# SQL Injection Attacks (SQLi)

- One of the most prevalent and dangerous network-based security threats
- Designed to exploit the nature of Web application pages
- Sends malicious SQL commands to the database server
- Most common attack goal is bulk extraction of data
- Depending on the environment SQL injection can also be exploited to:
  - Modify or delete data
  - Execute arbitrary operating system commands
  - Launch denial-of-service (DoS) attacks

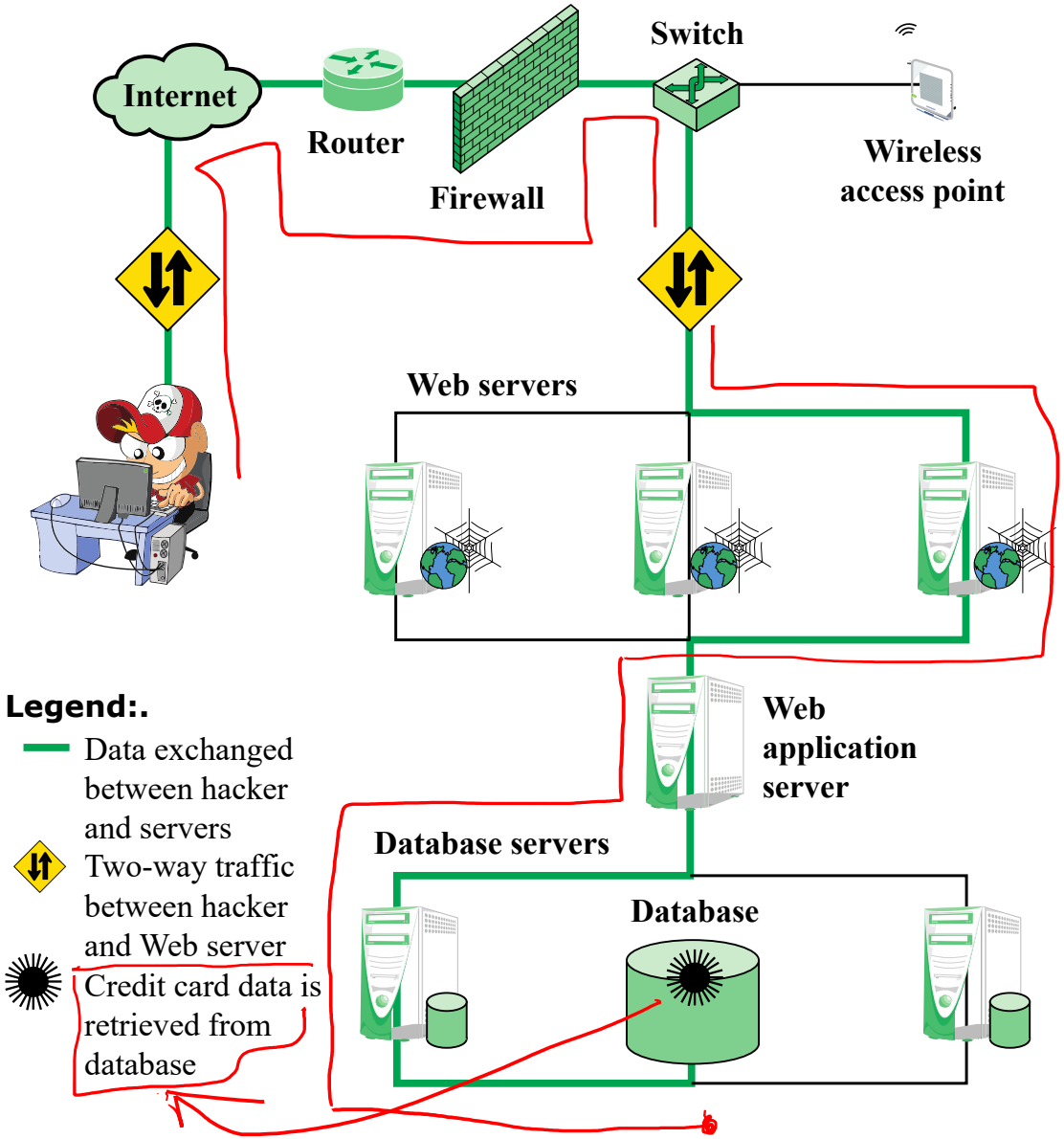
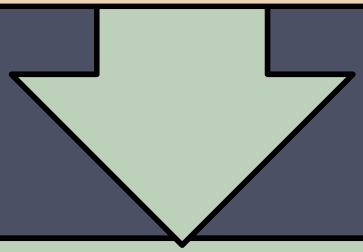



**Figure 5.5 Typical SQL Injection Attack**

# Injection Technique

**The SQLi attack typically works by prematurely terminating a text string and appending a new command**

Because the inserted command may have additional strings appended to it before it is executed the attacker terminates the injected string with a comment mark “- -”



**Subsequent text is ignored at execution time**

# SQL Injection.

User-Id : srinivas

Password : mypassword

```
select * from Users where user_id= ' srinivas '
                and password = ' mypassword '
```

User-Id : ' OR 1= 1; /*

Password : */--

A

B

```
select * from Users where user_id= '' OR 1 = 1; /* '
                and password = ' */-- '
```

C

# SQL statement processing

SELECT * FROM users WHERE email = '$email' AND password = md5('$password') ;

why this?

*Supplied values* { xxx@xxx.xxx  xxx') OR 1 = 1 -- ]

SELECT * FROM users WHERE email = 'xxx@xxx.xxx' AND password = md5('xxx') OR 1 = 1 -- ]');

SELECT * FROM users WHERE FALSE AND FALSE OR TRUE

SELECT * FROM users WHERE FALSE OR TRUE

SELECT * FROM users WHERE TRUE

# Construction of SQL statement as string

str='select * from emp where empid='|| :v_empid ||' and deptno='|| :v_dept ||
';'

(i) v_empid = 4850 and v_dept = 50

(ii) v_empid = '0R 1=1; --' and v_dept any value

(iii) v_empid = ' OR 1=1; update _____ ; --'

insert _____

delete _____

; drop database Pubs; --

# SQLi Attack Avenues

## 1 User input

- Attackers inject SQL commands by providing suitable crafted user input

## 2 Server variables

- Attackers can forge the values that are placed in HTTP and network headers and exploit this vulnerability by placing data directly into the headers

## 3 Second-order injection

- A malicious user could rely on data already present in the system or database to trigger an SQL injection attack, so when the attack occurs, the input that modifies the query to cause an attack does not come from the user, but from within the system itself

## 4 Cookies

- An attacker could alter cookies such that when the application server builds an SQL query based on the cookie's content, the structure and function of the query is modified

## 5 Physical user input

- Applying user input that constructs an attack outside the realm of web requests

Type of ATTACKS

# A Inband Attacks

- Uses the same communication channel for injecting SQL code and retrieving results
- The retrieved data are presented directly in application Web page
- Include:

## Tautology

This form of attack injects code in one or more conditional statements so that they always evaluate to true

## End-of-line comment

After injecting code into a particular field, legitimate code that follows are nullified through usage of end of line comments

## Piggybacked queries

The attacker adds additional queries beyond the intended query, piggy-backing the attack on top of a legitimate request

Type of ATTACKS

# (B) Inferential Attack

- There is no actual transfer of data, but the attacker is able to reconstruct the information by sending particular requests and observing the resulting behavior of the Website/database server
- Include:
  - Illegal/logically incorrect queries
    - This attack lets an attacker gather important information about the type and structure of the backend database of a Web application
    - The attack is considered a preliminary, information-gathering step for other attacks
  - Blind SQL injection
    - Allows attackers to infer the data present in a database system even when the system is sufficiently secure to not display any erroneous information back to the attacker

Type of ATTACKS.

# (c) Out-of-Band Attack

- Data are retrieved using a different channel
- This can be used when there are limitations on information retrieval, but outbound connectivity from the database server is lax

# SQLi Countermeasures

- Three types:

see next slide

**Defensive coding**

- Manual defensive coding practices
- Parameterized query insertion
- SQL DOM

**Detection**

- Signature based
- Anomaly based
- Code analysis

**Run-time prevention**

- Check queries at runtime to see if they conform to a model of expected queries

# SQLi Safety Guidelines

To counter SQL injection attacks, you need to:

→ use vendor/known libraries. Don't do it yourself

- **Constrain and sanitize input data.** Check for known good data by validating for type, length, format, and range.
- **Use type-safe SQL parameters for data access.** You can use these parameters with stored procedures or dynamically constructed SQL command strings. Parameter collections such as **SqlParameterCollection** provide type checking and length validation. If you use a parameters collection, input is treated as a literal value, and SQL Server does not treat it as executable code. An additional benefit of using a parameters collection is that you can enforce type and length checks. Values outside of the range trigger an exception. This is a good example of defense in depth.
- **Use an account that has restricted permissions in the database.** Ideally, you should only grant execute permissions to selected stored procedures in the database and provide no direct table access.
- **Avoid disclosing database error information.** In the event of database errors, make sure you do not disclose detailed error messages to the user.

# Parameterize SQL

```
protected void btn_Login_Click(object sender, EventArgs e)
{
    string constr = System.Configuration.ConfigurationManager.ConnectionStrings["Constr"].
    SqlConnection con = new SqlConnection(constr);
    string sql = "select count(UserID) from user_login where UserID=@UserID and pwd=@pwd";
    SqlCommand cmd = new SqlCommand(sql, con);
    SqlParameter[] param = new SqlParameter[2];
    param[0] = new SqlParameter("@UserID", txtUSerID.Text);
    param[1] = new SqlParameter("@pwd", txtPwd.Text);
    cmd.Parameters.Add(param[0]);
    cmd.Parameters.Add(param[1]);
    con.Open();
    object res = cmd.ExecuteScalar();
    con.Close();
    if (Convert.ToInt32(res) > 0) Response.Redirect("Home.aspx");
    else
    {
        Response.Write("Invalid Credentials");
        return;
    }
}
```

# Data Center Security

- Data center:
  - An enterprise facility that houses a large number of servers, storage devices, and network switches and equipment
  - The number of servers and storage devices can run into the tens of thousands in one facility
  - Generally includes redundant or backup power supplies, redundant network connections, environmental controls, and various security devices
  - Can occupy one room of a building, one or more floors, or an entire building
- Examples of uses include:
  - Cloud service providers
  - Search engines
  - Large scientific research facilities
  - IT facilities for large enterprises

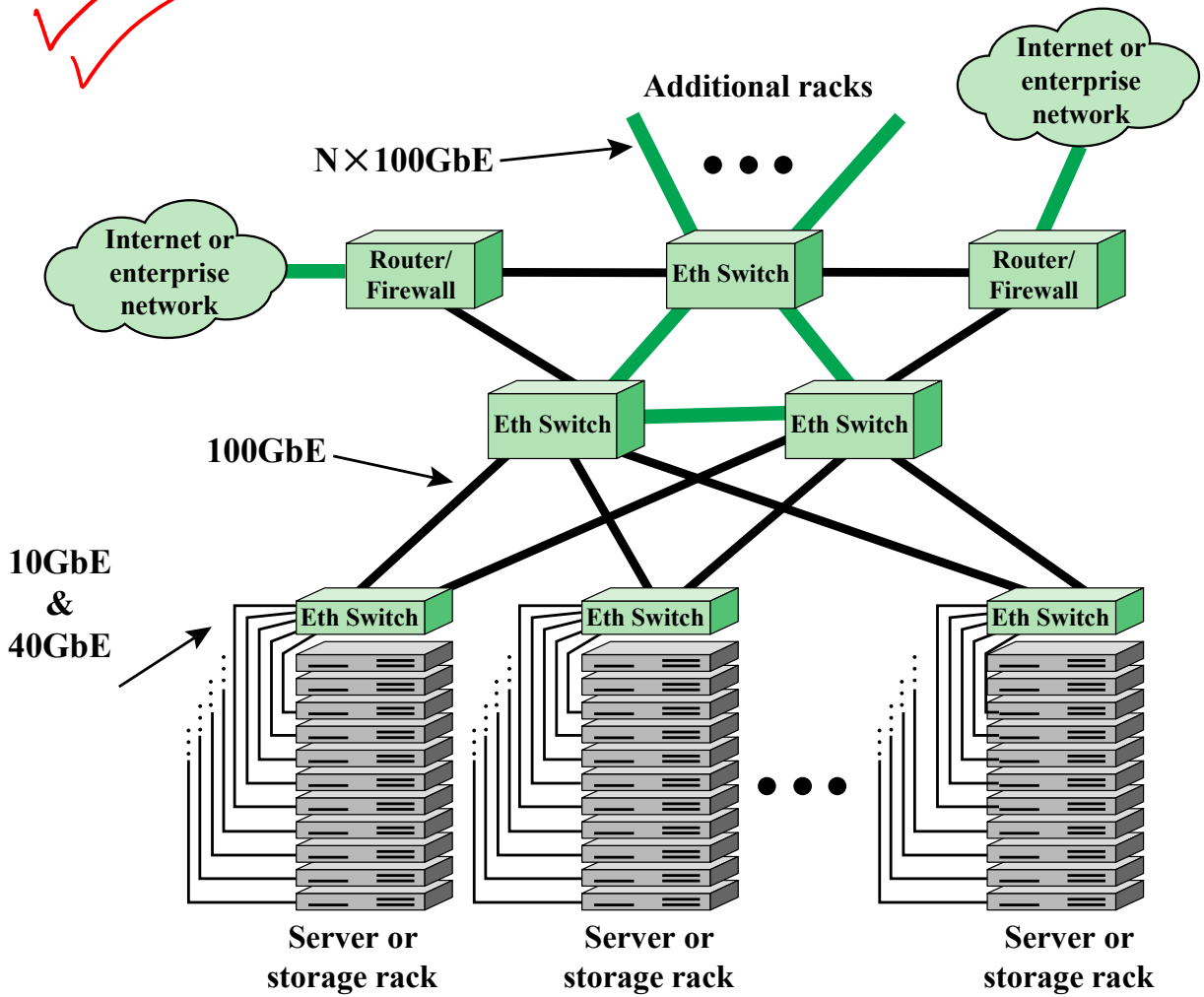



**Figure 5.11 Key Data Center Elements**

| | | |
|---|---|---|
| A | **Data Security** | **Encryption, Password policiy, secure IDs, Data Protection (ISO 27002), Data masking, Data retention, etc.** |
| B | **Network Security** | **Firewalls, Anti-virus, Intrusion detection/prevention, authentication, etc.** |
| C | **Physical Security** | **Surveillance, Mantraps, Two/three factor authentication, Security zones, ISO 27001/27002, etc.** |
| D | **Site Security** | **Setbacks, Redundant utilities Landscaping, Buffer zones, Crash barriers, Entry points, etc.** |

**Figure 5.12 Data Center Security Model**

# Reading Only. TIA-492

- The Telecommunications Industry Association (TIA)
- TIA-492 (*Telecommunications Infrastructure Standard for Data Centers*) specifies the minimum requirements for telecommunications infrastructure of data centers
- Includes topics such as:
  - Network architecture
  - Electrical design
  - File storage, backup, and archiving
  - System redundancy
  - Network access control and security
  - Database management
  - Web hosting
  - Application hosting
  - Content distribution
  - Environmental control
  - Protection against physical hazards
  - Power management

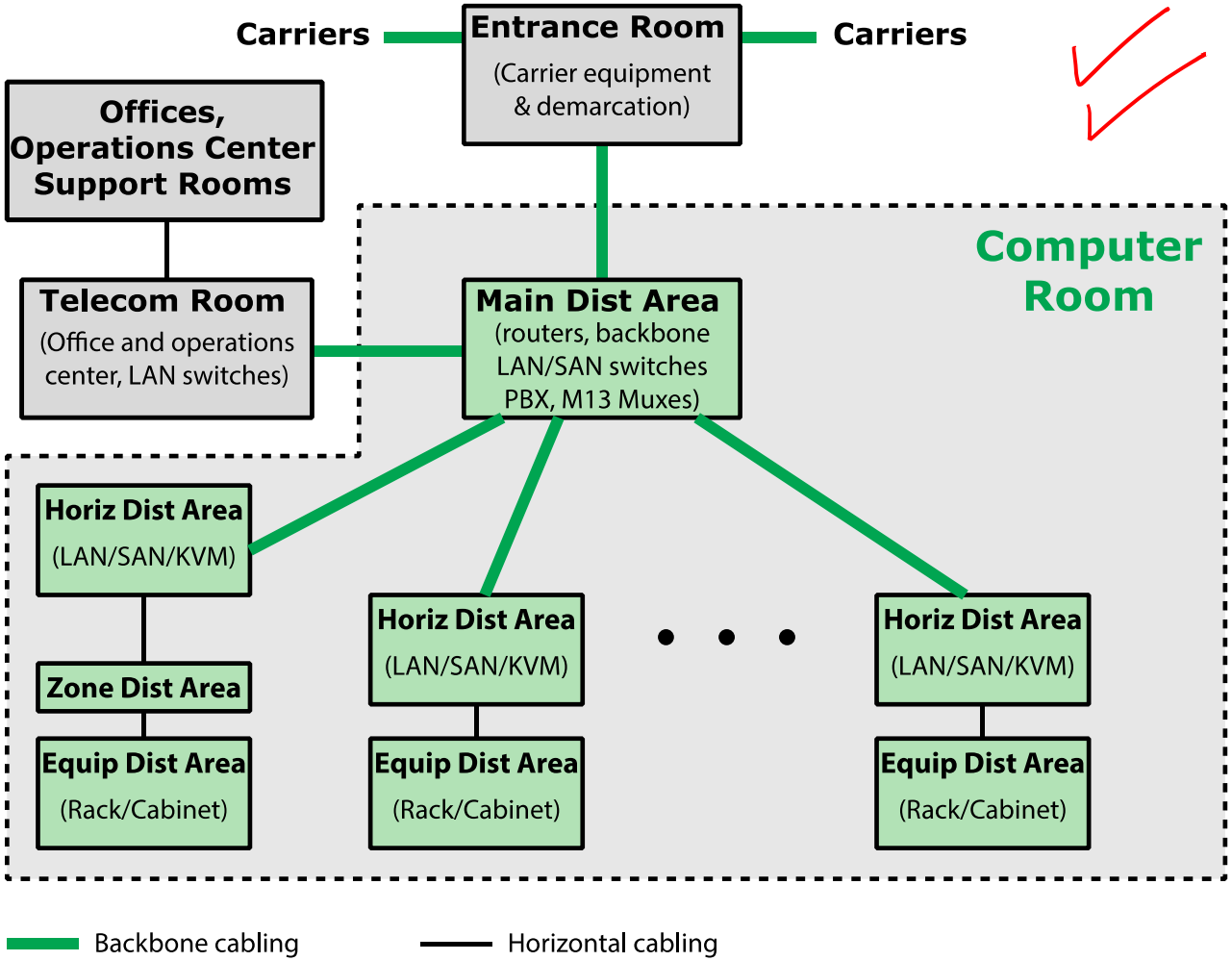



**Figure 5.13 TIA-942 Compliant Data Center Showing Key Functional Areas**

# Table 5.4

## Data Center Tiers Defined in TIA-942

| Tier | System design | Availability /Annual Downtime |
|---|---|---|
| 1 | •Susceptible to disruptions from both planned and unplanned activity<br>•Single path for power and cooling distribution, no redundant components<br>•May or may not have raised floor, UPS, or generator<br>•Takes 3 months to implement<br>•Must be shut down completely to perform preventive maintenance | 99.671%/<br>28.8 hours |
| 2 | •Less susceptible to disruptions from both planned and unplanned activity<br>•Single path for power and cooling distribution, includes redundant components<br>•Includes raised floor, UPS, and generator<br>•Takes 3 to 6 months to implement<br>•Maintenance of power path and other parts of the infrastructure require a processing shutdown | 99.741%/<br>22.0 hours |
| 3 | •Enables planned activity without disrupting computer hardware operation but unplanned events will still cause disruption<br>•Multiple power and cooling distribution paths but with only one path active, includes redundant components<br>•Takes 15 to 20 months to implement<br>•Includes raised floor and sufficient capacity and distribution to carry load on one path while performing maintenance on the other | 99.982%/<br>1.6 hours |
| 4 | •Planned activity does not disrupt critical load and data center can sustain at least one worst-case unplanned event with no critical load impact<br>• Multiple active power and cooling distribution paths, includes redundant components<br>•Takes 15 to 20 months to implement | 99.995%/<br>0.4 hours |

Reading Quiz

(Table is on page 177 in textbook)

# Summary

- The need for database security
- Database management systems
- Relational databases
  - Elements of a relational database system
  - Structured Query Language
- SQL injection attacks
  - A typical SQLi attack
  - The injection technique
  - SQLi attack avenues and types
  - SQLi countermeasures
- Database access control
  - SQL-based access definition
  - Cascading authorizations
  - Role-based access control
- Inference
- Database encryption
- Data center security
  - Data center elements
  - Data center security considerations
  - TIA-492 — reading only