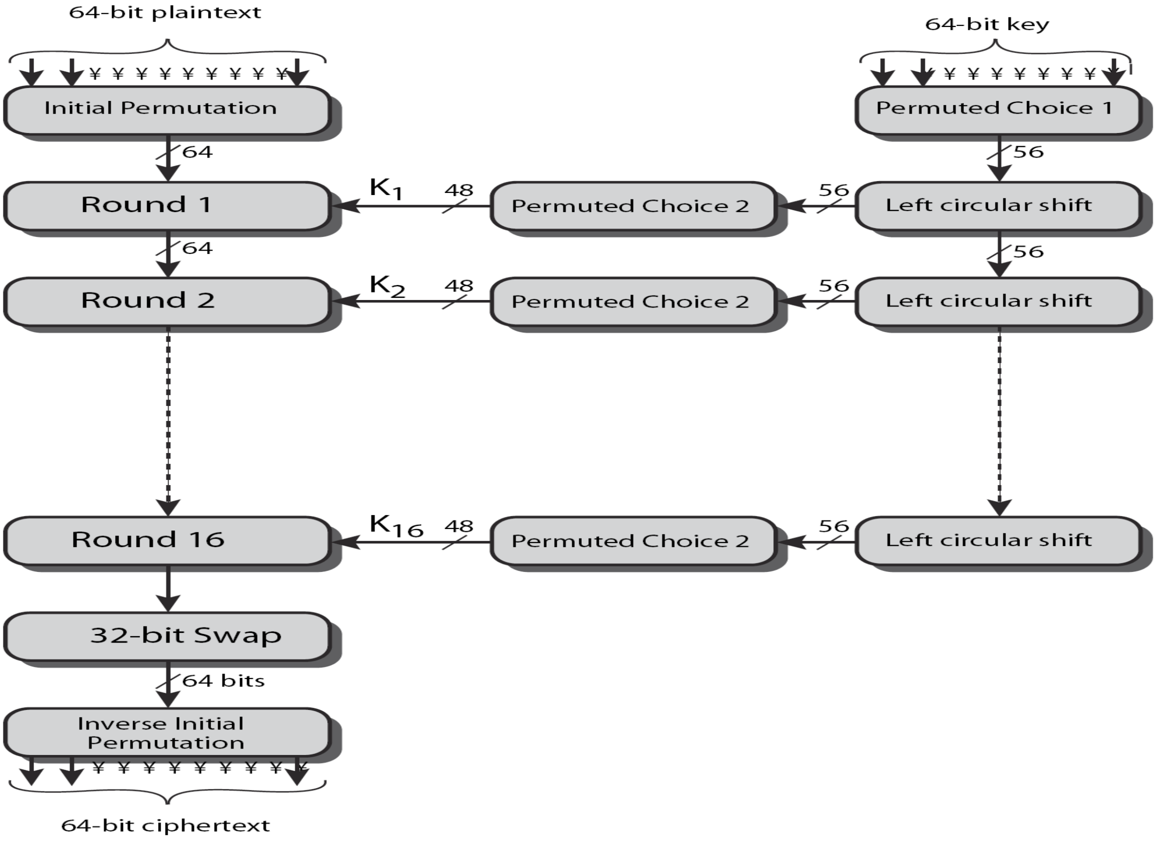


Figure 6: DES overview

Triple DES overview:

Triple-DES with two keys is a popular alternative to single-DES, but suffers from being 3 times slower to run. The use of encryption & decryption stages are equivalent, but the chosen structure allows for compatibility with single-DES implementations. 3DES with two keys is a relatively popular alternative to DES and has been adopted for use in the key management standards ANS X9.17 and ISO 8732. Currently, there are no practical cryptanalytic attacks on 3DES. Coppersmith notes that the cost of a brute-force key search on 3DES is on the order of 2^112 (=5*10^33) and estimates that the cost of differential cryptanalysis suffers an exponential growth, compared to single DES, exceeding 10^52.

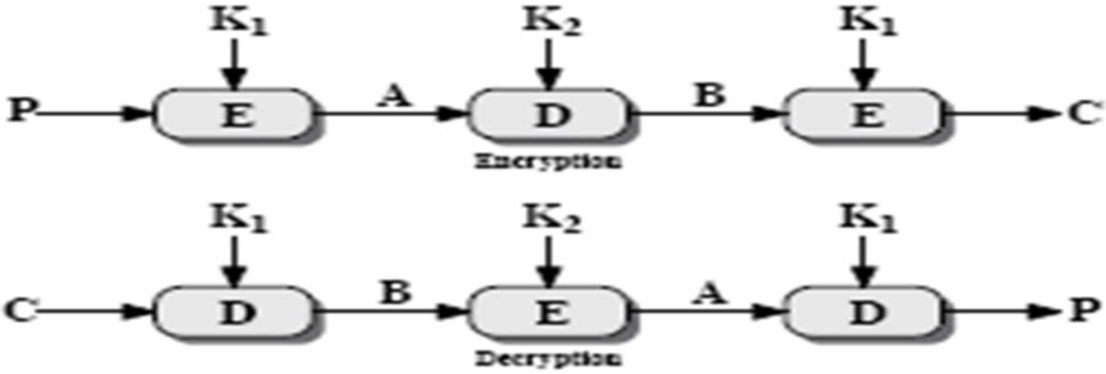


Figure 7: Triple-DES overview

RSA

RSA is the best known, and by far the most widely used general public key encryption algorithm, and was first published by Rivest, Shamir &Adleman of MIT in 1978 [RIVE78]. Since that time RSA has reigned supreme as the most widely accepted and implemented general-purpose approach to public-key encryption. It is based on exponentiation in a finite (Galois) field over integers modulo a prime, using large integers (eg. 1024 bits). Its security is due to the cost of factoring large numbers.

RSA Security:

Possible approaches to attacking RSA are:

- brute force key search (infeasible given size of numbers)
- mathematical attacks (based on difficulty of computing ø(n), by factoring modulus n)
- timing attacks (on running of decryption)
- chosen ciphertext attacks (given properties of RSA)

Public-Key Cryptography:

Public-key/two-key/asymmetric cryptography involves the use of **two** keys:

- a **public-key**, which may be known by anybody, and can be used to **encrypt messages**, and **verify signatures**
- a **private-key**, known only to the recipient, used to **decrypt messages**, and **sign** (create) **signatures**

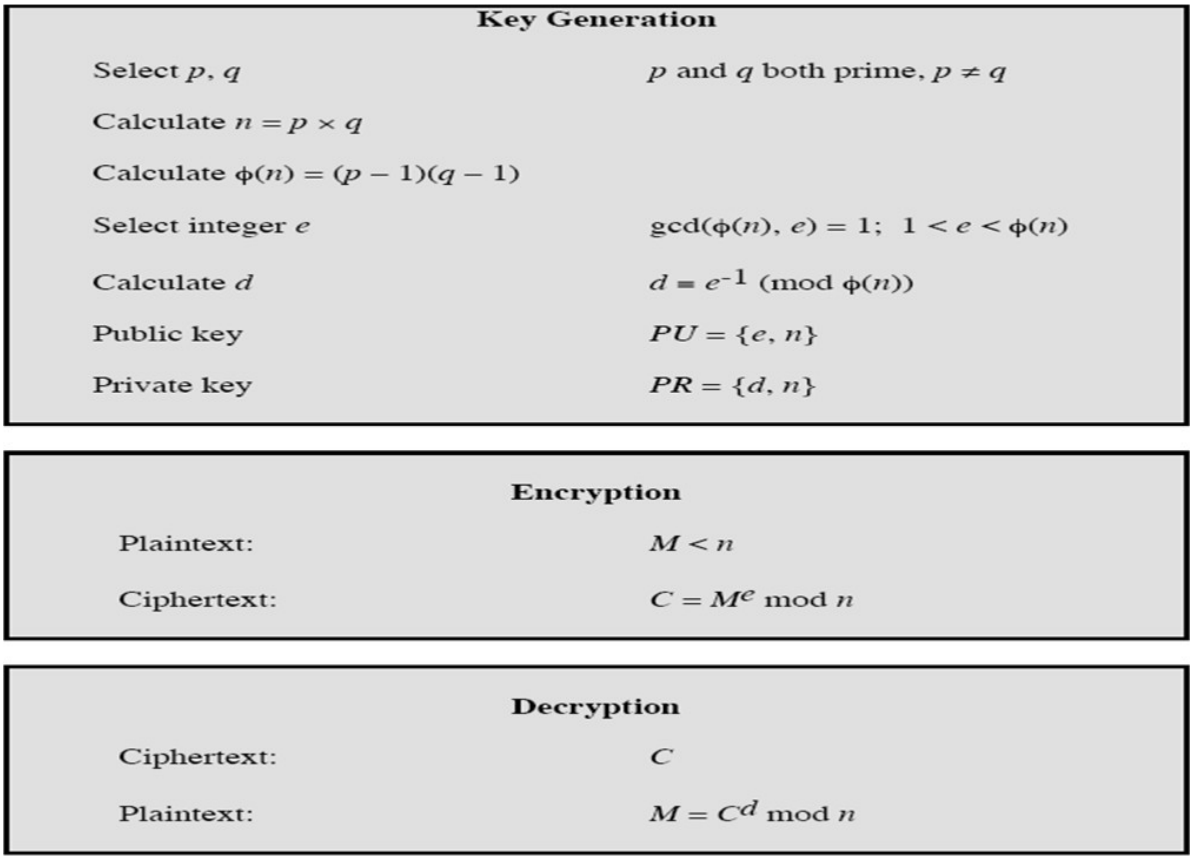
It is considered**asymmetric** becausethose who encrypt messages or verify signatures **cannot** decrypt messages or create signatures

Public-Key algorithms rely on two keys where:

- it is computationally infeasible to find decryption key knowing only algorithm & encryption key
- it is computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
- either of the two related keys can be used for encryption, with the other used for decryption (for some algorithms)

RSA Key Setup:

RSA key setup is done once (rarely) when a user establishes (or replaces) their public key, using the steps as shown. The exponent e is usually fairly small, just must be relatively prime to ø(n). Need to compute its inverse mod ø(n) to find d. It is critically important that the factors p & q of the modulus n are kept secret, since if they become known, the system can be broken. Note that different users will have different moduli-n.



Key Generation

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calculate d	$d = e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption

Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption

Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

Figure 8: RSA overview

KEY DISTRIBUTION CENTRE

The strength of any cryptographic system depends on the key distribution technique. For two parties A and B, key distribution can be achieved in a number of ways:

- A can select key and physically deliver to B
- third party can select & deliver key to A & B
- if A & B have communicated previously can use previous key to encrypt a new key
- if A & B have secure communications with a third party C, C can relay key between A & B

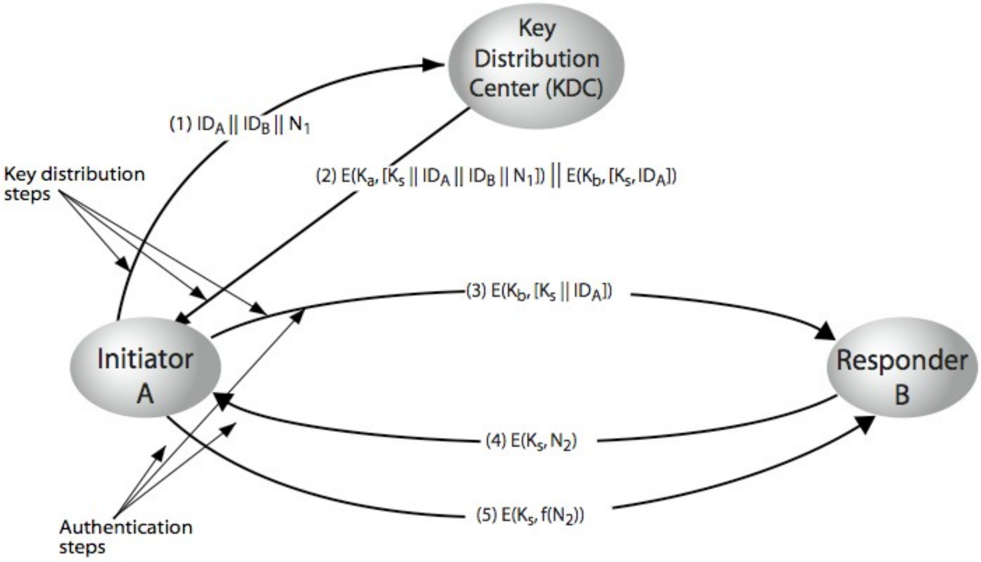

Key Distribution Center (KDC)
(1) $ID_A \| ID_B \| N_1$
Key distribution steps
(2) $E(K_a, [K_s \| ID_A \| ID_B \| N_1]) \| E(K_b, [K_s, ID_A])$
(3) $E(K_b, [K_s \| ID_A])$
Initiator A
Responder B
(4) $E(K_s, N_2)$
(5) $E(K_s, f(N_2))$
Authentication steps

Figure 9: Key Distribution Centre

CLIENT SIDE FLOW CHART

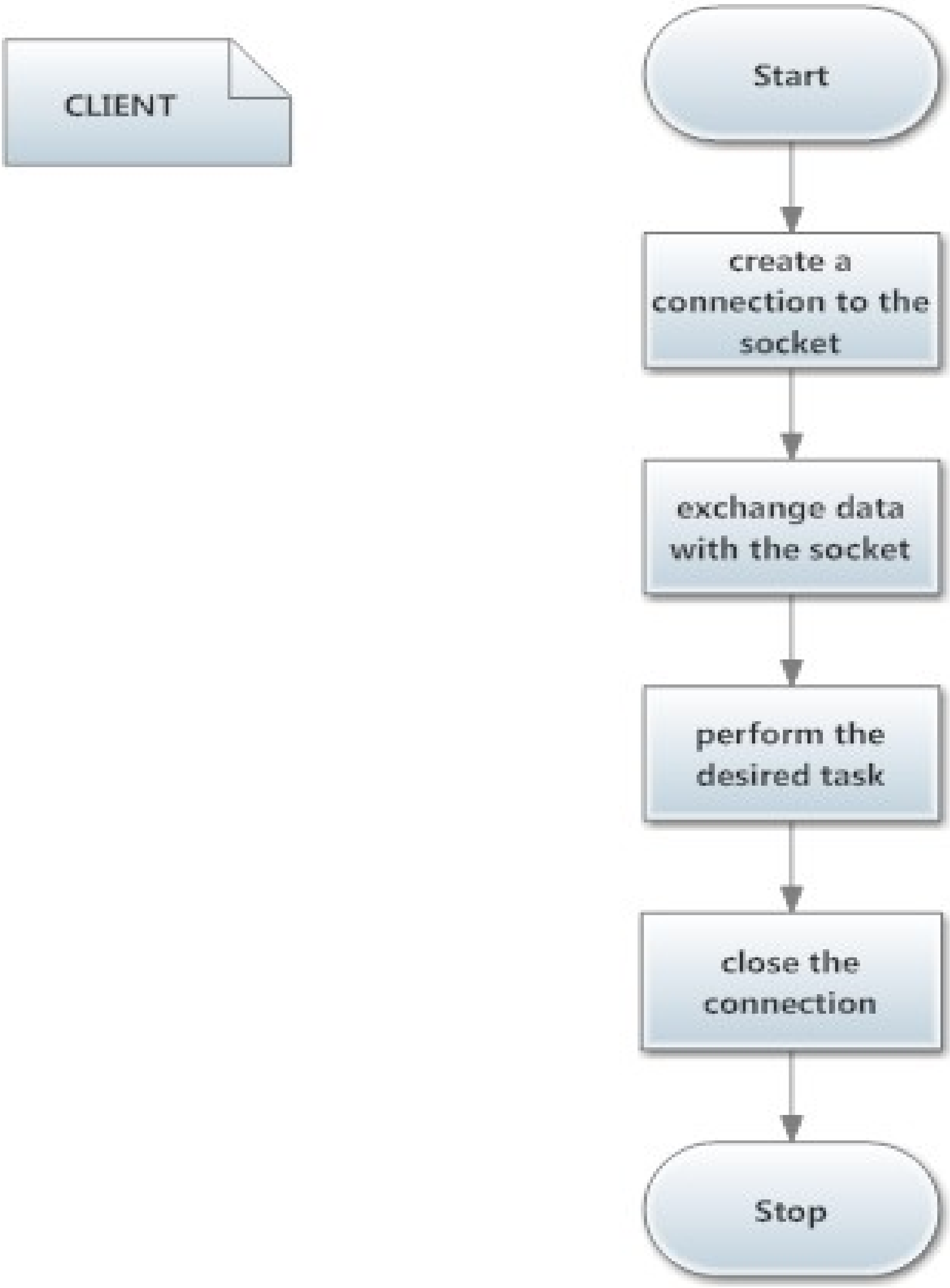
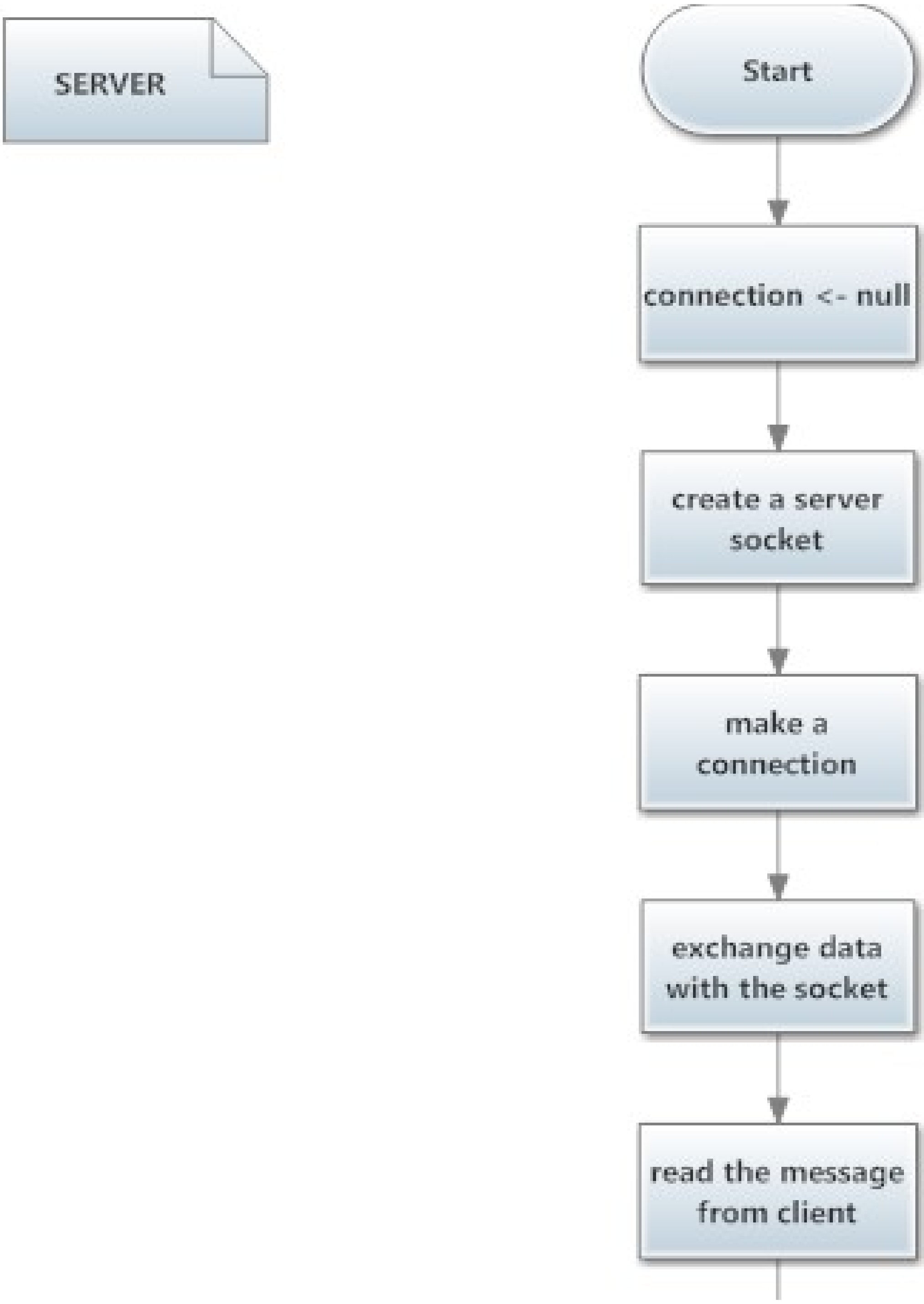

CLIENT
Start
create a connection to the socket
exchange data with the socket
perform the desired task
close the connection
Stop

Figure 10: Client side flow chart

SERVER SIDE FLOW CHART



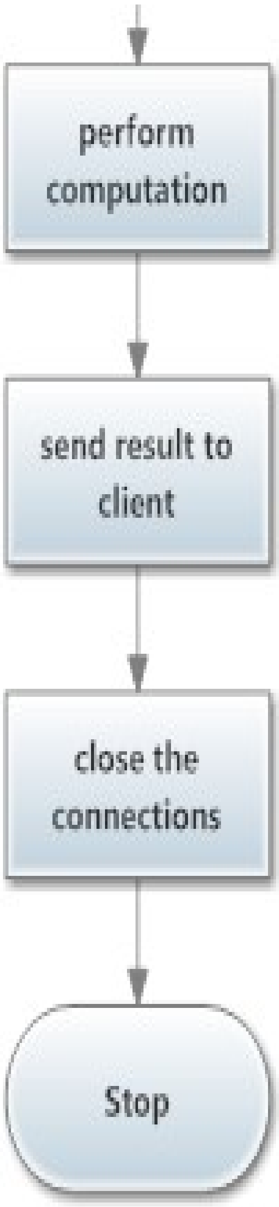
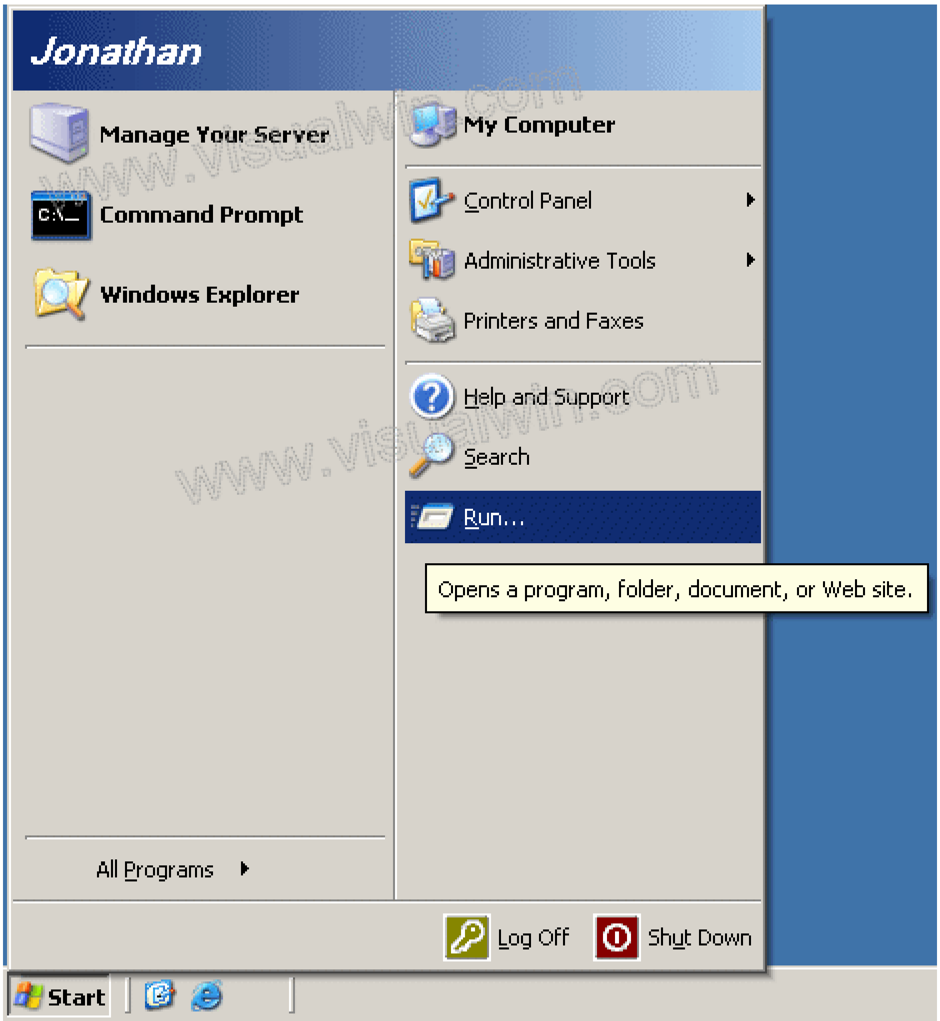


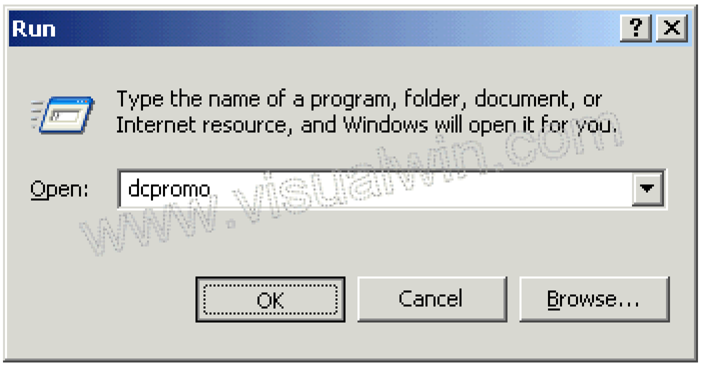
Figure 11: Server side flow chart

WINDOWS SERVER 2003

- It includes all the functionality customers expect from a critical Server operating system, such as security, reliability, availability, and scalability.
- We created a domain controller (DC) in the network which includes DNS server setup in windows server 2003 .We installed DNS server for DC, without which the client computers wouldn’t know which one is DC. Some of its most well-known features are its ability to store user names and passwords on a central computer (the Domain Controller)
- We have assigned a static IP address to our server.
- Creating new user(s): One of the “administrator’s group” and one “regular” user.
- It is not necessary to create a secondary account, but it is recommended in case you stay logged on, and someone gains control of the desktop (locally or remotely).

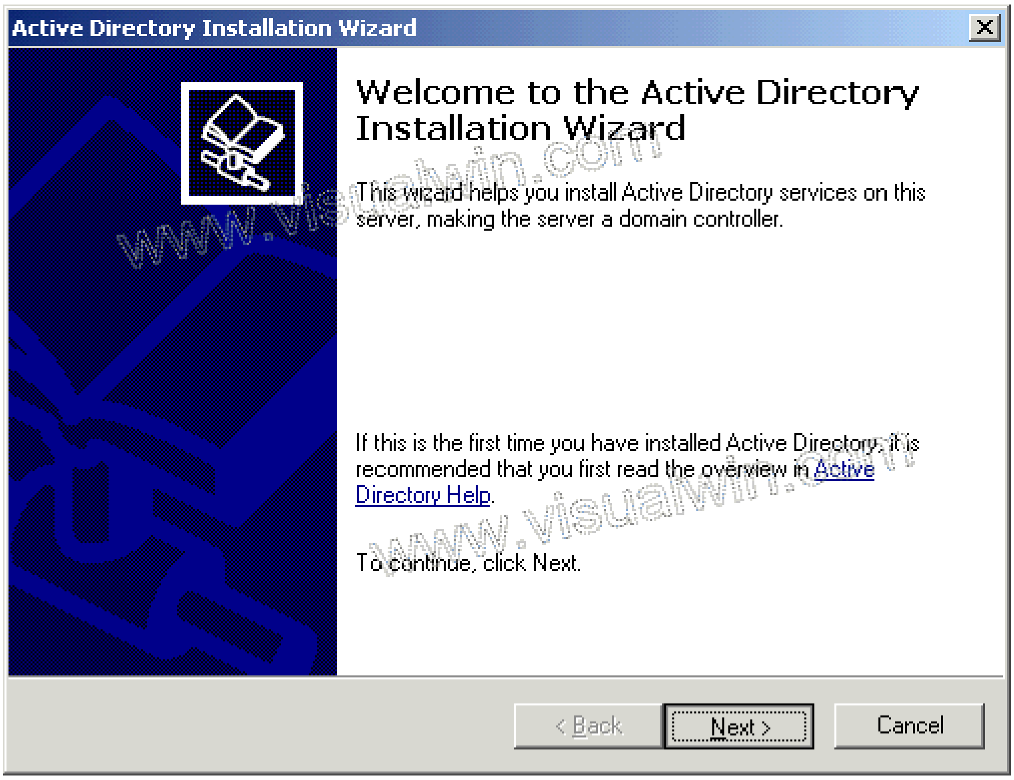
Creating the first windows server 2003 domain controller in a domain





Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: dcpromo

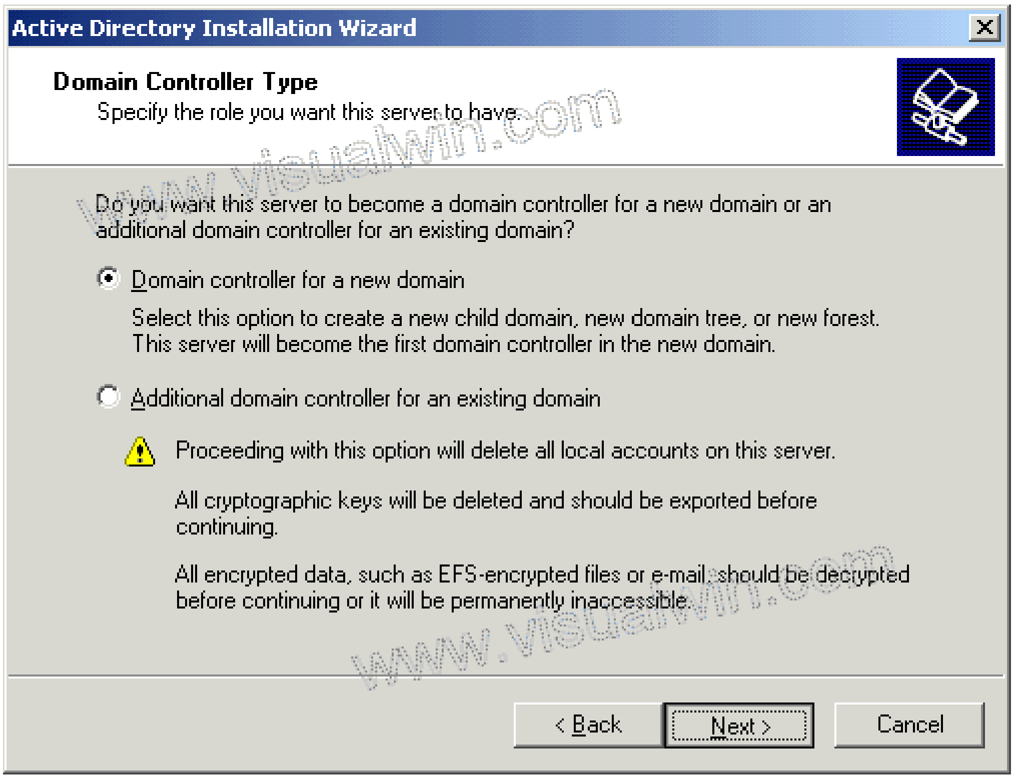


Welcome to the Active Directory Installation Wizard

This wizard helps you install Active Directory services on this server, making the server a domain controller.

If this is the first time you have installed Active Directory, it is recommended that you first read the overview in Active Directory Help.

To continue, click Next.



Active Directory Installation Wizard

Domain Controller Type

Specify the role you want this server to have.

Do you want this server to become a domain controller for a new domain or an additional domain controller for an existing domain?

● Domain controller for a new domain

Select this option to create a new child domain, new domain tree, or new forest. This server will become the first domain controller in the new domain.

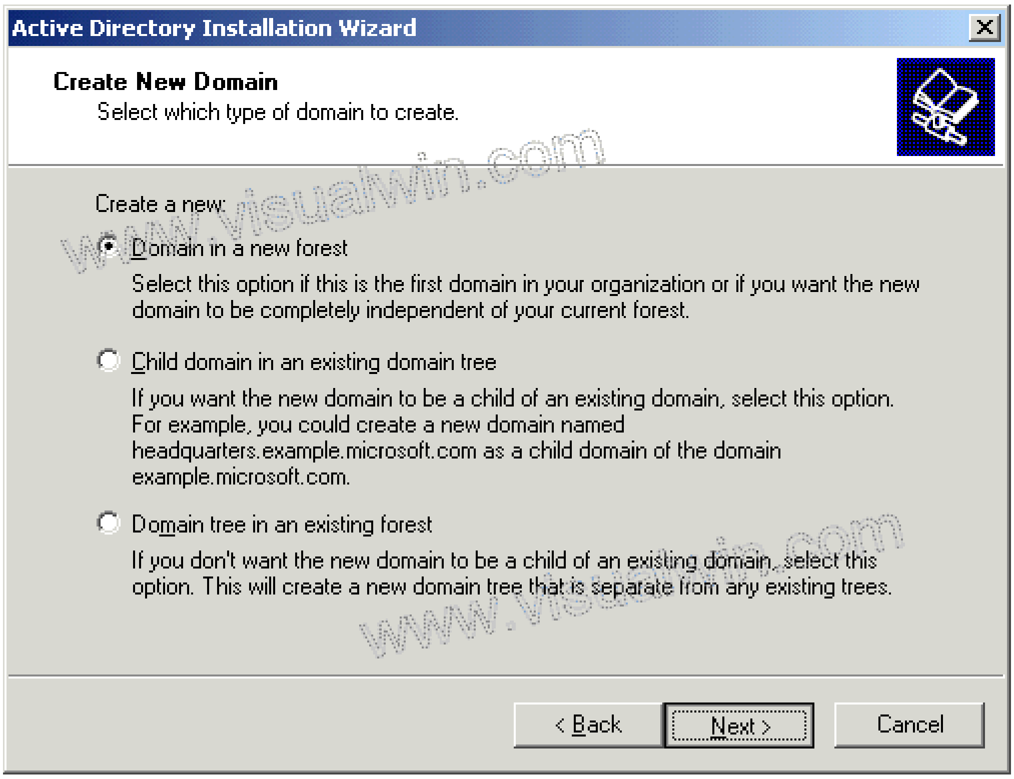
○ Additional domain controller for an existing domain

⚠ Proceeding with this option will delete all local accounts on this server.

All cryptographic keys will be deleted and should be exported before continuing.

All encrypted data, such as EFS-encrypted files or e-mail, should be decrypted before continuing or it will be permanently inaccessible.

We create a domain in a new forest, because it is the first DC



Active Directory Installation Wizard

Create New Domain
Select which type of domain to create.

Create a new:

● Domain in a new forest

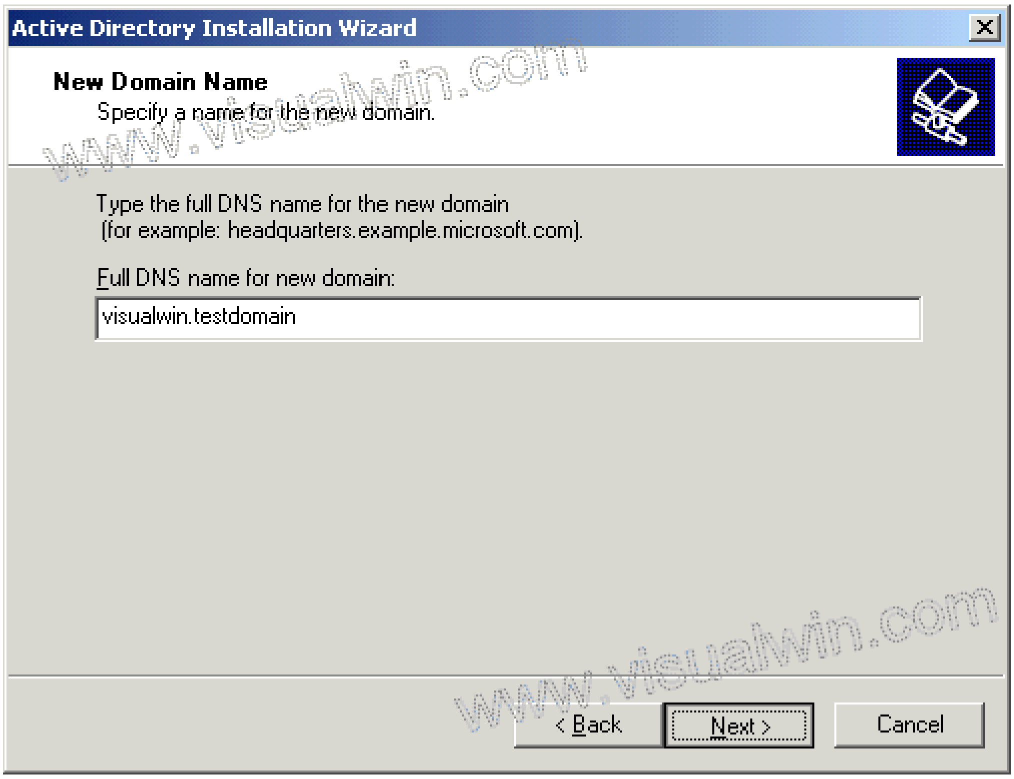
Select this option if this is the first domain in your organization or if you want the new domain to be completely independent of your current forest.

○ Child domain in an existing domain tree

If you want the new domain to be a child of an existing domain, select this option. For example, you could create a new domain named headquarters.example.microsoft.com as a child domain of the domain example.microsoft.com.

○ Domain tree in an existing forest

If you don't want the new domain to be a child of an existing domain, select this option. This will create a new domain tree that is separate from any existing trees.

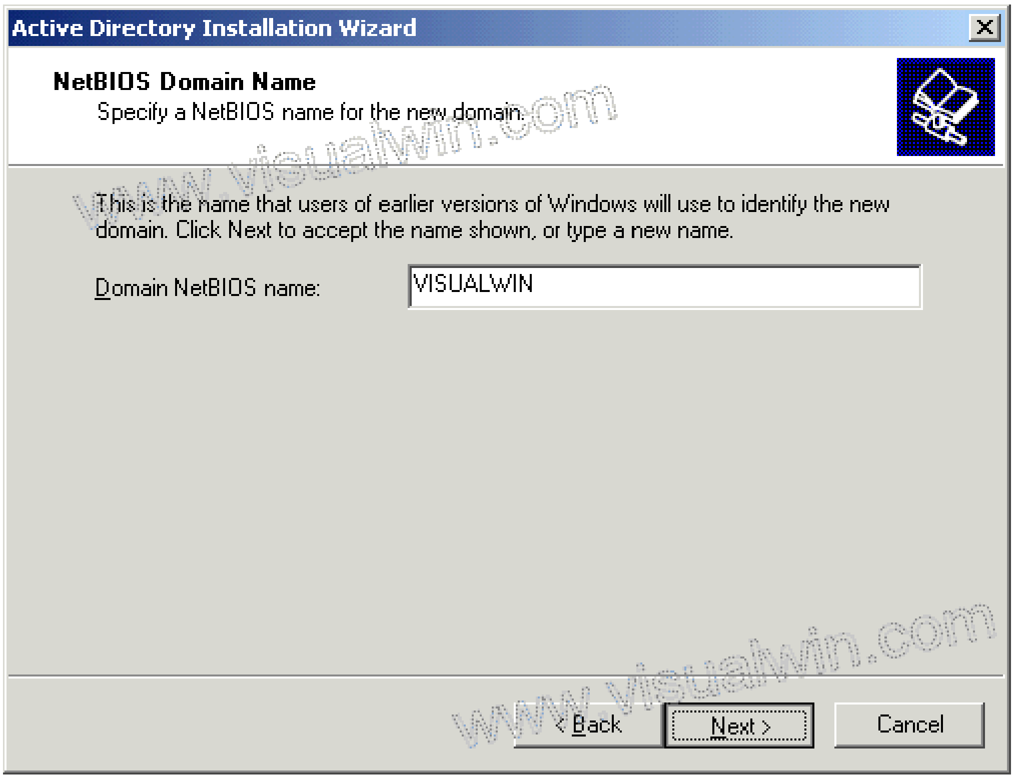


Active Directory Installation Wizard

New Domain Name
Specify a name for the new domain.

Type the full DNS name for the new domain
(for example: headquarters.example.microsoft.com).

Full DNS name for new domain:
visualwin.testdomain



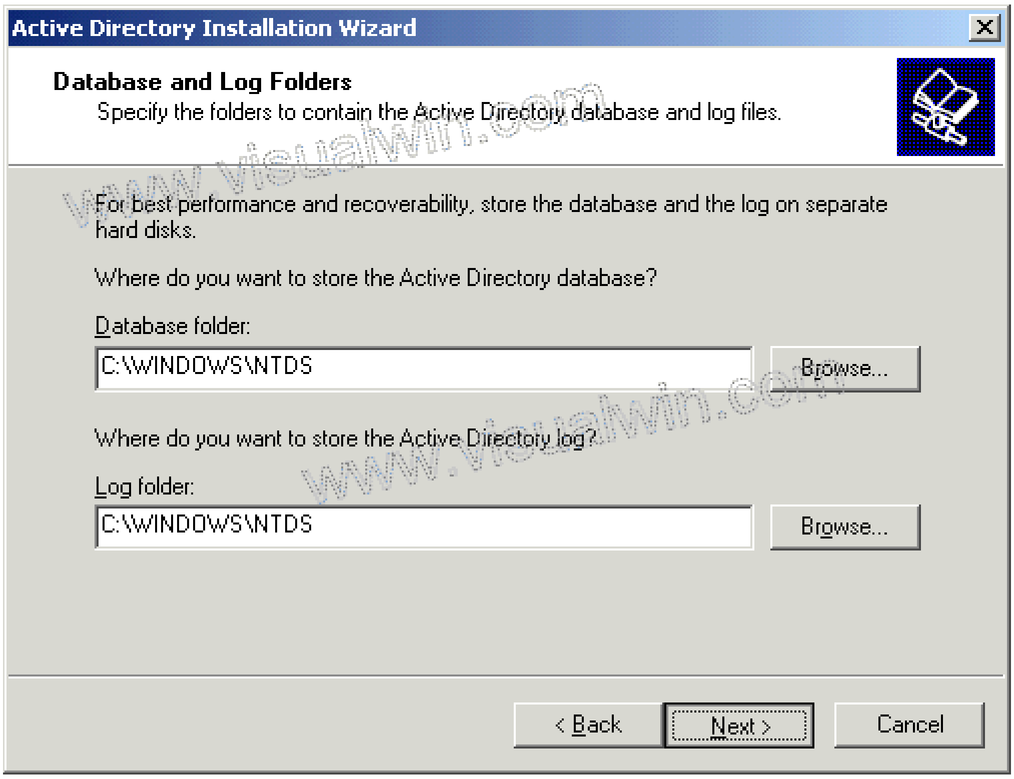
Active Directory Installation Wizard

NetBIOS Domain Name

Specify a NetBIOS name for the new domain.

This is the name that users of earlier versions of Windows will use to identify the new domain. Click Next to accept the name shown, or type a new name.

Domain NetBIOS name: VISUALWIN



Active Directory Installation Wizard

Database and Log Folders

Specify the folders to contain the Active Directory database and log files.

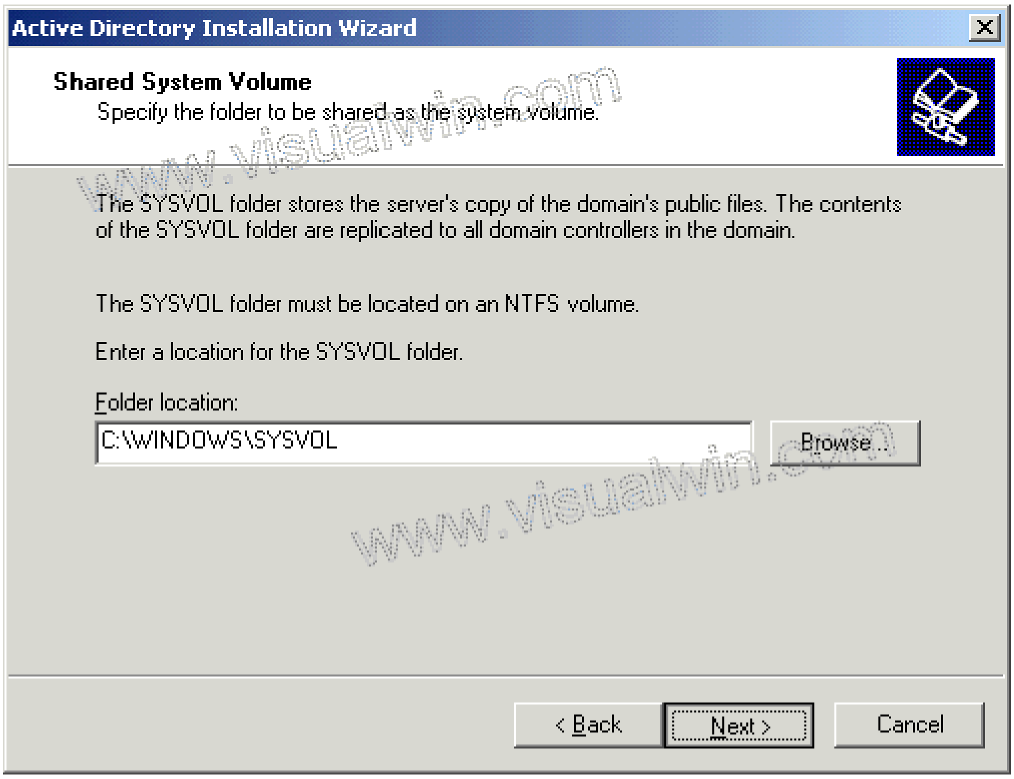
For best performance and recoverability, store the database and the log on separate hard disks.

Where do you want to store the Active Directory database?

Database folder: C:\WINDOWS\NTDS

Where do you want to store the Active Directory log?

Log folder: C:\WINDOWS\NTDS



Shared System Volume

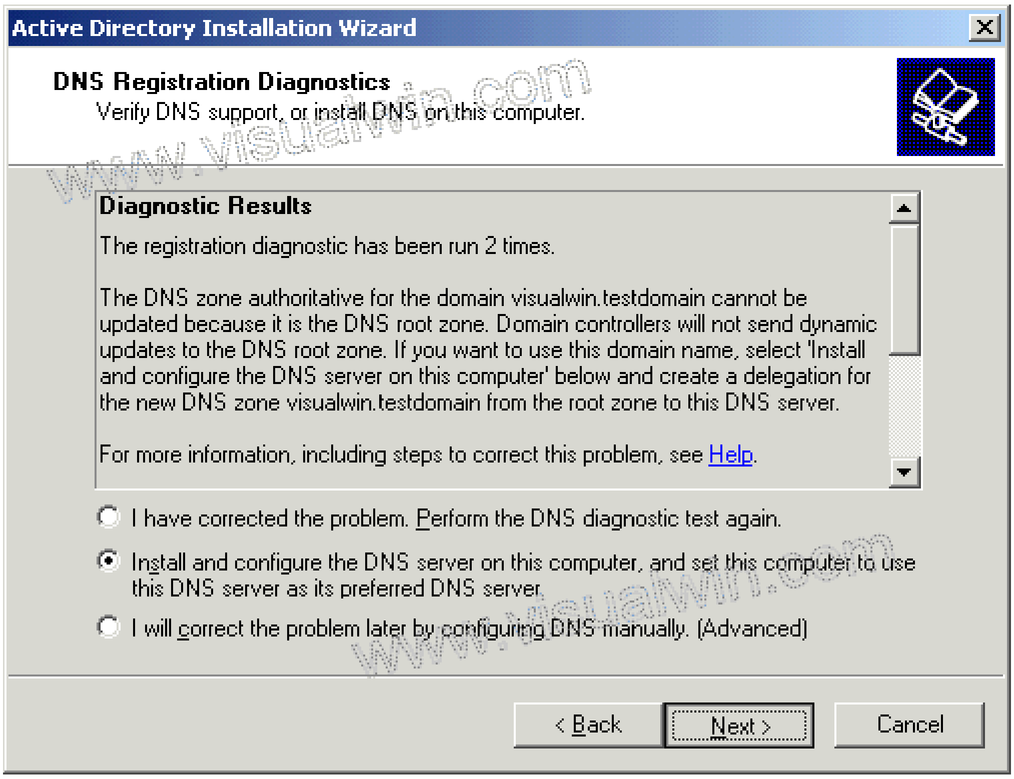
Specify the folder to be shared as the system volume.

The SYSVOL folder stores the server's copy of the domain's public files. The contents of the SYSVOL folder are replicated to all domain controllers in the domain.

The SYSVOL folder must be located on an NTFS volume.

Enter a location for the SYSVOL folder.

Folder location: C:\WINDOWS\SYSVOL



DNS Registration Diagnostics

Verify DNS support, or install DNS on this computer.

Diagnostic Results

The registration diagnostic has been run 2 times.

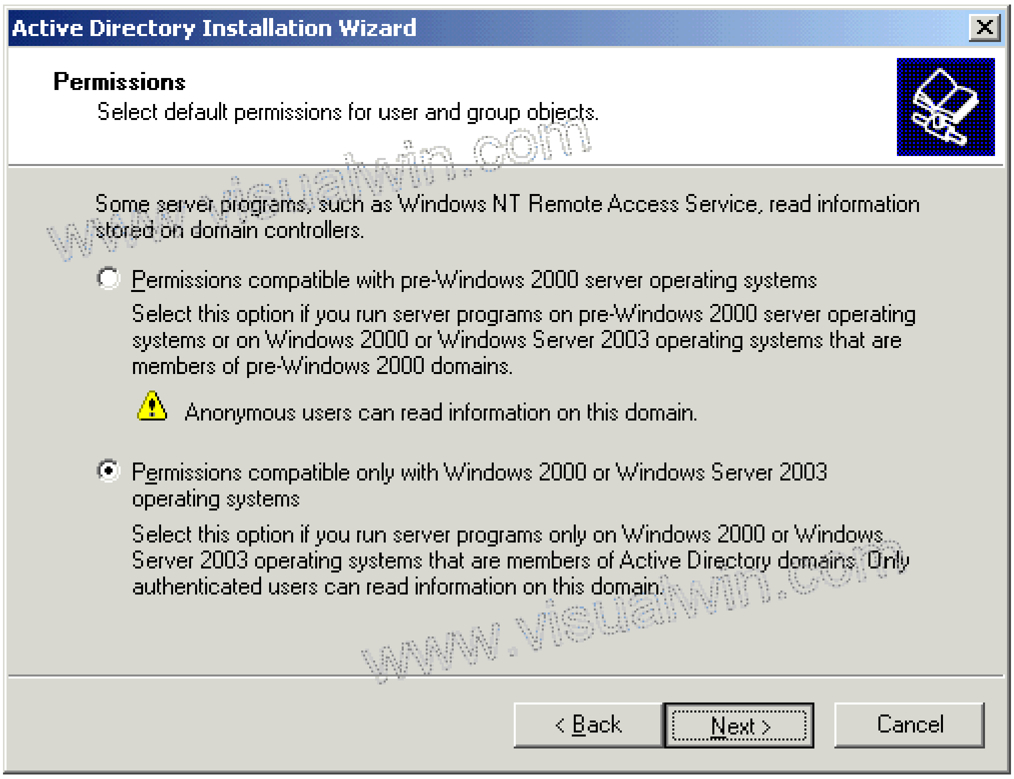
The DNS zone authoritative for the domain visualwin.testdomain cannot be updated because it is the DNS root zone. Domain controllers will not send dynamic updates to the DNS root zone. If you want to use this domain name, select 'Install and configure the DNS server on this computer' below and create a delegation for the new DNS zone visualwin.testdomain from the root zone to this DNS server.

For more information, including steps to correct this problem, see Help.

○ I have corrected the problem. Perform the DNS diagnostic test again.

● Install and configure the DNS server on this computer, and set this computer to use this DNS server as its preferred DNS server.

○ I will correct the problem later by configuring DNS manually. (Advanced)



Permissions

Select default permissions for user and group objects.

Some server programs, such as Windows NT Remote Access Service, read information stored on domain controllers.

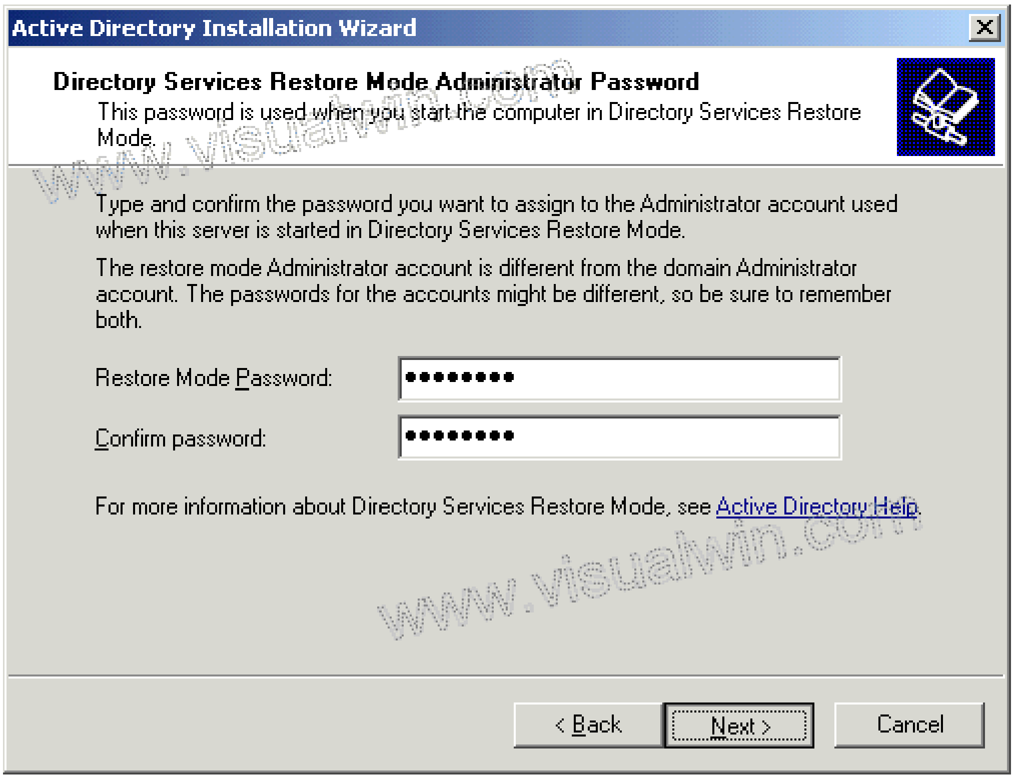
○ Permissions compatible with pre-Windows 2000 server operating systems

Select this option if you run server programs on pre-Windows 2000 server operating systems or on Windows 2000 or Windows Server 2003 operating systems that are members of pre-Windows 2000 domains.

⚠ Anonymous users can read information on this domain.

● Permissions compatible only with Windows 2000 or Windows Server 2003 operating systems

Select this option if you run server programs only on Windows 2000 or Windows Server 2003 operating systems that are members of Active Directory domains. Only authenticated users can read information on this domain.



Directory Services Restore Mode Administrator Password

This password is used when you start the computer in Directory Services Restore Mode.

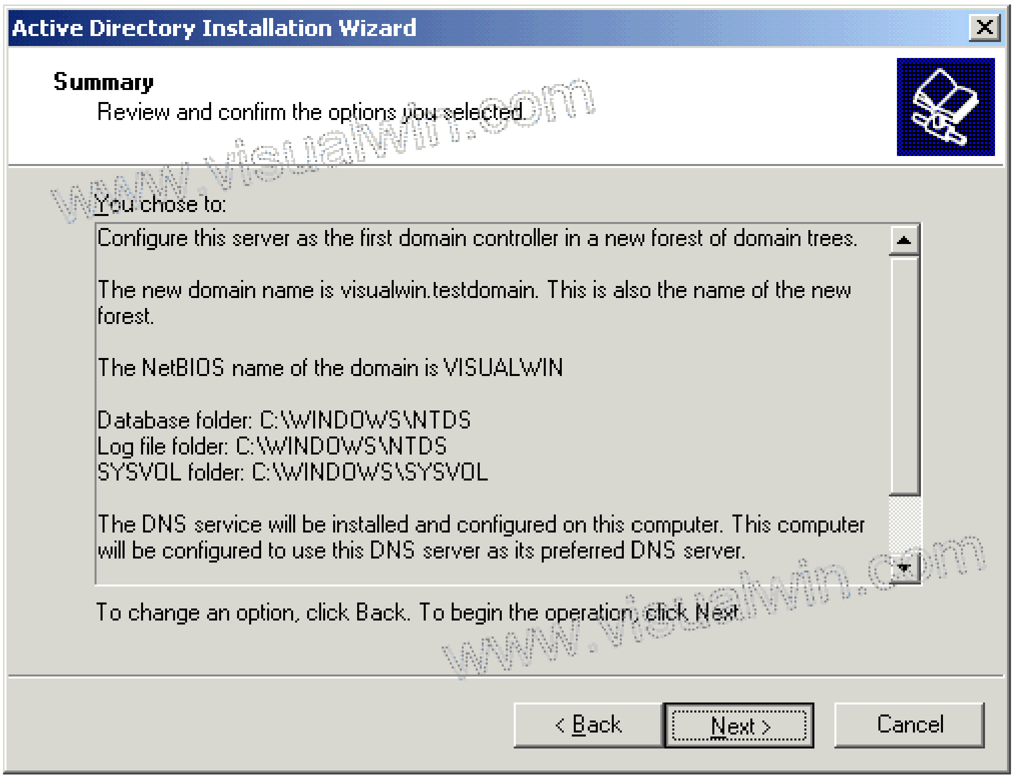
Type and confirm the password you want to assign to the Administrator account used when this server is started in Directory Services Restore Mode.

The restore mode Administrator account is different from the domain Administrator account. The passwords for the accounts might be different, so be sure to remember both.

Restore Mode Password: ●●●●●●●●●

Confirm password: ●●●●●●●●●

For more information about Directory Services Restore Mode, see Active Directory Help.



Summary

Review and confirm the options you selected.

You chose to:

Configure this server as the first domain controller in a new forest of domain trees.

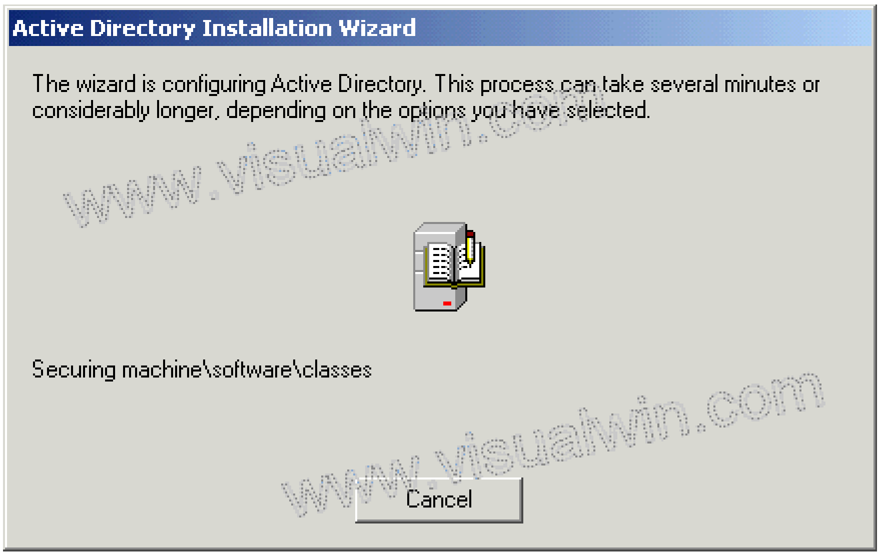
The new domain name is visualwin.testdomain. This is also the name of the new forest.

The NetBIOS name of the domain is VISUALWIN

Database folder: C:\WINDOWS\NTDS
Log file folder: C:\WINDOWS\NTDS
SYSVOL folder: C:\WINDOWS\SYSVOL

The DNS service will be installed and configured on this computer. This computer will be configured to use this DNS server as its preferred DNS server.

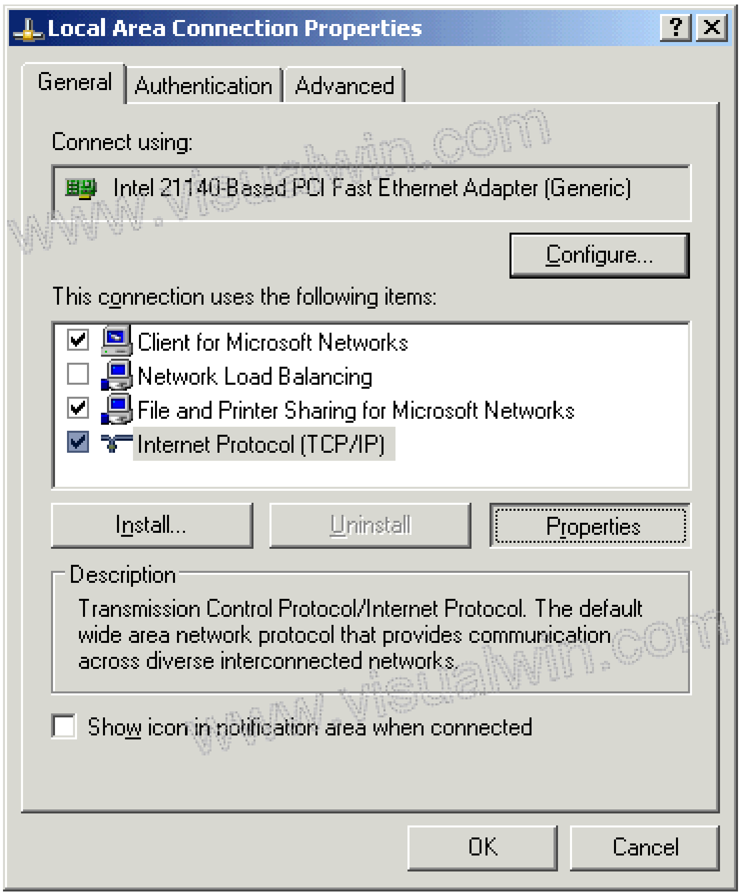
To change an option, click Back. To begin the operation, click Next.



The wizard is configuring Active Directory. This process can take several minutes or considerably longer, depending on the options you have selected.

Securing machine\software\classes

Click ok, and then in the Local Area Connection properties, click "Internet Protocol (TCP/IP)" and then "Properties"



Local Area Connection Properties

Connect using:

Intel 21140-Based PCI Fast Ethernet Adapter (Generic)

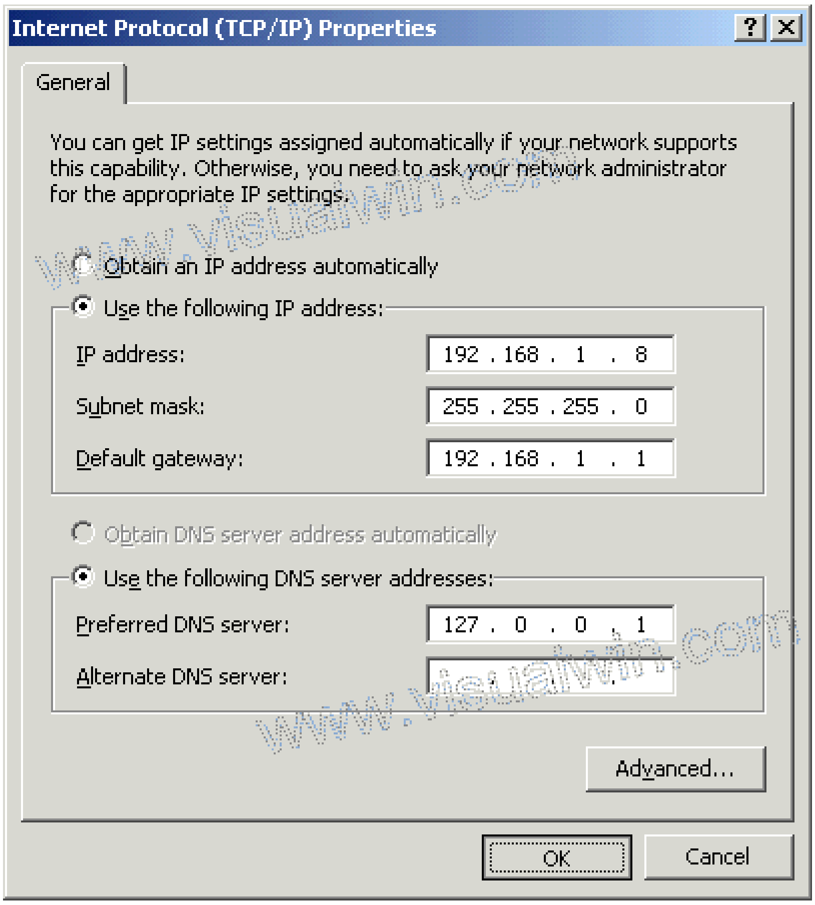
This connection uses the following items:

☑ Client for Microsoft Networks
□ Network Load Balancing
☑ File and Printer Sharing for Microsoft Networks
☑ Internet Protocol (TCP/IP)

Description

Transmission Control Protocol/Internet Protocol. The default wide area network protocol that provides communication across diverse interconnected networks.

□ Show icon in notification area when connected



You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically

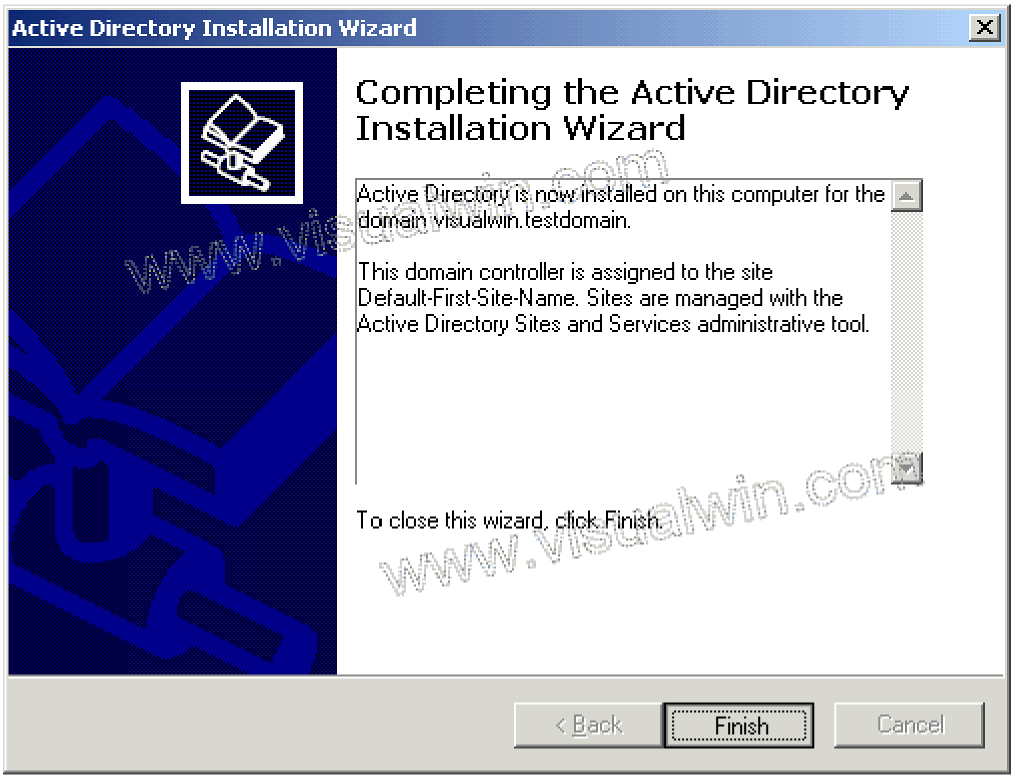
● Use the following IP address:

IP address:	192 . 168 . 1 . 8
Subnet mask:	255 . 255 . 255 . 0
Default gateway:	192 . 168 . 1 . 1

○ Obtain DNS server address automatically

● Use the following DNS server addresses:

Preferred DNS server:	127 . 0 . 0 . 1
Alternate DNS server:	. . .



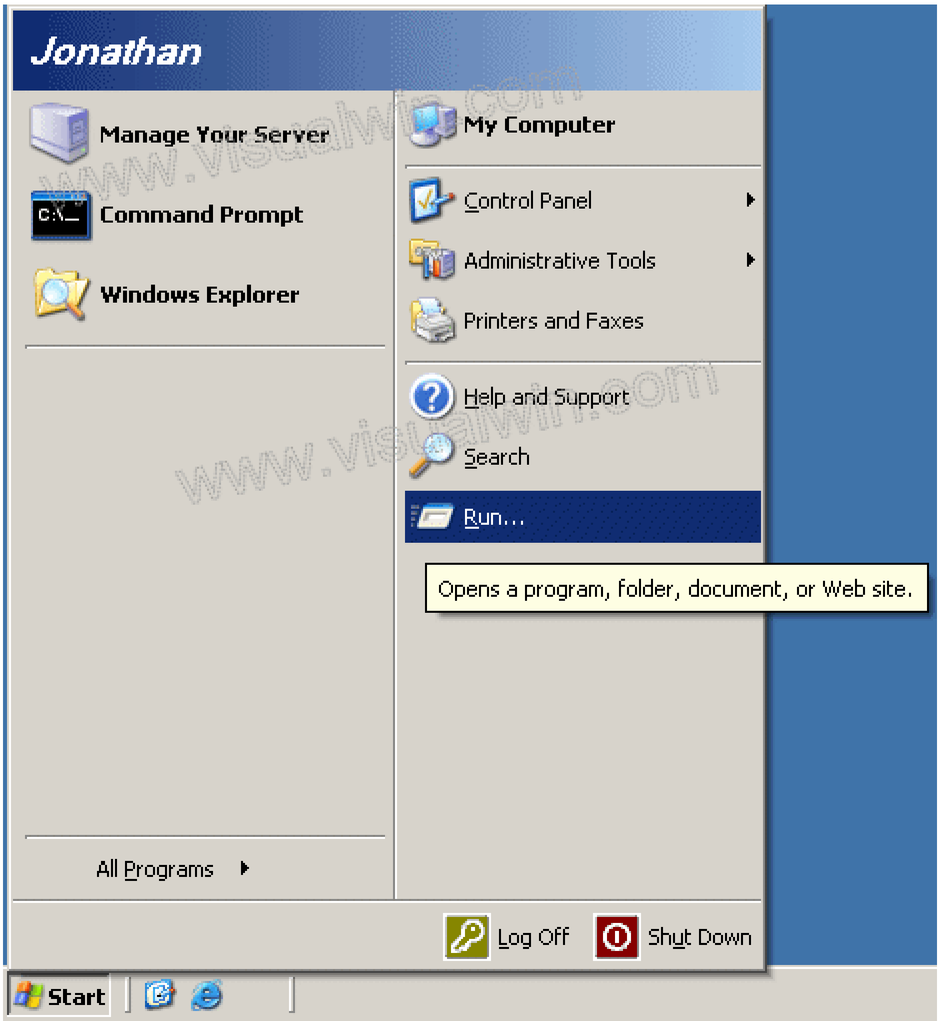
Completing the Active Directory Installation Wizard

Active Directory is now installed on this computer for the domain visualwin.testdomain.

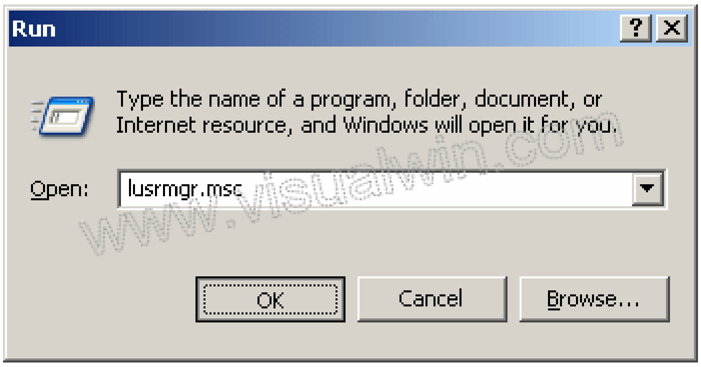
This domain controller is assigned to the site Default-First-Site-Name. Sites are managed with the Active Directory Sites and Services administrative tool.

To close this wizard, click Finish.

Creating a New User on Windows Server 2003

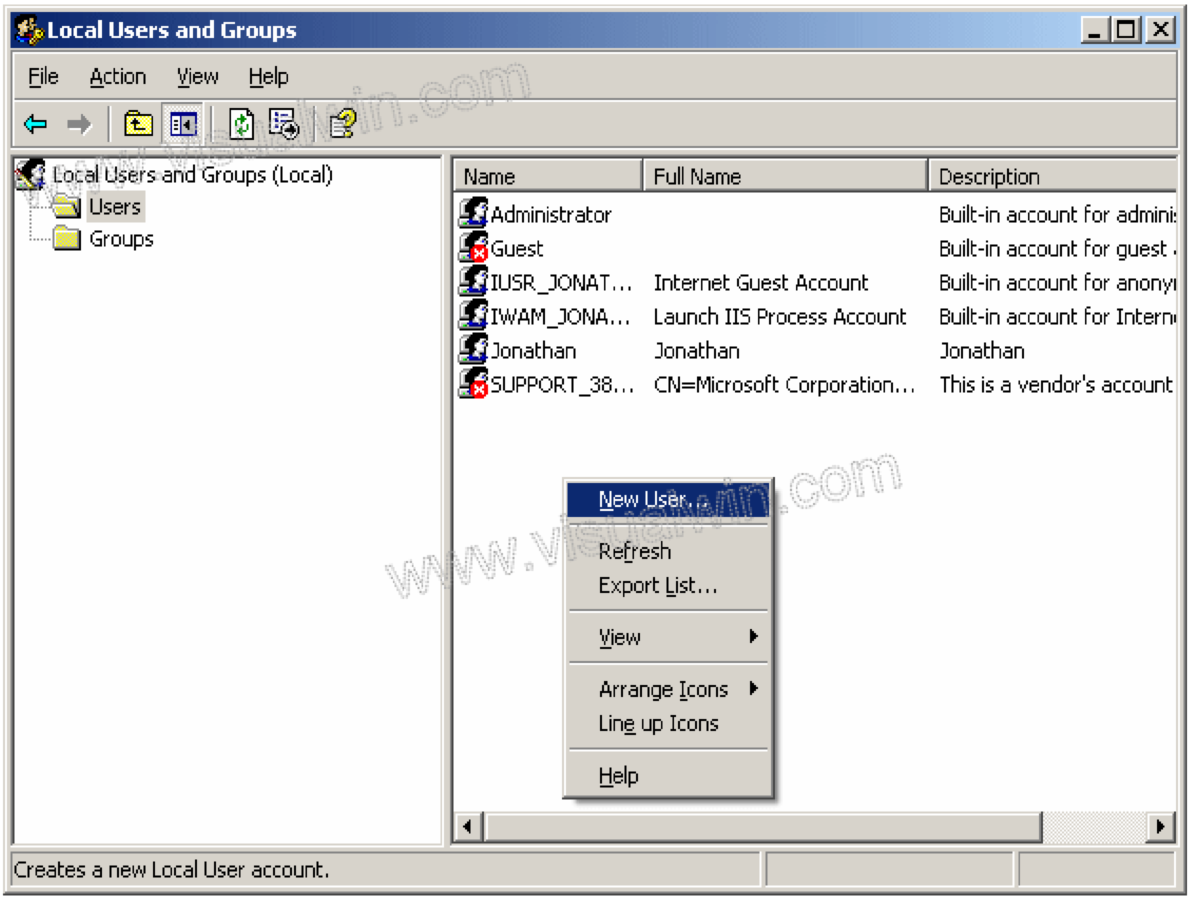


Opens a program, folder, document, or Web site.



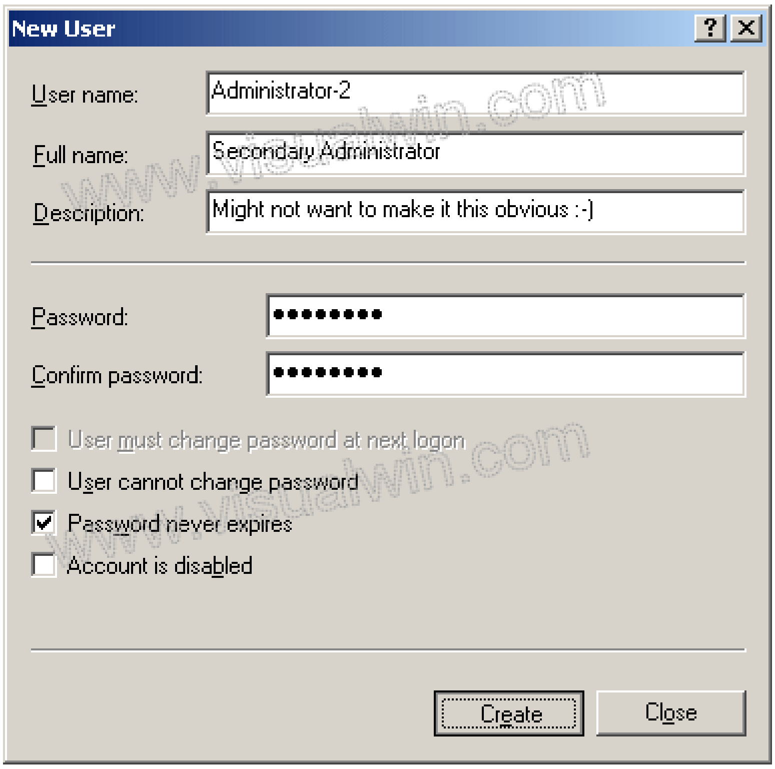
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: lusrmgr.msc



Name	Full Name	Description
Administrator		Built-in account for admini:
Guest		Built-in account for guest .
IUSR_JONAT...	Internet Guest Account	Built-in account for anonyı
IWAM_JONA...	Launch IIS Process Account	Built-in account for Intern
Jonathan	Jonathan	Jonathan
SUPPORT_38...	CN=Microsoft Corporation...	This is a vendor's account

Creates a new Local User account.



User name: Administrator-2

Full name: Secondary Administrator

Description: Might not want to make it this obvious :-)

Password: ●●●●●●●●●

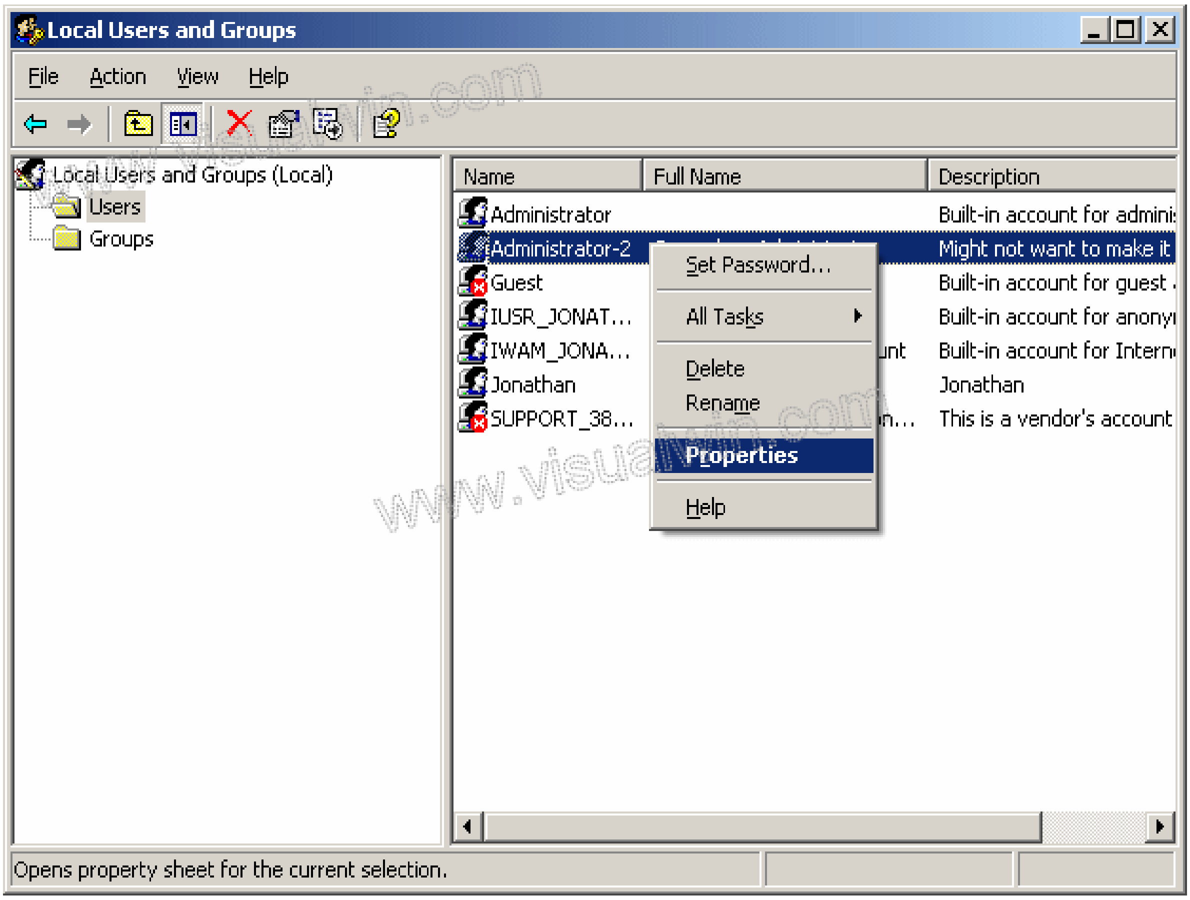
Confirm password: ●●●●●●●●●

□ User must change password at next logon

□ User cannot change password

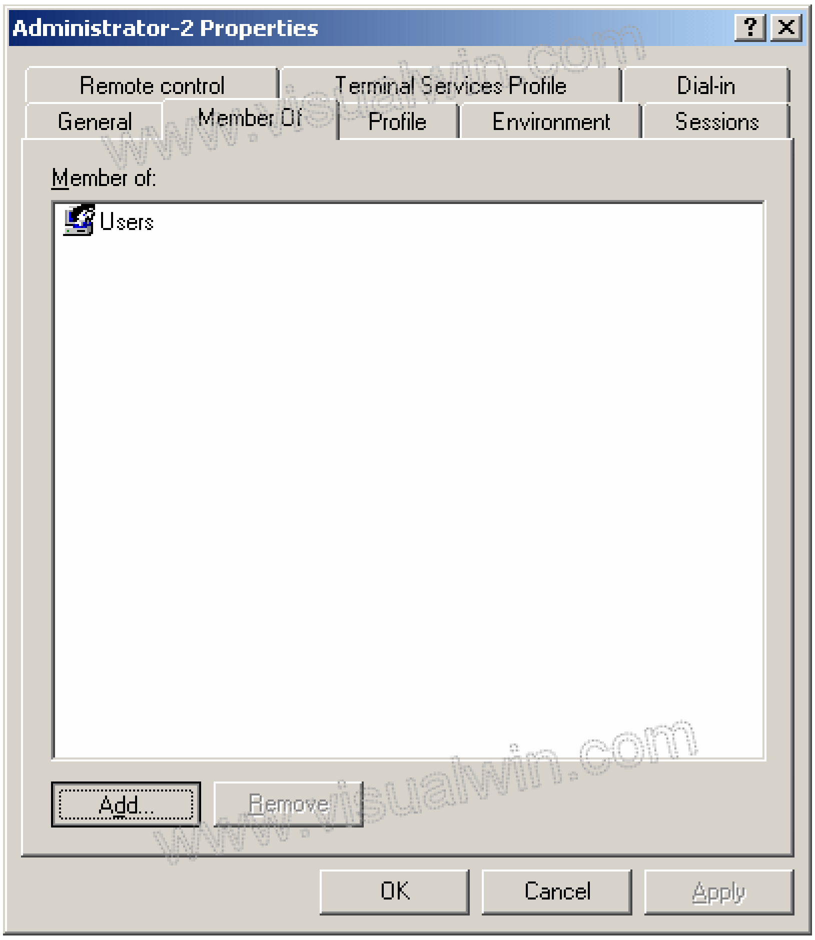
☑ Password never expires

□ Account is disabled

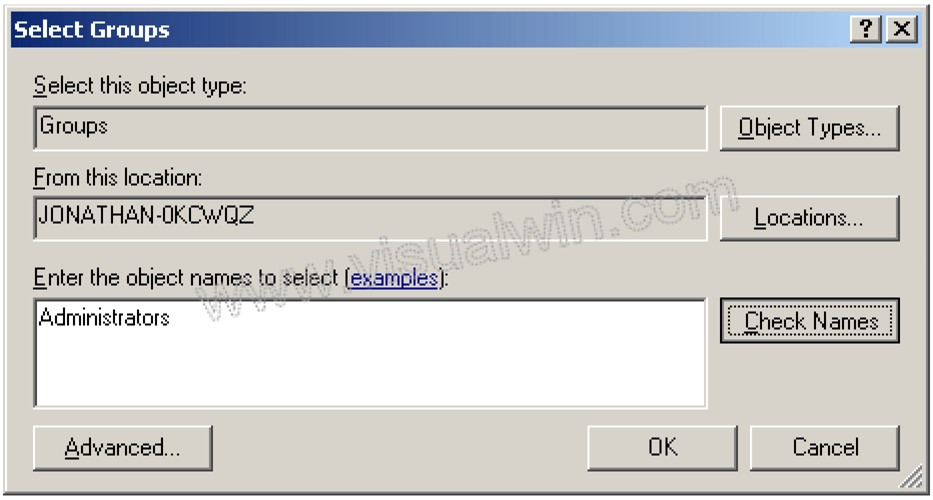


Name	Full Name	Description
Administrator		Built-in account for adminis
Administrator-2		Might not want to make it
Guest		Built-in account for guest
IUSR_JONAT...		Built-in account for anony
IWAM_JONA...	unt	Built-in account for Intern
Jonathan		Jonathan
SUPPORT_38...	n...	This is a vendor's account

Opens property sheet for the current selection.



Member of:
Users



Select this object type:
Groups

From this location:
JONATHAN-0KCWQZ

Enter the object names to select (examples):
Administrators

CODES

Server side:

```
import java.io.*;
import java.net.*;

publicclass Provider{
        ServerSocketproviderSocket;
        Socket connection = null;
        ObjectOutputStreamout;
        ObjectInputStreamin;
        String message;

        void run()
        {
                try{
                        providerSocket = newServerSocket(2004,10);
                        System.out.println("Waiting for connection");
                        connection = providerSocket.accept();
                        System.out.println("Connection received from " +
connection.getInetAddress().getHostName());
                        out = newObjectOutputStream(connection.getOutputStream());
                        out.flush();
                        in = newObjectInputStream(connection.getInputStream());
                        sendMessage("Connection successful");
                        do{
                                try{
                                        message = (String)in.readObject();
```

```
                    System.out.println("client>" + message);
                    if (message.equals("bye"))
                        sendMessage("bye");
                }
                catch(ClassNotFoundExceptionclassnot){
                    System.err.println("Data received in unknown
format");
                }
            }while(!message.equals("bye"));
        }
        catch(IOExceptionioException){
            ioException.printStackTrace();
        }
        finally{
            try{
                in.close();
                out.close();
                providerSocket.close();
            }
            catch(IOExceptionioException){
                ioException.printStackTrace();
            }
        }
    }
    voidsendMessage(String msg)
    {
        try{
            out.writeObject(msg);
            out.flush();
            System.out.println("server>" + msg);
        }
        catch(IOExceptionioException){
            ioException.printStackTrace();
        }
        }
        publicstaticvoid main(String args[])
        {
                Provider server = newProvider();
                while(true){
                        server.run();
                }
        }
}
```

Client side:

```
import java.io.*;
import java.net.*;

publicclass Requester{
    Socket requestSocket;
    ObjectOutputStreamout;
    ObjectInputStreamin;
    String message;

    void run()
    {
        try{
            requestSocket = new Socket("localhost", 2004);
            System.out.println("Connected to localhost in port 2004");

            out =
newObjectOutputStream(requestSocket.getOutputStream());
            out.flush();
            in = newObjectInputStream(requestSocket.getInputStream());
            do{
                try{
                    message = (String)in.readObject();
                    System.out.println("server>" + message);
                    sendMessage("Hi my server");
                    message = "bye";
                    sendMessage(message);
                }
                catch(ClassNotFoundExceptionclassNot){
                    System.err.println("data received in unknown
format");
```

```
                }
            }while(!message.equals("bye"));
        }
        catch(UnknownHostExceptionunknownHost){
            System.err.println("You are trying to connect to an unknown
host!");
        }
        catch(IOExceptionioException){
            ioException.printStackTrace();
        }
        finally{
            try{
                in.close();
                out.close();
                requestSocket.close();
            }
            catch(IOExceptionioException){
                ioException.printStackTrace();
            }
        }
    }
    voidsendMessage(String msg)
    {
        try{
            out.writeObject(msg);
            out.flush();
            System.out.println("client>" + msg);
        }
        catch(IOExceptionioException){
            ioException.printStackTrace();
        }
    }
    publicstaticvoid main(String args[])
    {
```

```
                Requester client = newRequester();
                client.run();
        }
}
```

User authentication:

```
importjava.awt.*;
importjavax.swing.*;
importjava.awt.event.*;

publicclass Info implementsActionListener
{
        publicintflag = 0;
        DataInsertiondi = newDataInsertion();
        publicJFramef;
        publicJLabelname,pass,intr;
        publicJTextFieldtname;
        publicJPasswordFieldtpass;
        publicJButtonbi,bu,ex,su;

        public Info()
        {
                f = newJFrame();
                f.setLayout(newGridLayout(1,2));
                ex = newJButton("EXISTING USER");
                su = newJButton("SIGN UP");
        }

        publicvoid page1()
        {
                f.add(ex);
                f.add(su);
                ex.addActionListener(this);
                su.addActionListener(this);
                f.setSize(400,200);
                f.setLocation(400,200);
```

```
			f.setVisible(true);
		}
		publicvoid page2()
		{
			f.setVisible(false);
			f = newJFrame();
			f.setLayout(newGridLayout(6,1));
//			f.setBackground(Color.WHITE);
//			f.setForeground(Color.BLUE);
			intr = newJLabel("SIGN IN");
			name = newJLabel("NAME");
			tname = newJTextField(30);
			pass = newJLabel("PASSWORD");
			tpass = newJPasswordField(20);
			bi = newJButton("SUBMIT");
			f.getRootPane().setDefaultButton(bi);
			f.add(intr);
			f.add(name);
			f.add(tname);
			f.add(pass);
			f.add(tpass);
			f.add(bi);
			bi.addActionListener(this);
			f.setSize(400,200);
			f.setLocation(400,200);
			f.setVisible(true);
		}

		publicvoidsignUp()
		{
			f.setVisible(false);
			f = newJFrame();
			f.setLayout(newGridLayout(6,1));
			intr = newJLabel("SIGN UP");
```

```
        name = newJLabel("NAME");
        tname = newJTextField(30);
        pass = newJLabel("PASSWORD");
        tpass = newJPasswordField(20);
        bu = newJButton("SUBMIT");
        f.add(intr);
        f.add(name);
        f.add(tname);
        f.add(pass);
        f.add(tpass);
        f.add(bu);
        f.getRootPane().setDefaultButton(bu);
        bu.addActionListener(this);
        f.setSize(400,200);
        f.setLocation(400,200);
        f.setVisible(true);
    }
    publicvoidactionPerformed(ActionEvent e)
    {
        if(e.getSource() == bi)
        {
            try
            {
                char[] ch = (char[]) tpass.getPassword();
                String str = newString(ch);
                //System.out.println("Password = " + str);
                f.setVisible(false);
                di.checkIn((String) tname.getText(),str);
            }
            catch(Exception ex)
            {}
        }
        elseif(e.getSource() == bu)
        {
```

```
                        try
                        {
                                char[] ch = (char[]) tpass.getPassword();
                                String str = newString(ch);
                                //System.out.println("Password = " + str);
                                f.setVisible(false);
                                di.insertIn((String) tname.getText(),str);
                        }

                        catch(Exception ex)
                        {}

                }
                elseif(e.getSource() == ex)
                {
                        flag = 1;
                        page2();
                }
                elseif(e.getSource() == su)
                {
                        flag = 2;
                        signUp();
                }
        }

        /*public static void main(String args[])
        {
                Info i = new Info();
                i.page1();

        }*/

}
```

Brute Force by limiting the number of attempts:

```
publicclass BruteForce1 {

        publicint attack1(intctr)
        {
                if(ctr> 5)
                        return 1;

                elseif(ctr == 5)
                        return 0;

                else
                        return 2;
        }
}
```

Brute Force Method by imposing a time limit:

```
publicclass BruteForce2 {

        publicint attack2(java.util.Datefrst)
        {
                java.util.Date today = newjava.util.Date();

                if((today.compareTo(frst) > 0)&&(today.compareTo(frst) < 300000))
                return 2;

                elseif(today.compareTo(frst) == 300000)
                        return 0;

                else
                        return 1;
        }

}
```

SQL connectivity:

```
import java.io.*;
importjava.sql.*;
importjava.awt.*;
importjavax.swing.*;

publicclassDataInsertion
{
        JFramef;
        JLabeljl;
        staticintctr = 0;
        BruteForce1 b1 = newBruteForce1();
        BruteForce2 b2 = newBruteForce2();
        java.util.Datefrst;

        publicvoidiniti()
        {
            f = newJFrame();
            f.setLayout(newBorderLayout());
            jl = newJLabel("CONNECTING...............");
            f.add(jl,BorderLayout.CENTER);
            f.setSize(400,200);
            f.setLocation(400,200);
            f.setVisible(true);
        }

        publicvoidinsertIn(String name, String pass1) throwsIOException
        {
            initi();

            String url = "jdbc:mysql://localhost:3306/db";
```

```
                    Connection conn;
                    ResultSetrs;

                    try
                    {
                            Class.forName("com.mysql.jdbc.Driver");
                            conn = DriverManager.getConnection(url,
"root", "root");
                            Statement statement = conn.createStatement();

                            String query = "insert into authvalues('" + name
+ "','" + pass1 + "','" + "'unblocked','123')";
                            int i = statement.executeUpdate(query);
                            if(i!=0)
                            {
                                    System.out.println("The record has been
inserted");
                            }
                            else
                            {
                                    System.out.println("Sorry!! Failure");
                            }

                            rs = statement.executeQuery("select * from
auth");
                            System.out.println(" NAME \t\t\t\t PASSWORD
\t\t\t\t STATUS");
                            while(rs.next())
                                    System.out.println(" " + rs.getString(1) +
" \t\t\t\t " + rs.getString(2) + " \t\t\t\t " + rs.getString(3));
                            rs.close();
                            statement.close();
                    }
                    catch (Exception e)
                    {
                                System.out.println(e);
                    }
          }

               publicvoidcheckIn(String name, String pass) throwsIOException
               {
                    Info in = newInfo();
                    String url = "jdbc:mysql://localhost:3306/db";
                    Connection conn;
                    ResultSetrs;
                    try
                    {
                                Class.forName("com.mysql.jdbc.Driver");
                                conn = DriverManager.getConnection(url,
"root", "root");
                                Statement statement = conn.createStatement();
                                int flag = 0;
                                rs = statement.executeQuery("select * from
auth");
                                //System.out.println(" NAME \t\t\t\t
PASSWORD \t\t\t\t STATUS");
                                initi();

                                while(rs.next())
                                {

          if((rs.getString(1).equals(name))&&(rs.getString(2).equals(pass)))
                                     {

          if((rs.getString(3).equals("unblocked")))
                                          {
                                               flag = 1;
                                jl.setText("LOGIN
SUCCESSFUL");
                            }
                            else
                            {
                                flag = 2;
                                jl.setText("BLOCKED
ACCOUNT                 CONTACT ADMINISTRATOR");
                            }
                        }
                    }
                    if(flag == 0)
                    {
                        jl.setText("INCORRECT
ID/PASSWORD    TRY AGAIN");
                        for(int i=1; i<=10000000; i++);
                            ctr++;
                        intct = b1.attack1(ctr);

                        if(ct == 0)
                        {
                            rs =
statement.executeQuery("select * from auth");
                            while(rs.next())
                            {

    if(rs.getString(1).equals(name))
                                {

    statement.executeUpdate("update auth set status='blocked' where username='"
+ name + "'");

    jl.setText("ACCOUNT BLOCKED");
                                }
```

```
                                                        }
                                                }

                                                in.page2();
                                                if(ctr == 1)
                                                {
                                                        frst = newjava.util.Date();
                                                        //System.out.println(frst.toString(
));
                                                }
                                                        int ta = b2.attack2(frst);
                                                        if(ta == 0 &&ctr>= 5)
                                                        {
                                                                rs =
statement.executeQuery("select * from auth");

        if(rs.getString(1).equals(name))
                                                                {

        statement.executeUpdate("update auth set status='blocked' where username='"
+ name + "'");

        jl.setText("ACCOUNT BLOCKED");
                                                                }
                                                        }
                                                }
                                        rs.close();
                                        statement.close();
                        }
                        catch (Exception e)
                        {
                                        System.out.println(e);
                        }       }}
```

Triple- DES:

```
importjava.security.*;
importjavax.crypto.*;

publicclass Encrypt {
privatestatic String algorithm = "DESede";
privatestatic Key key = null;
privatestatic Cipher cipher = null;
privatestaticvoidsetUp() throws Exception
        {
                        key = KeyGenerator.getInstance(algorithm).generateKey();
                        cipher = Cipher.getInstance(algorithm);
        }
publicstaticvoid main(String[] args) throws Exception
        {
setUp();
byte[] encryptionBytes = null;
            String input = "453";
System.out.println("Entered: " + input);
encryptionBytes = encrypt(input);
System.out.println(encryptionBytes);
System.out.println("Recovered: " + decrypt(encryptionBytes));
        }
privatestaticbyte[] encrypt(String input)throws
InvalidKeyException,BadPaddingException,IllegalBlockSizeException
        {
cipher.init(Cipher.ENCRYPT_MODE, key);
byte[] inputBytes = input.getBytes();
returncipher.doFinal(inputBytes);
        }
privatestatic String decrypt(byte[] encryptionBytes)throws
InvalidKeyException,BadPaddingException,IllegalBlockSizeException
        {
cipher.init(Cipher.DECRYPT_MODE, key);
byte[] recoveredBytes = cipher.doFinal(encryptionBytes);
            String recovered = newString(recoveredBytes);
return recovered;
         }
   }
```

RSA:

```
importjava.security.*;
importjavax.crypto.*;

publicclass Encrypt {
privatestatic String algorithm = "DESede";
privatestatic Key key = null;
privatestatic Cipher cipher = null;
privatestaticvoidsetUp() throws Exception
        {
                        key = KeyGenerator.getInstance(algorithm).generateKey();
                        cipher = Cipher.getInstance(algorithm);
        }
publicstaticvoid main(String[] args) throws Exception
        {
setUp();
byte[] encryptionBytes = null;
            String input = "453";
System.out.println("Entered: " + input);
encryptionBytes = encrypt(input);
System.out.println(encryptionBytes);
System.out.println("Recovered: " + decrypt(encryptionBytes));
        }
privatestaticbyte[] encrypt(String input)throws
InvalidKeyException,BadPaddingException,IllegalBlockSizeException
        {
cipher.init(Cipher.ENCRYPT_MODE, key);
byte[] inputBytes = input.getBytes();
returncipher.doFinal(inputBytes);
        }
privatestatic String decrypt(byte[] encryptionBytes)throws
InvalidKeyException,BadPaddingException,IllegalBlockSizeException
        {
cipher.init(Cipher.DECRYPT_MODE, key);
byte[] recoveredBytes = cipher.doFinal(encryptionBytes);
            String recovered = newString(recoveredBytes);
return recovered;
         }
   }
```

Echo Client – Server:

```
importjava.awt.*;
importjava.awt.event.*;
import java.io.*;
import java.net.*;
importjava.util.Scanner;
importjavax.swing.*;
importjavax.swing.text.DefaultCaret;

publicclassSerCliimplementsActionListener, Runnable {

privatestaticfinal String HOST = "127.0.0.1";
privatestaticfinalintPORT = 2004;
privatefinalJFramef = newJFrame();
privatefinalJTextFieldtf = newJTextField(25);
privatefinalJTextAreata = newJTextArea(15, 25);
privatefinalJButtonsend = newJButton("Send");
volatilePrintWriterout;
private Scanner in;
private Thread thread;
private Kind kind;
public String str;


publicstaticenum Kind {

Client(100, "Trying"), Server(500, "Awaiting");
privateintoffset;
private String activity;

private Kind(int offset, String activity) {
```

```
this.offset = offset;
this.activity = activity;
        }
    }

publicSerCli()
    {


    }
publicSerCli(Kind kind) {
this.kind = kind;
f.setTitle("Echo " + kind);
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
f.getRootPane().setDefaultButton(send);
f.add(tf, BorderLayout.NORTH);
f.add(newJScrollPane(ta), BorderLayout.CENTER);
f.add(send, BorderLayout.SOUTH);
f.setLocation(kind.offset, 300);
f.pack();
send.addActionListener(this);
ta.setLineWrap(true);
ta.setWrapStyleWord(true);
DefaultCaret caret = (DefaultCaret) ta.getCaret();
caret.setUpdatePolicy(DefaultCaret.ALWAYS_UPDATE);
display(kind.activity + HOST + " on port " + PORT);
thread = new Thread(this, kind.toString());
    }

publicvoid start() {
f.setVisible(true);
thread.start();
    }

//@Override
```

```
publicvoidactionPerformed(ActionEventae) {
        String s = tf.getText();
if (out != null) {
out.println(s);
        }
display(s);
tf.setText("");
    }

//@Override
publicvoid run() {
try {
            Socket socket;
if (kind == Kind.Client)
            {
socket = new Socket(HOST, PORT);
            }
else
            {
ServerSocketss = newServerSocket(PORT);
socket = ss.accept();
            }
in = new Scanner(socket.getInputStream());
out = newPrintWriter(socket.getOutputStream(), true);
display("Enter Existing User/Sign Up:");
str = in.nextLine();
while (true) {
display(str);


            }
        } catch (Exception e) {
display(e.getMessage());
e.printStackTrace(System.err);
        }
```

```
   }

void display(final String s) {
EventQueue.invokeLater(new Runnable() {
publicvoid run() {
                ta.append(s + "\u23CE\n");
        }
    });

   }



publicstaticvoid main(String[] args) {
EventQueue.invokeLater(new Runnable() {
publicvoid run() {
SerCliser = newSerCli(Kind.Server);
ser.start();
SerCli cli = newSerCli(Kind.Client);
cli.start();
            Info inf = newInfo();
/*if(str ==  "existing user")
            {
        str = "";
        inf.page2();
            }
else if(str ==  "existing user")
            {
        str = "";
        inf.signUp();
            }*/
/*if(inf.flag == 1)
        out.write("Existing User");
else if(inf.flag == 2)
```

```
            ser.display("Signing Up");*/
            }
        });
    }
}
```

Key Distribution Centre

KDC:

```
importjava.security.*;
importjavax.crypto.*;

publicclass KDC {

        publicstatic Key key = null;
        publicstatic Cipher cipher = null;

        public KDC()throws Exception
        {

                        String algorithm = "DESede";
        key = KeyGenerator.getInstance(algorithm).generateKey();
        cipher = Cipher.getInstance(algorithm);
        }

        public Key getKey()
   {
                /*String algorithm = "DESede";
     //private static int[] key = new int[24];
key = KeyGenerator.getInstance(algorithm).generateKey();
cipher = Cipher.getInstance(algorithm);*/
        returnkey;
   }
```

```
        public Cipher getCipher()
    {
        returncipher;
    }

        staticPublicKeypubli = null;
        staticPrivateKeyprivatei = null;

        publicvoidkeyRsa() throws Exception
        {
                KeyPairGeneratorkpg = KeyPairGenerator.getInstance("RSA");
                kpg.initialize(512);
                KeyPairkp = kpg.genKeyPair();
                publi = kp.getPublic();
                privatei = kp.getPrivate();
                //return kp;
        }

        publicPublicKeygetPublicKey()
        {
                returnpubli;
        }

        publicPrivateKeygetPrivateKey()
        {
                returnprivatei;
        }
}
```

Holder:

```
importjava.security.*;

        publicclass Holder
        {
                publicstaticvoid main(String args[])
                {
                        try
                        {
                                        KDC k = newKDC();
                                        k.keyRsa();
                                        PublicKeypubli = k.getPublicKey();
                                        PrivateKeyprivatei = k.getPrivateKey();
                                        RSAEncrypt en = newRSAEncrypt();
                                        byte[] cipherData = en.encryption(publi);
                                        RSADecrypt de = newRSADecrypt();
                                        de.decryption(privatei, cipherData);
                        }
                        catch(Exception e)
                        {
                                System.out.println(e.getMessage());
                        }               }               }
```

DES working with KDC:

```
importjava.security.*;
importjavax.crypto.*;
importjava.util.Scanner;

publicclass DES {
//private static String algorithm = "DESede";
/*private static Key key = null;
	//private static int[] key = new int[24];
private static Cipher cipher = null;*/

publicstaticvoid main(String[] args) throws Exception
	{
	    Scanner s = newScanner(System.in);
			KDC kdc = newKDC();
			DESDecrypt d = newDESDecrypt();
			Key key = kdc.getKey();
			Cipher cipher = kdc.getCipher();
byte[] encryptionBytes = null;
System.out.println("Enter the Data:");
	    String input = s.next();
System.out.println("Entered Data: " + input);
encryptionBytes = encrypt(input, key, cipher);
System.out.println(encryptionBytes);
System.out.println("Recovered Data: " + d.decrypt(encryptionBytes, key, cipher));
	}
staticbyte[] encrypt(String input, Key key, Cipher cipher) throws
InvalidKeyException,BadPaddingException,IllegalBlockSizeException
	{
cipher.init(Cipher.ENCRYPT_MODE, key);
byte[] inputBytes = input.getBytes();
returncipher.doFinal(inputBytes);
        }
    }
```

DES Decryption:

```
importjava.security.InvalidKeyException;
importjava.security.Key;

importjavax.crypto.BadPaddingException;
importjavax.crypto.Cipher;
importjavax.crypto.IllegalBlockSizeException;

publicclassDESDecrypt {


        String decrypt(byte[] encryptionBytes, Key key, Cipher cipher)throws
InvalidKeyException,BadPaddingException,IllegalBlockSizeException
      {
cipher.init(Cipher.DECRYPT_MODE, key);
byte[] recoveredBytes = cipher.doFinal(encryptionBytes);
        String recovered = newString(recoveredBytes);
return recovered;
      }
}
```

RSA Encryption:

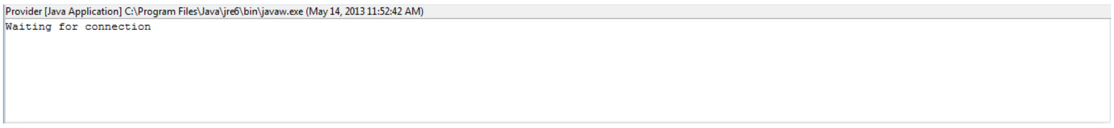
```
importjava.security.*;
import java.io.*;
importjavax.crypto.Cipher;

    publicclassRSAEncrypt
    {
        publicbyte[] encryption(PublicKeypubli)
        {
            String srci="";
            byte[] cipherData = null;
            try
            {
                    BufferedReaderbr = newBufferedReader(newInputStreamReader(System.in));
                    System.out.println("Please enter any string you want to encrypt");
                    srci = br.readLine();
            }
            catch(IOExceptionioe)
            {
                System.out.println(ioe.getMessage());
            }

            try
            {
                    System.out.println("Public Key = " + publi);
                    Cipher cipher = Cipher.getInstance("RSA");
                    cipher.init(Cipher.ENCRYPT_MODE, publi);
                    byte[]src = srci.getBytes();
                    cipherData = cipher.doFinal(src);
                    String srco = newString(cipherData);
                    System.out.println("Encrypted data is: " + srco);
            }
            catch(Exception e)
            {
                System.out.println(e.getMessage());
            }
            returncipherData;
        }
}
```

RSA Decryption:

```
importjava.security.*;
importjavax.crypto.Cipher;

    publicclassRSADecrypt
    {
        publicvoid decryption(PrivateKeyprivatei, byte[] cipherData)
        {
            try
            {
                    Cipher cipheri = Cipher.getInstance("RSA");
                    cipheri.init(Cipher.DECRYPT_MODE, privatei);
                    byte[] cipherDat = cipheri.doFinal(cipherData);
                    String decryptdata = newString(cipherDat);
                    System.out.println("Decrypted data: "+
decryptdata);
            }
            catch(Exception e)
            {
                System.out.println(e.getMessage());
            }
        }
}
```

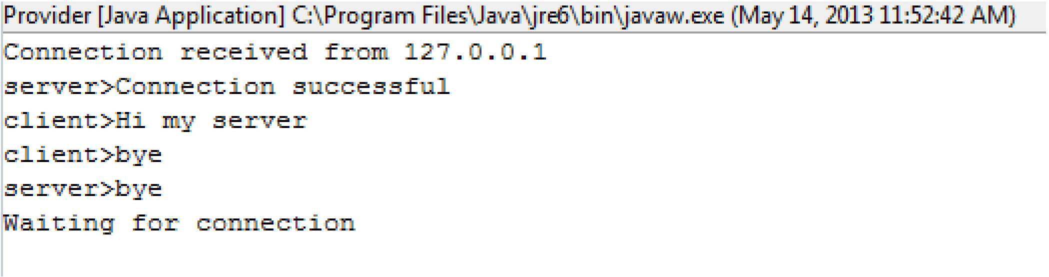
OUTPUTS

Basic interaction between client and server:



Provider [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 14, 2013 11:52:42 AM)

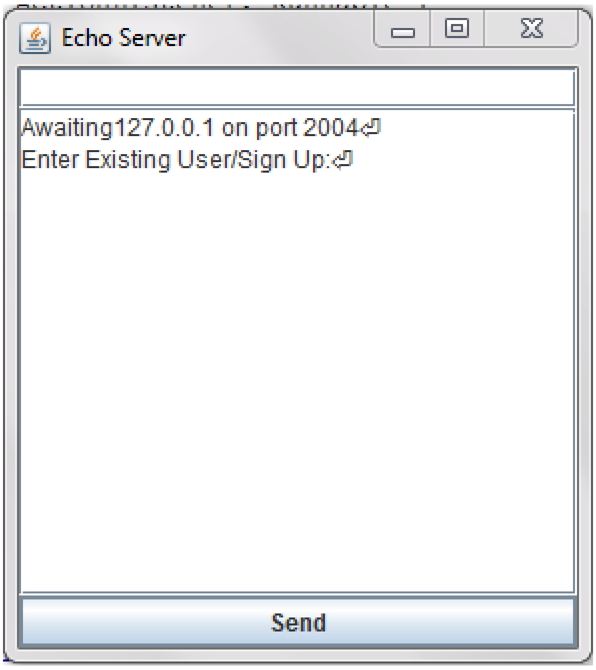
```
Waiting for connection
```



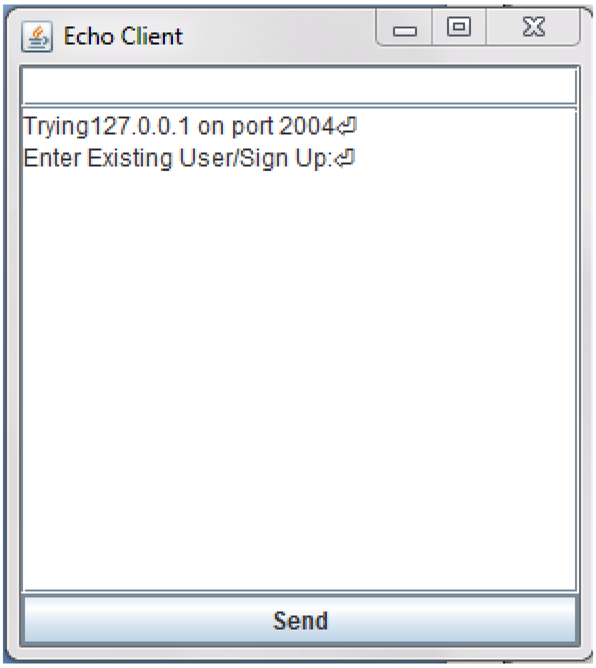
Provider [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 14, 2013 11:52:42 AM)

```
Connection received from 127.0.0.1
server>Connection successful
client>Hi my server
client>bye
server>bye
Waiting for connection
```

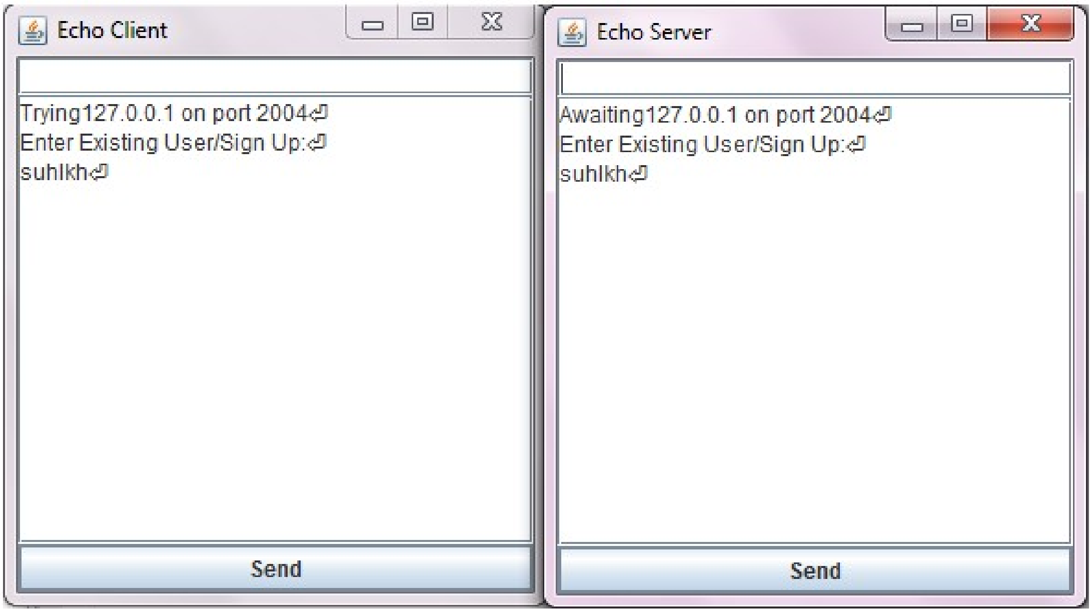
Echo Client Server:



```
Awaiting127.0.0.1 on port 2004⏎
Enter Existing User/Sign Up:⏎
```



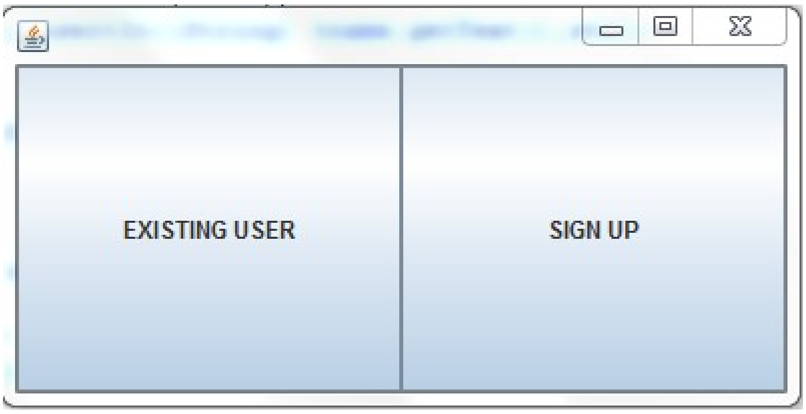
```
Trying127.0.0.1 on port 2004⏎
Enter Existing User/Sign Up:⏎
```

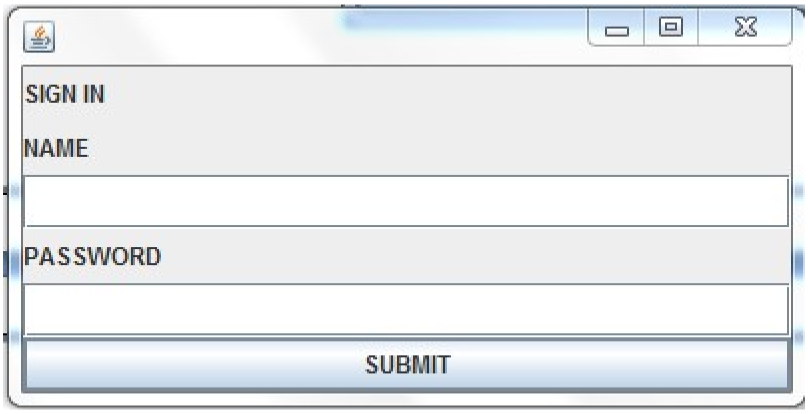


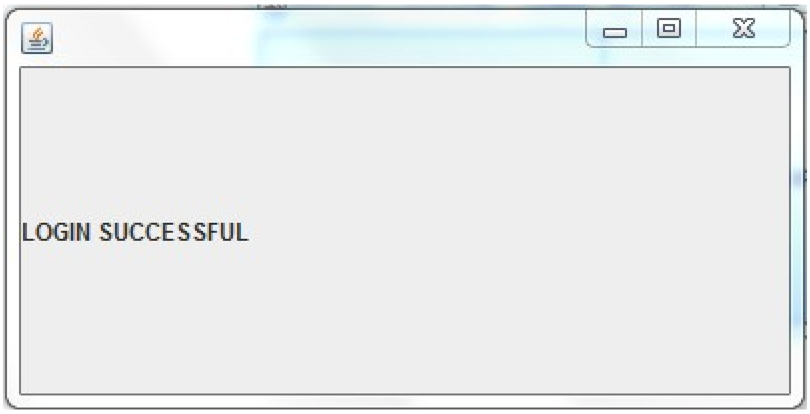
```
Trying127.0.0.1 on port 2004⏎
Enter Existing User/Sign Up:⏎
suhlkh⏎
```

```
Awaiting127.0.0.1 on port 2004⏎
Enter Existing User/Sign Up:⏎
suhlkh⏎
```

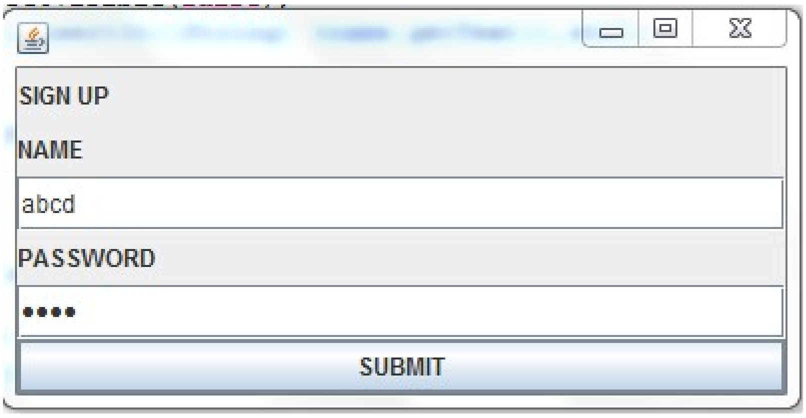
User interface for creating new user or signing up for existing user:







LOGIN SUCCESSFUL



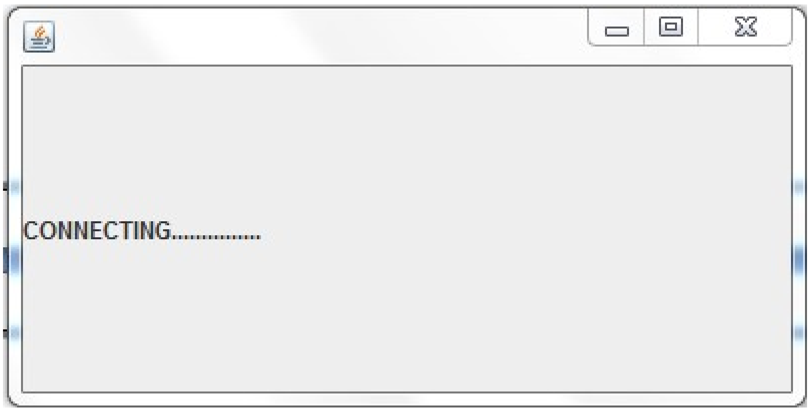
SIGN UP

NAME

abcd

PASSWORD

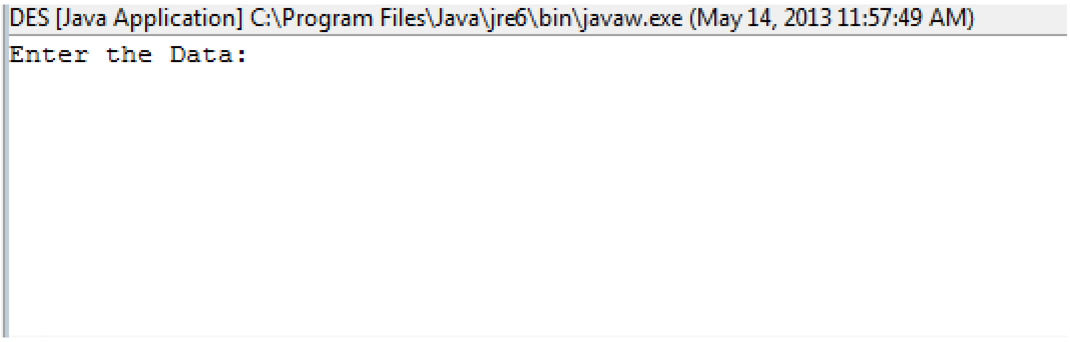
••••



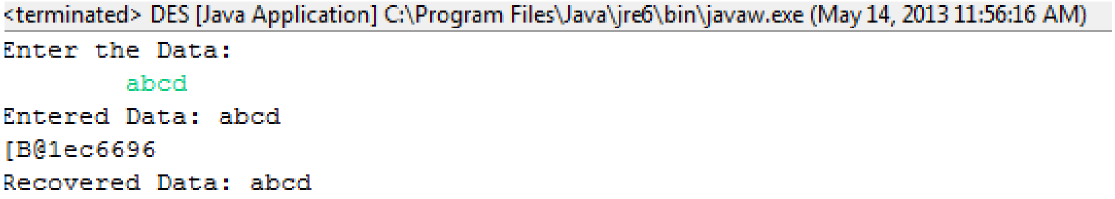
CONNECTING..............

Encryption / Decryption:

3DES:

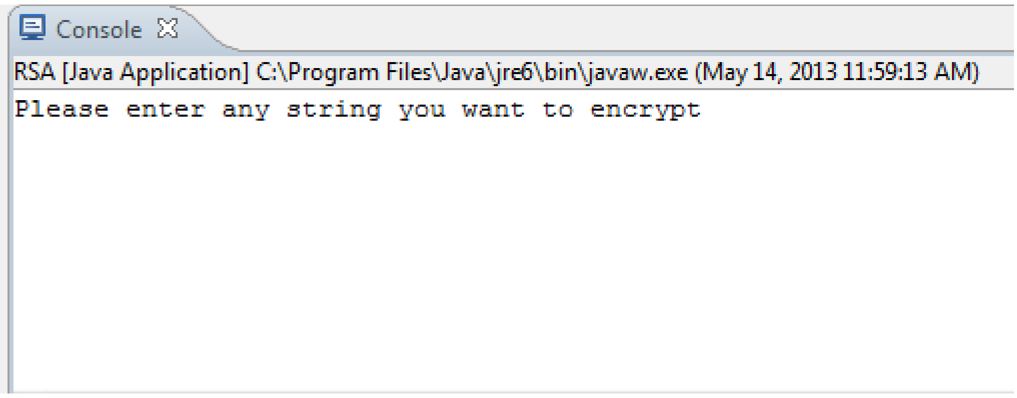


```
DES [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 14, 2013 11:57:49 AM)
Enter the Data:
```

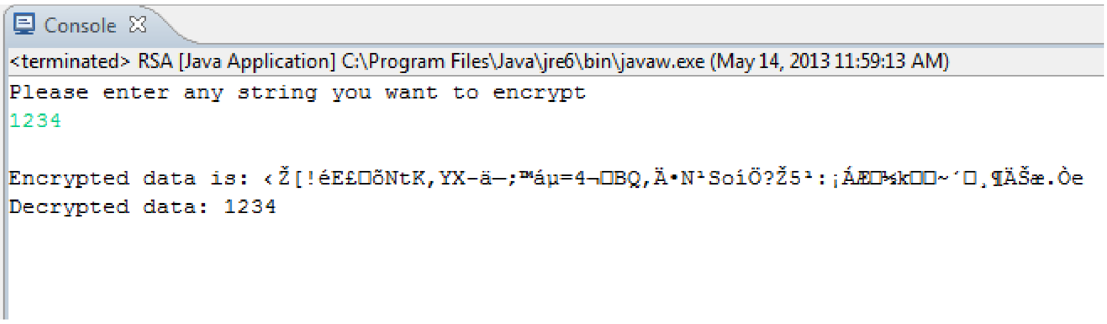


```
<terminated> DES [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 14, 2013 11:56:16 AM)
Enter the Data:
        abcd
Entered Data: abcd
[B@1ec6696
Recovered Data: abcd
```

RSA:

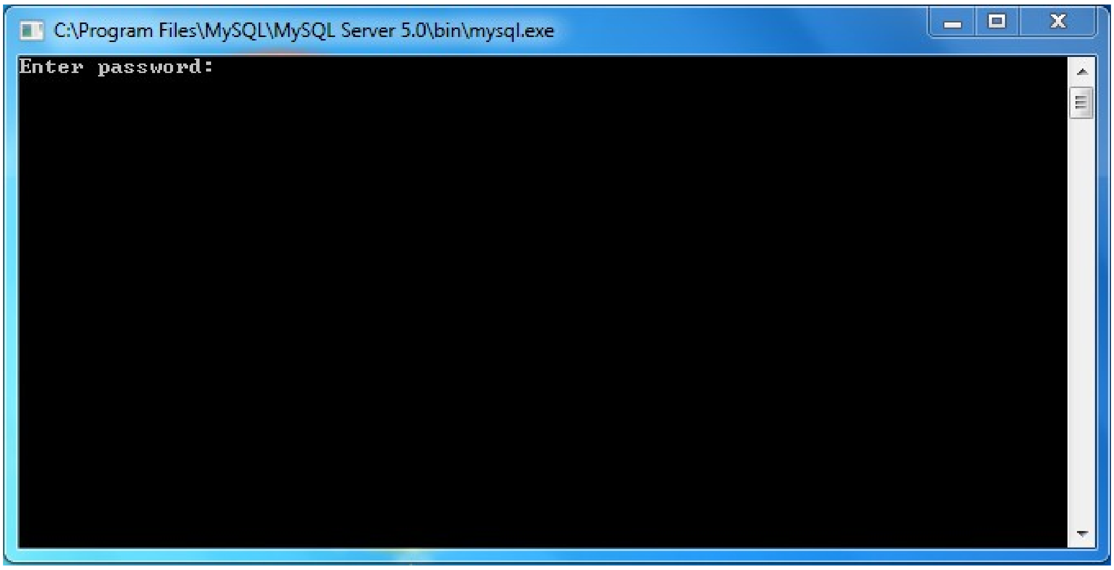


```
RSA [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 14, 2013 11:59:13 AM)
Please enter any string you want to encrypt
```

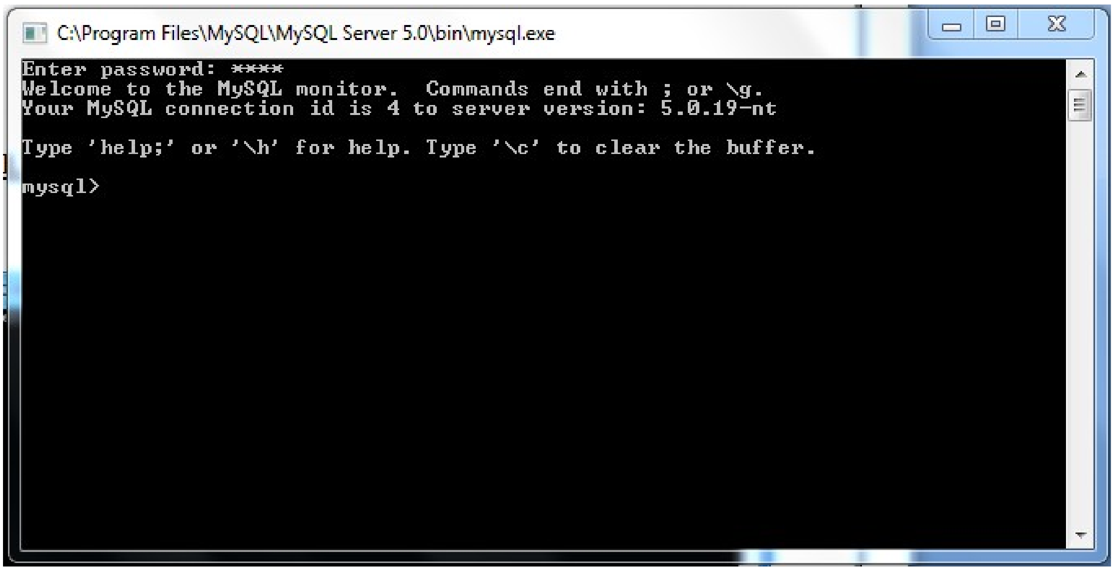


```
<terminated> RSA [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 14, 2013 11:59:13 AM)
Please enter any string you want to encrypt
1234

Encrypted data is: ‹Ž[!éE£□õNtK,YX-ä—;™áµ=4¬□BQ,Ä•N¹SoíÖ?Ž5¹:¡ÁÆ□¼k□□~´□¸¶ÄŠæ.Òe
Decrypted data: 1234
```

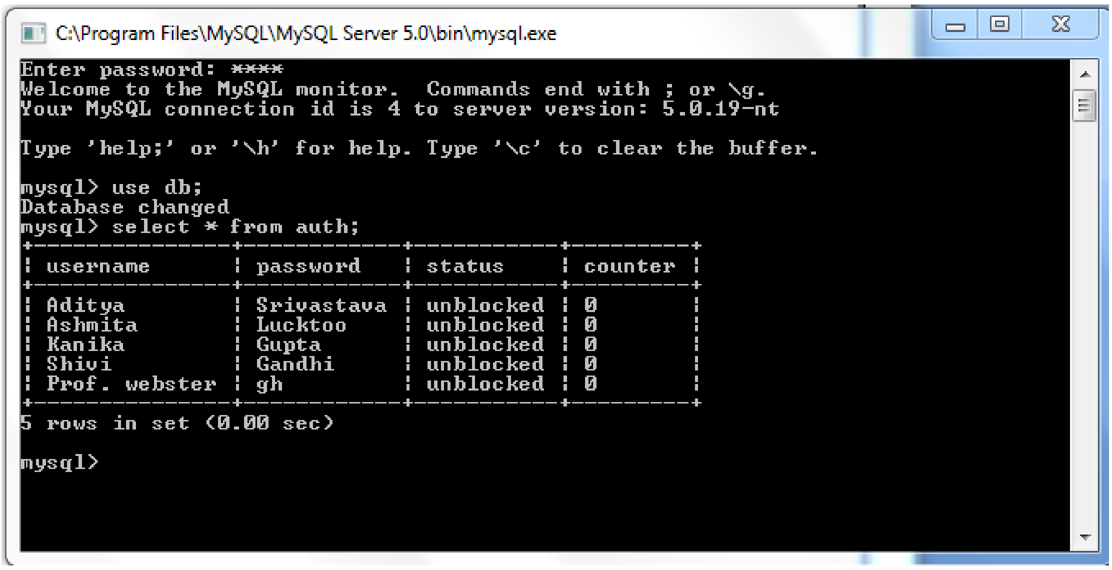
SQL Connectivity:



```
Enter password:
```

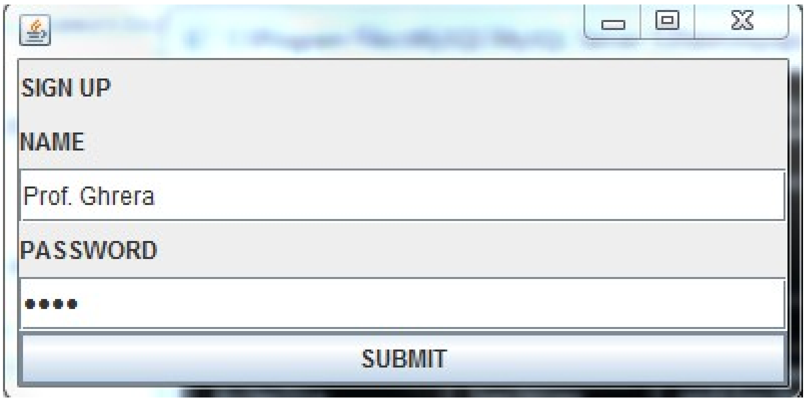


```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4 to server version: 5.0.19-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```



C:\Program Files\MySQL\MySQL Server 5.0\bin\mysql.exe

```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4 to server version: 5.0.19-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use db;
Database changed
mysql> select * from auth;
+---------------+------------+-----------+---------+
| username      | password   | status    | counter |
+---------------+------------+-----------+---------+
| Aditya        | Srivastava | unblocked | 0       |
| Ashmita       | Lucktoo    | unblocked | 0       |
| Kanika        | Gupta      | unblocked | 0       |
| Shivi         | Gandhi     | unblocked | 0       |
| Prof. webster | gh         | unblocked | 0       |
+---------------+------------+-----------+---------+
5 rows in set (0.00 sec)

mysql>
```



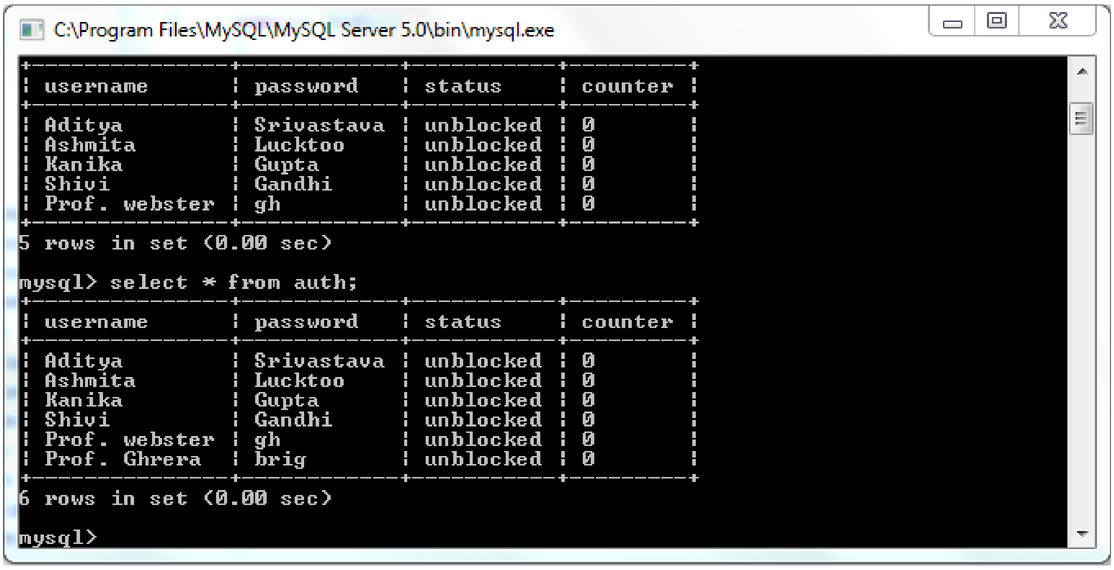
SIGN UP

NAME

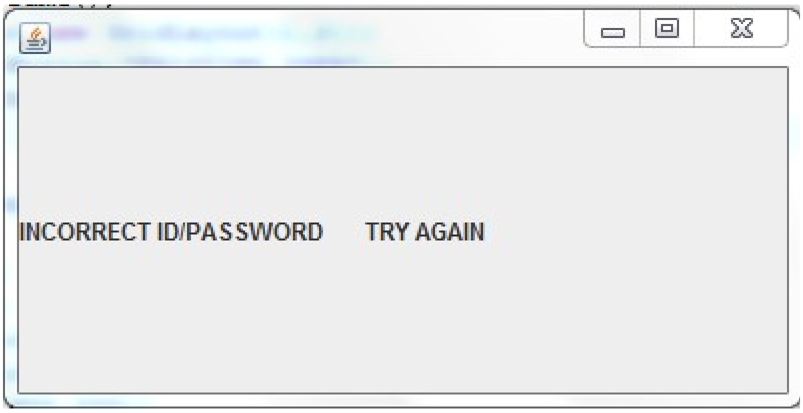
Prof. Ghrera

PASSWORD

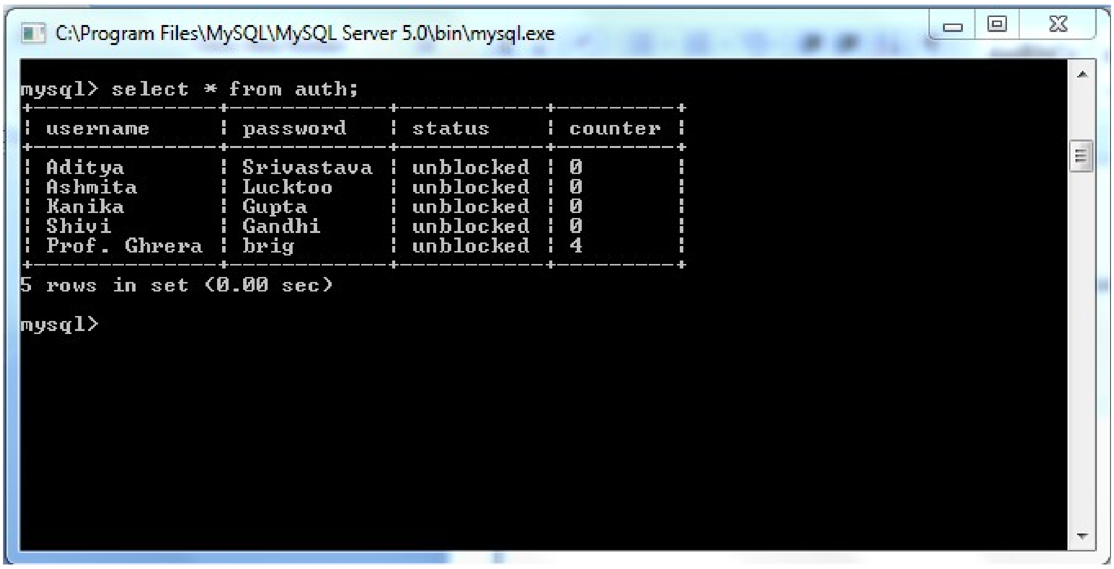
••••



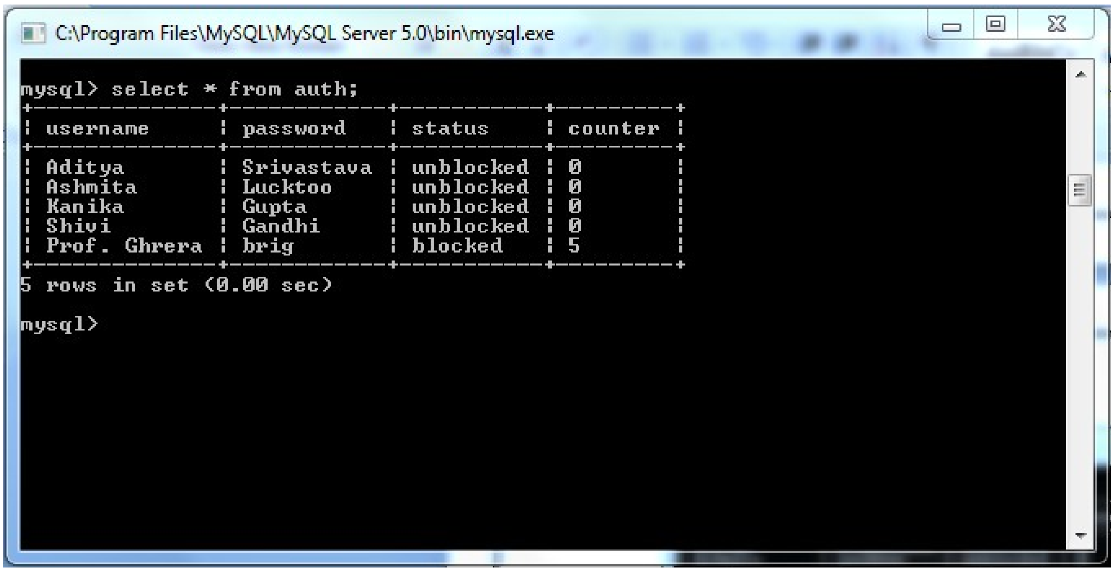
```
+----------------+------------+-----------+---------+
| username       | password   | status    | counter |
+----------------+------------+-----------+---------+
| Aditya         | Srivastava | unblocked | 0       |
| Ashmita        | Lucktoo    | unblocked | 0       |
| Kanika         | Gupta      | unblocked | 0       |
| Shivi          | Gandhi     | unblocked | 0       |
| Prof. webster  | gh         | unblocked | 0       |
+----------------+------------+-----------+---------+
5 rows in set (0.00 sec)

mysql> select * from auth;
+----------------+------------+-----------+---------+
| username       | password   | status    | counter |
+----------------+------------+-----------+---------+
| Aditya         | Srivastava | unblocked | 0       |
| Ashmita        | Lucktoo    | unblocked | 0       |
| Kanika         | Gupta      | unblocked | 0       |
| Shivi          | Gandhi     | unblocked | 0       |
| Prof. webster  | gh         | unblocked | 0       |
| Prof. Ghrera   | brig       | unblocked | 0       |
+----------------+------------+-----------+---------+
6 rows in set (0.00 sec)

mysql>
```



INCORRECT ID/PASSWORD   TRY AGAIN

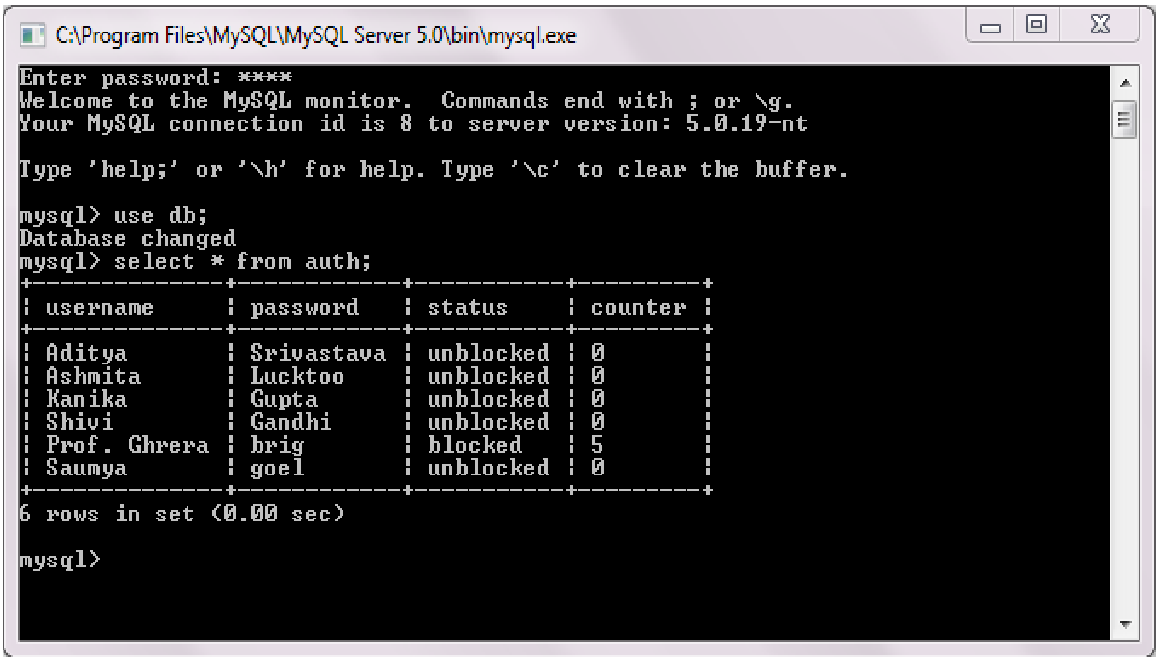


```
mysql> select * from auth;
+---------------+------------+-----------+---------+
| username      | password   | status    | counter |
+---------------+------------+-----------+---------+
| Aditya        | Srivastava | unblocked | 0       |
| Ashmita       | Lucktoo    | unblocked | 0       |
| Kanika        | Gupta      | unblocked | 0       |
| Shivi         | Gandhi     | unblocked | 0       |
| Prof. Ghrera  | brig       | unblocked | 4       |
+---------------+------------+-----------+---------+
5 rows in set (0.00 sec)

mysql>
```

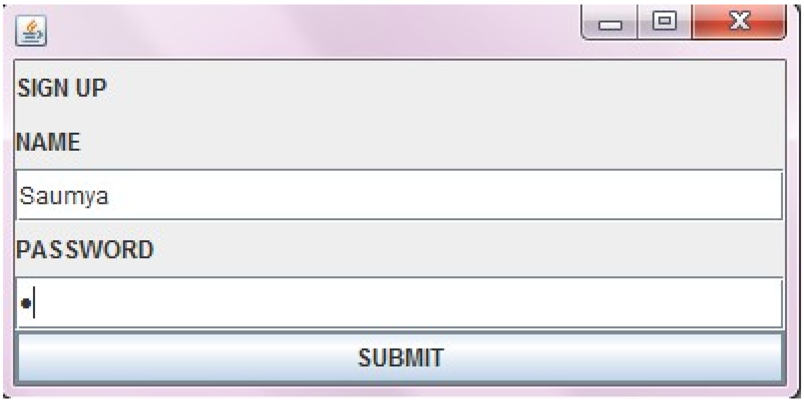


```
mysql> select * from auth;
+---------------+------------+-----------+---------+
| username      | password   | status    | counter |
+---------------+------------+-----------+---------+
| Aditya        | Srivastava | unblocked | 0       |
| Ashmita       | Lucktoo    | unblocked | 0       |
| Kanika        | Gupta      | unblocked | 0       |
| Shivi         | Gandhi     | unblocked | 0       |
| Prof. Ghrera  | brig       | blocked   | 5       |
+---------------+------------+-----------+---------+
5 rows in set (0.00 sec)

mysql>
```

Repetition of Username Not Allowed:



```
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8 to server version: 5.0.19-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use db;
Database changed
mysql> select * from auth;
+--------------+------------+-----------+---------+
| username     | password   | status    | counter |
+--------------+------------+-----------+---------+
| Aditya       | Srivastava | unblocked | 0       |
| Ashmita      | Lucktoo    | unblocked | 0       |
| Kanika       | Gupta      | unblocked | 0       |
| Shivi        | Gandhi     | unblocked | 0       |
| Prof. Ghrera | brig       | blocked   | 5       |
| Saumya       | goel       | unblocked | 0       |
+--------------+------------+-----------+---------+
6 rows in set (0.00 sec)

mysql>
```



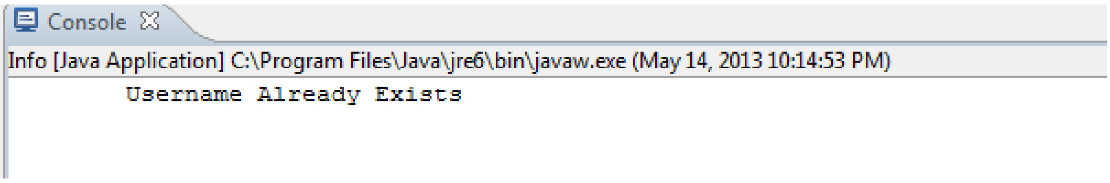
SIGN UP

NAME

Saumya

PASSWORD

•



```
Info [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (May 14, 2013 10:14:53 PM)
        Username Already Exists
```

REFERENCES

- Online Detection and Prevention of Phishing Attacks (Invited Paper) by Juan Chen, Institute of Communications Engineering Nanjing 210007, P.R. China

- DETECTION AND PREVENTION OF PHISHING ATTACKS *Under the guidance of* C.M.T. KARTHIGEYAN, B.E, M.S

- Best Practices for Institutions and Consumers by Gregg Tally, Roshan Thomas and Tom Van Vleck
- http://www.windowsreference.com/dns/step-by-step-guide-for-windows-server-2003-domain-controller-and-dns-server-setup/
- http://support.microsoft.com/kb/814591
- http://www.visualwin.com/New-User/
- http://www.microsoft.com/middleeast/windows/windowsserver2003/evaluation/whyupgrade/top10best.mspx
- http://support.microsoft.com/kb/278007
- http://www.wikipedia.org/
- http://nmap.org/book/idlescan.html
- http://its.sdsu.edu/blackboard/student/resources/phishing.html
- http://searchsecurity.techtarget.com/definition/brute-force-cracking
- http://en.wikipedia.org/wiki/Brute-force_attack
- http://en.wikipedia.org/wiki/RSA
- http://en.wikipedia.org/wiki/Triple_DES

- http://en.wikipedia.org/wiki/Data_Encryption_Standard
- http://en.wikipedia.org/wiki/Public-key_cryptography
- http://docs.oracle.com/javase/6/docs/api/java/security/PublicKey.html
- https://www.cpug.org/forums/ipsec-vpn-blade-virtual-private-networks/49-difference-between-3des-aes.html
- http://security.stackexchange.com/questions/26179/security-comparsion-of-3des-and-aes
- http://www.asecuritysite.com/security/encryption/threedes

THANK

YOU