COMMUNICATION AND INFORMATION AGENCY OF UZBEKISTAN

TASHKENT UNIVERSITY OF INFORMATION TECHNOLOGIES

For the Chief of

Defense Department

_____

_____2011 y.

Bachelor's final qualification work

Titled: "Developing the program complex web-servers from internal and external threats"

Graduate _____ U.R.Akhmedjanov
(signature) (name)

Advisor _____ K.A. Tashev
(signature) (name)

Reviewer _____ _____
(signature) (name)

ES and LP advisor_____ _____
(signature) (name)

Tashkent 2011

COMMUNICATION AND INFORMATION AGENCY OF UZBEKISTAN

TASHKENT UNIVERSITY OF INFORMATION TECHNOLOGIES

Faculty IT chair IT

Course (specialization) Information Security

Direction № 5523500

Approved

Chief of chair _____

«____»_____

Assignment

For the Bachelor's final qualifying work

Akhmedjanov Umid Rustamovich

(last name, name, patronymic)

1. The diploma's theme. "Developing the program complex web-servers from internal and external threats"

2. It is approved by the order on university from « » 20 11 y. №

3. Deadline for final work.

4. Raw data for work References, problems and possible solutions in Networ

Security. Standards in using security tools and implement the suggested solution into

TUIT network based on Linux system.

5. The contents of a settlement-explanatory note (the list of questions liable for working up) Introduction, 1) Overwiev of web servers 2) Analyzing methods and tools of protecting web sites 3) Producing protecting system of web servers from external effects 4) Safety of vital activities

6. The list of drawing

Presentation file

7. Date of delivery of the task

Advisor _____

(signature)

Assignment accept_____

(signature)

8. Advisers on separate sections of final work

Section	Surname, Name, Patronymic of chief	Signature, date	
		Assignment given	Assignment taken
Chapter 1	Tashev K.A.		
Chapter 2	Tashev K.A.		
Chapter 3	Tashev K.A.		
Chapter 4			

9. The schedule of performance of work

№	The name of work's section	Date of performance	The record of advisor about performance
1	Introduction	20.01.2011 – 15.02.2011	
2	"Overwiev of web servers"	16.02.2011- 28.03.2011	
3	"Analyzing methods and tools of protecting web sites "	1.04.2011 – 10.05.2011	

4 5	“Producing protecting system of web servers ” “Safety of vital activities”	11.05.2011- 26.05.2011	

Graduating student _____ «____»_____2011 y.

(signature)

Advisor _____ «____»_____2011 y.

(signature)

I. OVERWIEV OF WEB SERVERS

1.1. The architecture of Web Servers and their security issues

A web server is a combination of the two things a computer machine and software installed on it such as Apache and IIS to process the HTTP requests from the client browsers. A web server is also known as the HTTP server. Client's software (browser) sends a request to the web server for the specific page and if the web server finds this page it sends back to the browser, which then displays it at the client computer.

Web server can refer to either the hardware (the computer) or he software (the computer application) that helps to deliver content that can be accessed through the Internet.

The most common use of Web servers is to host Web sites but there are other uses like data storage or for running enterprise applications.

The primary function of a web server is to deliver web pages on the request to clients. This means delivery of HTML documents and any additional content that may be included by a document, such as images, style sheets and JavaScripts.

A client, commonly a web browser or web crawler, initiates communication by making a request for a specific resource using HTTP and the server responds with the content of that resource or an error message if unable to do so. The resource is typically a real file on the server's secondary memory, but this is not necessarily the case and depends on how the web server is implemented.

While the primary function is to serve content, a full implementation of HTTP also includes ways of receiving content from clients. This feature is used for submitting web forms, including uploading of files.

Many generic web servers also support server-side scripting, e.g., Apache HTTP Server and PHP. This means that the behaviour of the web server can be scripted in separate files, while the actual server software remains unchanged. Usually, this function is used to create HTML documents "on-the-fly" as opposed to returning fixed documents. This is referred to as dynamic and static content

respectively. The former is primarily used for retrieving and/or modifying information from databases. The latter is, however, typically much faster and more easily cached.

Web servers are not always used for serving the World Wide Web. They can also be found embedded in devices such as printers, routers, webcams and serving only a local network. The web server may then be used as a part of a system for monitoring and/or administrating the device in question. This usually means that no additional software has to be installed on the client computer, since only a web browser is required (which now is included with most operating systems).

It is clear that, every web server has a unique IP address and they allocate a unique or shared IP addresses to the websites they host. Apache web server software is the most popular web server followed by the Microsoft's IIS (Internet Information Server). When you browsers send out a request for the specific website like www.tuit.uz, your browse sends the request for the tuit.uz IP address for example IP address of www.tuit.uz is 195.158.2.220.

This request also includes the return information. When the web server receives the request it process the request and sends out the page in the HTML code to your IP address. Your browser interprets the HTML and displays the graphical website on your computer. The more powerful is your server and faster it serves out the requests. Every computer on the internet that hosts a website must have a unique IP address and the web server software like Apache and IIS. Before choosing a web server you need to know it's compatibility with the operating system, it's capability to handle the server side programming, response time, reliability, error handling and the security characteristics.

Currently in the world, there is also a trend of increasing practical use of government and commercial organizations, public Web-portals connected to the Internet. Portals of this type can be used to solve various problems, such as advertising on the Internet nature of the company, an organization of Internet commerce, or to ensure the system "Client-Bank". This is facilitated by the fact

that today in the domestic market of information technology are a few ready-made industrial solutions, based on which it is possible to construct full-featured Web-portals. These solutions include product family «Internet Information Services» company Microsoft, «Sun ONE Portal» Sun Microsystems, and «WebSphere» of IBM.

Typical architecture of Web-portal, as a rule, includes the following components:

- Public Web-servers that provide users access the Internet portal to information resources;
- Cache servers that provide temporary storage of copies of the resources that are accessed by Internet users. When accessing the resources of Web-portal initially tries to extract the resource from the memory cache servers, and only if the resource is not there, then the request is forwarded to public Web-servers. The use of cache servers reduces the load on the public servers as well as reduce the time users access to the cached resources;
- DNS-servers that provide the ability to convert symbolic names to servers Web-portal to their corresponding IP-address;
- Application servers that have installed software specially designed for content management of Web-portal;
- Database servers that provide centralized storage of information resources, Web-portal;
- Communications equipment, providing the interaction between different servers Web-portal.

Typically, servers of Web-hosted portal in the Internet service providers who are able to provide the necessary bandwidth of the channels through which the portal servers are connected to the Internet. Managing Web-portal, in this case is carried out remotely via the Internet workstations (AWS) administrators. Generalized architecture of Web-portal is shown in Fig. 1.

Given the fact that the resources of a public Web-based portal for the definition open to anyone on the Internet, they become potential targets for attack intruders. It should be noted that over the past few years have seen considerable growth of information attacks, most of which focused specifically on public resources, which include Web-based portals. Attacks violators may be directed to a breach of confidentiality, integrity or availability of information resources stored on servers Web-portal.

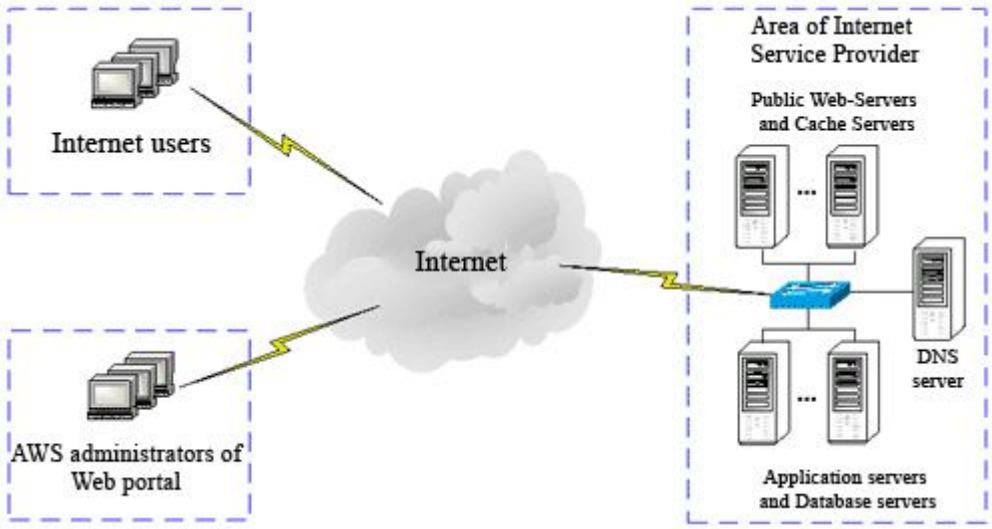


Fig. 1.1 Typical architecture of Web-portal

To protect Web-portal is most advisable to adopt an integrated approach combining organizational and technical means of protection. Organizational remedies associated with the development and implementation of legal documents, such as politics and the concept of information security of Web-portal, job descriptions for staff work with the automated system portal, etc. Technical same remedies are implemented using appropriate software, hardware or software and hardware tools that provide performance goals and objectives identified in the relevant legal documents. Using an integrated approach involves pooling of technical means of Web-portal to an integrated suite that includes antivirus protection subsystem, integrity, access control, intrusion detection, security analysis, cryptographic protection, and control subsystem. The following describes the basic features of these subsystems, as well as features of their application for the protection of Web-portal. [3]

1.2 Threats to Web Servers

A web server delivers data to users' web browsers and picks up their responses and transports this data back to the database. There are a number of different web servers used in web applications. Microsoft Internet Information Services (IIS) and Apache are two examples.

Vulnerabilities have been found in web servers that may allow hackers to damage business systems and gain access through the server to other systems. These vulnerabilities are usually fixed quickly, but new problems are constantly being discovered.

Below, there is given some examples about threats:

Denial of service. The denial of service (DoS) attack is one of the real "old-school" attacks that a server can face. The attack is very simple, and nowadays it's carried out by those individuals commonly known as *script kiddies,* who basically have a low skill level. In a nutshell, a DoS attack is an attack in which one system attacks another with the intent of consuming all the resources on the system (such as bandwidth or processor cycles), leaving nothing behind for legitimate requests. Generally, these attacks have been relegated to the category of annoyance, but don't let that be a reason to lower your guard, because there are plenty of other things to keep you up at night.

Distributed denial of service. The distributed DoS (DDoS) attack is the big brother of the DoS attack and as such is meaner and nastier (yes, I do have two older brothers: Why do you ask?). The goal of the DDoS attack is to do the same thing as the DoS, but on a much grander and more complex scale. In a DDoS attack, instead of one system attacking another, an attacker uses multiple systems to target a server (your server), and by *multiple systems* I mean (in some cases) not hundreds or thousands, but more on the order of hundreds of thousands. Where DoS is just an annoyance, a DDoS attack can be downright deadly, as it can take a server offline quickly. The good news is that the skill level required to pull a DDoS attack off is fairly high.

Some of the more common DDoS attacks include:

- **FTP bounce attacks.** A File Transfer Protocol (FTP) bounce attack is enacted when an attacker uploads a specially constructed file to a vulnerable FTP server, which in turn forwards it to another location, which generally is another server inside the organization. The file that is forwarded typically contains some sort of payload designed to make the final server do something that the attacker wants it to do.
- **Port scanning attack.** A port scanning attack is performed through the structured and systematic scanning of a host. For example, someone may scan your Web server with the intention of finding exposed services or other vulnerabilities that can be exploited. This attack can be fairly easily performed with any one of a number of port scanners available freely on the Internet. It also is one of the more common types of attacks, as it is so simple to pull off that script kiddies attempt it just by dropping the host name or IP address of your server (however, they typically don't know how to interpret the results). Keep in mind that a more advanced attacker will use port scanning to uncover information for a later effort.
- **Ping flooding attack.** A ping flooding attack is a simple DDoS attack in which a computer sends a packet (ping) to another system with the intention of uncovering information about services or systems that are up or down. At the low end, a ping flood can be used to uncover information covertly, but throttle up the packets being sent to a target or victim so that now, the system will go offline or suffer slowdowns. This attack is "old school" but still very effective, as a number of modern operating systems are still susceptible to this attack and can be taken down.
- **Smurf attack.** This attack is similar to the ping flood attack but with a clever modification to the process. In a Smurf attack, a ping command is sent to an intermediate network, where it is amplified and forwarded to the victim. What was

once a single "drop" now becomes a virtual tsunami of traffic. Luckily, this type of attack is somewhat rare.

- **SYN flooding.** This attack requires some knowledge of the TCP/ IP protocol suite—namely, how the whole communication process works. The easiest way to explain this attack is through an analogy. This attack is the networking equivalent of sending a letter to someone that requires a response, but the letter uses a bogus return address. That individual sends your letter back and waits for your response, but the response never comes, because it went into a black hole some place. Enough SYN requests to the system, and an attacker can use all the connections on a system so that nothing else can get through.
- **P fragmentation/fragmentation attack.** In this attack, an attacker uses advanced knowledge of the TCP/IP protocol to break packets up into smaller pieces, or "fragments", that bypass most intrusion-detection systems. In extreme cases, this type of attack can cause hangs, lock-ups, reboots, blue screens, and other mischief. Luckily, this attack is a tough one to pull off.
- **Simple Network Management Protocol (SNMP) attack.** SNMP attacks are specifically designed to exploit the SNMP service, which is used to manage the network and devices on it. Because SNMP is used to manage network devices, exploiting this service can result in an attacker getting detailed intelligence on the structure of the network that he or she can use to attack you later.

Web page defacement. Web page defacement is seen from time to time around the Internet. As the name implies, a Web page defacement results when a Web server is improperly configured, and an attacker uses this flawed configuration to modify Web pages for any number of reasons, such as for fun or to push a political cause.

SQL injection. *Structured Query Language (SQL) injections* are attacks carried out against databases. In this attack, an attacker uses weaknesses in the design of the database or Web page to extract information or even manipulate

information within the database. Although I can't get into the specifics of how to pull this type of attack off, you can look it up if you have SQL knowledge, which—if you're hosting database on your Web server—you should have.

Examples of the results of this type of attack include:

- Circumventing the username and password required to gain access to secure data, such as your ledgers. The hacker could gain enough access to change your stored data.
- Fooling your web applications into accepting a fictitious trading account name - one that does not exist but the application thinks represents a valid account so that the hacker is able to order goods fraudulently.

SQL injection can only occur if:

- Your web application does not properly check the input that the user provides. The threat relies on the hacker carefully crafting a strange-looking response which is not checked by the application and is sent directly to the DBMS.
- The database is designed in an insecure way, allowing users to directly change the data. The techniques necessary to make the database secure should be understood by the application designer.

You are unlikely to be directly involved in writing a web application. You can, however, ask the supplier what precautions they have taken against SQL injection and other potential vulnerabilities.

Poor coding. Anyone who has been a developer or worked in information technology has seen the problems associated with sloppy or lazy coding practices. Poor coding problems can result from any one of a number of factors, including poor training, new developers, or insufficient quality assurance for an application. At its best, poor coding can be an annoyance, where features don't work as advertised; at its worst, applications can have major security holes.

Shrink-wrapped code. This problem is somewhat related to the above issues with poor coding, but with a twist: Basically, this problem stems from the convenience of obtaining precompiled or pre-written components that can be used

as building blocks for your own application, shortening your development cycle. The downside is that the components you're using to help build your application may not have gone through the same vetting process as your in-house code, and applications may have potential problem areas. Additionally, it's not unheard of for developers who don't really know how to analyze the code and understand what it's actually doing to put so-called "shrink-wrapped" components in applications. In at least one case I can think of, I'm aware of a developer using a piece of shrink-wrapped code to provide an authentication mechanism for an application that was actually authenticating users, but also covertly e-mailing the same credentials to a third-party.[4]

1.3 Methods and means of protecting Web servers

A Web site is a powerful tool that enables businesses, government, and private users to share information and conduct business on the Internet. Organizations - small and large, private and public – are devoting many resources to creating attractive, attention-getting Web sites, but they may be neglecting basic security controls. Recent attacks on Web sites have shown that the computers that support Web sites are vulnerable to attacks that can range from minor nuisances to significant interruptions of service. This ITL Bulletin discusses the most commonly employed methods for protecting Web servers and provides practical guidance on steps that organizations can take to reduce the threat of attacks.

Creating a Plan to Secure Your Web Server. While most incidents cause minor embarrassment or inconvenience, it is possible for an intruder to cause real problems and severe losses. Every organization should establish a security program that assesses the risks of attacks and takes steps to reduce the risks to an acceptable level. Each organization has to decide its sensitivity to risk and how open it wants to be to the external world.

When resources are limited, the cost of security incidents should be considered, and the investment in protective measures should be concentrated on areas of highest sensitivity.

There are three levels of Web security techniques that can be applied:

Level 1: Minimum Security

1. Upgrading Software/Installing Patches
2. Using Single Purpose Servers
3. Removing Unnecessary Applications

Level 2: Penetration Resistance

1. External Firewalls
2. Remote Administration Security
3. Restrict Server Scripts
4. Web Server Shields with Packet Filtering
5. Education and Personnel Resource Allocation
6. Techniques listed in level 1

Level 3: Attack Detection and Mitigation

1. Separation of Privilege
2. Hardware-Based Solutions
3. Internal Firewalls
4. Network-Based Intrusion Detection
5. Host-Based Intrusion Detection
6. Techniques listed in level 2

Techniques to Secure Web Servers

The most common methods for protecting Web servers include:

* Removal of unnecessary software,

* Detection of attacks upon a Web server,

* Correction of flaws in remaining software,

* Restriction of an attacker's actions once a part of a Web server is compromised,

* Protection of the rest of the network if a Web server is compromised.[5]

Upgrading Software/Installing Patches. One of the simplest and yet most effective techniques for reducing risk is the installation of the latest software updates and patches. Web servers should be frequently (sometimes daily) examined to determine what software needs to be updated or patched. (NIST is actively working with other government agencies to develop tools to assist in the finding and applying of patches.

When available, details will appear on the NIST Computer Security Resource Clearinghouse. Any software on a Web server that an attacker could use to penetrate the system must be regularly updated. Software in this category includes the operating system, servers or any software that receives network packets, software running as root or administrator, and security software. The following process should be followed:

* Make a list of such software and write down the associated version numbers.

* Find the Web page for each piece of software and make sure that you have installed the latest version.

* Find and install the available patches for the applicable version of the software. Each software vendor provides unique instructions on how to install its patches and usually these instructions are very simple. Be careful to follow vendor instructions; patches must often be installed in a set sequence for the process to work.

* Verify that patched software functions correctly.

Using Single-Purpose Servers. Organizations should run Web servers on computers dedicated exclusively to that task. A common mistake is to try to save money by running multiple servers on the same host. For example, it is not uncommon to run an e-mail server, Web server, and database server on the same computer. However, each server run on a host provides an attacker with avenues for attack. Each newly installed server then increases the organization's reliance

upon that host while simultaneously decreasing its security. Given the decreasing cost of hardware and the increasing importance of having fast Web servers, it is generally effective to buy a dedicated host for each Web server. Also, in situations where a Web server constantly interacts with a database, it is best to use two separate hosts.

Removing Unnecessary Applications. All privileged software not specifically required by the Web server should be removed. For the purposes of this document, privileged software is defined as software that runs with administrator privileges or that receives packets from the network. Operating systems often run a variety of privileged programs by default. Many systems administrators are not even aware of the existence of many of these programs. Each privileged program provides another avenue by which an attacker can compromise a Web server. It is therefore crucial that Web servers be purged of unnecessary programs. For greater security and because it is often difficult to identify what software is privileged, many systems administrators remove all software not needed by a Web server.

External Firewalls. Install public Web servers outside of an organization's firewall. In this configuration, the firewall prevents the Web server from sending packets into an organization's network. If an attacker on the Internet penetrates the external Web server, they have no more access to the organization's internal network than they had before. If a Web server is inside the organization's firewall and is penetrated by an attacker on the Internet, the attacker can use the Web server as a launching point for attacks on the internal systems. Thus, these attacks would completely bypass the security provided by the firewall.

Remote Administration Security. Since it is often inconvenient to administer a host from the physical console, system administrators often install software on Web servers to allow remote administration. From a security perspective, this practice is dangerous and should be minimized or eliminated. In order to increase the security where this practice is necessary:

* Encrypt remote administration traffic such that attackers monitoring network traffic cannot obtain passwords or inject malicious commands into conversations.

* Use packet filtering (see description below) to allow remote administration only from a designated set of hosts.

* Maintain this designated set of hosts at a higher degree of security than normal hosts.

* Do not use packet filtering as a replacement for encryption since attackers can spoof Internet Protocol (IP) addresses. (With IP spoofing, an attacker lies about their location by sending messages from an IP address other than their own.)

Restrict Server Scripts. Most Web sites contain scripts (small programs) created locally by Web site developers. A Web server runs these scripts when a user requests a particular page. Attackers can use these scripts to penetrate Web sites by finding and exercising flaws in the code. To find such flaws, an attacker does not necessarily need the script source code. Scripts must be carefully written with security in mind and system administrators should inspect them before placing them on a Web site. Do not allow scripts to run arbitrary commands on a system or to launch insecure (or non-patched) programs. Scripts should restrain users to doing a small set of well-defined tasks. They should carefully restrict the size of input parameters so that an attacker cannot give a script more data than it expects. If an attacker is allowed to do this, a system can often be penetrated using a technique called buffer overflow. (With a buffer overflow attack, an attacker convinces a Web server to run arbitrary code by giving it more information than it expected to receive.) Run scripts with non-administrator privileges to prevent an attacker from compromising the entire Web server in the event that a script contains flaws.

Web Server Shields with Packet Filtering. A router set up to separate a Web server from the rest of the network can shield a Web server from many

attacks. The router can thwart attacks before they reach the Web server by dropping all packets that do not access valid Web server services. Typically, the router should drop all network packets that do not go either to the Web server (port 80) or to the remote administration server being used. For additional security, only allow a pre-approved list of hosts to send traffic to a Web server's remote administration server. By doing so, an attacker can only compromise a Web server using the remote administration server via a restricted set of network paths. The filtering router shield offers similar protection to that of removing all unneeded software from a host since it prevents an attacker from requesting certain vulnerable services. Be aware that setting up a router with many filtering rules may noticeably slow its ability to forward packets.

Education and Personnel Resource Allocation. Attackers are able to penetrate most Web servers because the systems administrators are either not knowledgeable about Web server security or did not take the time to properly secure the system. Web site administrators must be trained about Web server security techniques and rewarded for spending time securing the site. Several excellent books and training seminars exist to aid administrators in securing Web sites.

Separation of Privilege. Regardless of the security measures established for a Web server, penetration may still occur. If this happens, it is important to limit the attacker's actions on the penetrated host. Separation of privilege is a key concept for restricting actions once a part of the host is penetrated. To establish such control, partition the various host resources among a set of user accounts. An attacker who penetrates some software will then be limited to acting within that single user account instead of having control over the entire system. For example, a Web server can run as one user, but the Web pages can be owned by another user and with the Web server given read-only access. Then, if attackers penetrate the Web server, they cannot change the Web pages owned by other users.

Likewise, intrusion detection software can run as another user to protect it from being modified by an attacker penetrating the Web server user. For the best security, run the Web server process as a user that has write privilege only in a few privately owned temporary directories. This requires storing the Web server software as read-only under one user but running it as a different user.

Hardware-Based Solutions. Hardware can implement separation of privilege concepts with a greater degree of security than software because hardware is not as easily modified as software. With software implementations, if the underlying operating system is penetrated, the attacker has complete control of all files on a Web server. Using read-only external hard disks or CD-ROMs, Web pages and even critical software can be stored in a way that an attacker cannot modify the files. The usual configuration is for the Web server to have a read-only port to the external hard disk while another well-protected computer has a read-write port so that the Web pages can be updated. Note that an attacker who penetrates a protected Web server can still copy data, change the copied data, and serve up the changed pages.

Internal Firewalls. Modern Web servers often serve as front ends to complex and possibly distributed applications. In this situation, a Web server often communicates with several other hosts, each of which contains particular data or performs particular computations. It is tempting to locate these computers inside of an organization's firewall for ease of maintenance and to protect these important computers. However, if an attacker can compromise a Web server, these back end systems may be penetrated using the Web server as a launching point. Instead, it is a good idea to separate the Web server back end systems from the rest of the organization's networks using an internal firewall. Then, penetration of the Web server and subsequently the Web server's back end systems does not provide access to the rest of the organization's networks.

Network-Based Intrusion Detection. Despite all attempts to patch a Web server and to securely configure it, vulnerabilities may still exist that are known to

the outside world. Also, the Web server may be perfectly secure but an attacker may cleverly overwhelm the host's services such that it ceases to operate. In this kind of environment, it is important to know when your Web server has been compromised or shut down so that service can be quickly restored. Network-based intrusion detection systems (IDSs) monitor network traffic to determine whether a Web server is under attack or has been compromised or disabled. Modern IDSs have the ability to launch a limited response to attacks or notify systems administrators via e-mail, pagers, or messages on a security console. Typical automated responses include killing network connections and blocking sets of IP addresses.

Host-Based Intrusion Detection. Host-based IDSs reside on a Web server. Thus, they are better positioned to determine the state of the Web server than a network-based IDS. They provide the same benefits as network-based IDSs and in some circumstances can detect attacks better because they have finer grained access to the Web server's state. However, some drawbacks exist. An attacker that penetrates a Web server can disable a host-based IDS, thereby preventing it from issuing a warning. In addition, remote denial-of-service (DOS) attacks often disable host-based IDSs while disabling the Web server. Remote DOS attacks enable an attacker to remotely shut down a Web server without actually penetrating it. Thus, host-based IDSs are useful but they should be used in conjunction with the typically more secure network-based IDSs.

Limitations of Existing Solutions and Gaining Additional Assurance. Considerable research addresses issues of proving software secure. In some cases, it is possible to do this but it is very costly and time-consuming. Usually, by the time software is proven secure, it is obsolete and replaced with an unproven new version. Therefore, today's software is not proven secure and application of standard Web security techniques cannot guarantee that a Web server will be impenetrable.

However, a Web server can be made quite resistant to attacks by using the stated Web server security techniques in addition to using trustworthy software. By trustworthy, we mean software that can be demonstrated by some measure to be secure. The security afforded by software can be assessed by studying past vulnerabilities, using software specifically created with security as the principle goal, and using software evaluated by trusted third parties.

First, some level of assurance in software can be gained by looking at the past vulnerabilities discovered in different Web server software. The number of past vulnerabilities is an indicator of future vulnerabilities and also reflects how well the software was crafted. Trustworthiness is directly related to the quality of the software product. A poorly crafted product built explicitly to meet security needs remains a poorly crafted product and therefore not trustworthy.

Second, some companies specialize in creating very secure Web server software and some boast that no vulnerabilities have ever been discovered. Users have to balance vendor's security claims against any security-performance tradeoffs that have been made.

A third way to gain a level of assurance in software is to use evaluated and validated software. Many private-sector organizations perform third-party evaluations of commercial products in order to verify a particular level of security. One of the largest of these efforts is the National Information Assurance Partnership (NIAP). A joint venture between NIST and NSA, NIAP has helped create an international standard (ISO/IEC 15408) for specifying security requirements of IT products and evaluating them to that specification. It provides a framework by which commercial companies can have product claims tested by a third party and (if desired) obtain a certificate of validation from NIAP. Various security-enhanced products are currently under evaluation, including the firewalls of three major U.S. vendors. Look in the future for NIAP-evaluated Web server software.

II. Analyzing methods and tools of protecting WEB Sites

2.1 Methods of protecting from SQL injunctions

SQL injection, an attack method where hacker insert malicious SQL code into a Web form to gain malicious access to resources, applications or databases, has been on the rise with the advancement of automated exploit tools, and the attack method, which can enable data manipulation and the spread of malware, is becoming more advanced and popular among attackers.

The vulnerability is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed. It is an instance of a more general class of vulnerabilities that can occur whenever one programming or scripting language is embedded inside another. SQL injection attacks are also known as SQL insertion attacks

This guide offers expert advice and best practices on SQL injection protection. Learn how to stop SQL injection attacks and how to prevent SQL vulnerabilities from being actively exploited by a hacker. Also included are tips and information on the latest SQL injection defense mechanisms and techniques, such as secure input validation and perimeter-based vendor products.

Often, the availability of SQL-Injection can tell the error, clearly indicating that there was an error in the SQL-query. At the same time, the presence of errors in the SQL-query can be judged indirectly. To test the fully filtered some parameter or not, will give the program a few change the setting. For example, instead http://site/test.php?id=12 pass.

- http://site/test.php?id=12
- http://site/test.php?id=aaa
- http://site/test.php?id=13-1

If the last request to eject the page, similar to http://site/test.php?id=12, in most cases can clearly reveal the existence of vulnerabilities in the unfiltered option.

Analysis of the database using SQL-Injection. Suppose we know about the lack of filtering the id parameter in the script http://site/test.php?id=12. Availability of detailed error messages with the text SQL-query, where the error occurred, will reduce the difficulty of manual SQL-Injection to a minimum. However, much can be done even if error messages are not displayed at all. Should take note of the fact that even if the error is not displayed, you can still clearly judge whether the error occurred, or not (for example, the query returns an empty result).

In particular, it is possible that when an error is returned a response code 500, or is redirected to the main page, whereas with an empty result set will be returned to a blank page. In order to identify these secondary symptoms, you should make a http-requests, which are known, which will lead to a correct (but it returns an empty output) SQL-query, and which will lead to an incorrect SQL-query. For example, if not filtered parameter id:

• http://site/test.php?id=99999, will probably be returned an empty result, while

• http://site/test.php?id=99999 'should generate an error.

Now, knowing how to distinguish false from the empty query, we begin to consistently extract information about the query and the database. Consider the case of SQL-Injection occurs after where. If we consider the database MySQL, is getting information from the database may be possible only if the server is version 4 .*, they have the ability to insert into the query union.

Other types of SQL-Injection. Filter integer values for integer parameters, and quotes for string parameters is not enough sometimes. Sometimes a violation of the software may use% and _ -special characters inside like the query. For example:

• *mysql_query ("select id from users where password like '". Addslashes ($ password). "' And user like '". Addslashes ($ user)."');*

in this case to suit any user password%

In some cases, SQL-Injection is possible even in the parameter, which is converted by methods mod_rewrite module of Apache web server to GETparameter script.For example, type scripts */ news/127.html converted to / news / news.php? id =127* the following rule: RewriteRule ^ / news /(.*) \. html $ "/ news / news.php? id = $1"

This will allow to pass malicious values to the script. So, for example */ news/128-1.html*

If you see error messages, you can immediately find the address of a squeak, and further, to choose the parameters to work with him already. If not, then you can explore the vulnerability directly editing the file name[6].

Protection against SQL-Injection. To protect against all of the above is enough to stick a few simple rules:

1.For integer and fractional values before using them in a query result in sufficient quantity to the desired type.

$ id = (int) $ id; $ total = (float) $ total;

Instead, you can insert a tracking system for testing for SQL injection.

if ((string) $ id <> (string) (int) $ id) { / / write to a log of attempted burglary die ('ops'); }

2. For string parameters, which are not used to like, regexp, etc., escapes quotes. *$ str = addslashes ($ str);* or, better, *mysql_escape_string ($ str)*

3. In lines that are supposed to be used inside like, regexp, etc, should also screen the special characters used in these statements, if necessary. Otherwise, you can document the use of these symbols[7].

2.2 Protecting Web Servers from DoS attacks

A denial-of-service attack (DoS attack) or distributed denial-of-service attack (DDoS attack) is an attempt to make a computer resource unavailable to its intended users. Although the means to carry out, motives for, and targets of a DoS

attack may vary, it generally consists of the concerted efforts of a person or people to prevent an Internet site or service from functioning efficiently or at all, temporarily or indefinitely. Perpetrators of DoS attacks typically target sites or services hosted on high-profile web servers such as banks, credit card payment gateways, and even root name-servers. The term is generally used with regards to computer networks, but is not limited to this field; for example, it is also used in reference to CPU resource management.

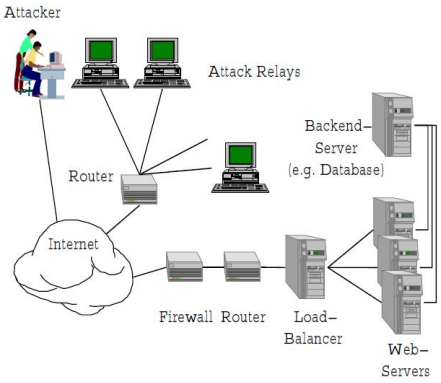


Fig. 2.1 Overview of attack scenario

One common method of attack involves saturating the target machine with external communications requests, such that it cannot respond to legitimate traffic, or responds so slowly as to be rendered effectively unavailable. In general terms, DoS attacks are implemented by either forcing the targeted computer(s) to reset, or consuming its resources so that it can no longer provide its intended service or obstructing the communication media between the intended users and the victim so that they can no longer communicate adequately.

Denial-of-service attacks are considered violations of the IAB's Internet proper use policy, and also violate the acceptable use policies of virtually all Internet service providers. They also commonly constitute violations of the laws of individual nations.

A "denial-of-service" attack is characterized by an explicit attempt by attackers to prevent legitimate users of a service from using that service. There are two general forms of DoS attacks: those that crash services and those that flood services. Attacks can be directed at any network device, including attacks on routing devices and web, electronic mail, or Domain Name System servers.

A DoS attack can be perpetrated in a number of ways. The five basic types of attack are:

1. Consumption of computational resources, such as bandwidth, disk space, or processor time.

2. Disruption of configuration information, such as routing information.

3. Disruption of state information, such as unsolicited resetting of TCP sessions.

4. Disruption of physical network components.

5. Obstructing the communication media between the intended users and the victim so that they can no longer communicate adequately.

A DoS attack may include execution of malware intended to:

- Max out the processor's usage, preventing any work from occurring.
- Trigger errors in the microcode of the machine.
- Trigger errors in the sequencing of instructions, so as to force the computer into an unstable state or lock-up[8].
- Exploit errors in the operating system, causing resource starvation and/or thrashing, i.e. to use up all available facilities so no real work can be accomplished.
- Crash the operating system itself.[9]

Below there are given some type of DoS attacks.

- ICMP flood
- SYN flood
- Teardrop attacks
- Peer-to-peer attacks
- Asymmetry of resource utilization in starvation attacks
- Permanent denial-of-service attacks and etc[24].

Network mechanisms can either be deployed at the victim-network, Intermediate network or at the source-network. DDoS defense mechanisms deployed at the victim network protect this network from DDoS attacks and respond to detected attacks by alleviating the impact on the victim. DDoS defense mechanisms deployed at the intermediate network such as ISP's provide

infrastructural service to a large number of Internet hosts. DDoS defense mechanisms deployed at the source network is to prevent customers using this network from generating DDoS attacks.

Boarder Routers. Many mechanisms for defeating DDoS attacks at routers have been proposed, examples include Ingress, Egress Filtering, and MULTOPS.

Ingress Filtering. Ingress Filtering is an Intermediate-network mechanism. Internet Service Providers (ISPs) can take actions against DoS/DDoS that include: eliminating routing of spoofed packets by discarding any packet that contains any RFC 1918 or reserved IP address in the IP source address or destination address. Also they should perform Ingress filtering [10] on their routers to drop packets with IP addresses outside the range of a customer's network, so that they can prevent attackers from using forged source addresses to launch a DoS attack. The weaknesses of applying ingress filtering technique is that, it does nothing to address flooding attacks that originate from valid IP addresses, and may negatively affect mobile IP services.

Egress Filtering. Egress Filtering is a source-network mechanism. SANS institute urged network administrators to adopt egress filtering which prevents one's network from being the source of forged communications used in DoS attacks [11]. An egress filter is designed for implementation in the routers at the edge of a network. These filters analyze packets as they are forwarded to their intended destination, looking for forged (spoofed) IP addresses. Since any particular network is assigned a specific subset of IP addresses, any packet containing an invalid IP address is assumed to be spoofed, and the filter drops such packets. This ensures that only IP packets with valid source IP addresses leave the network and thus protects the outside from spoofed packets. Egress filtering has two severe shortcomings. Firstly, there is little incentive for an ISPs to provide egress filtering since it does not protect from the attack, it only keeps an attacker from using the network for a DDoS attack. If egress filtering is not employed by a

significant number of networks, it will not be a viable solution to DDoS attacks. Secondly, egress filtering will not detect internally spoofed IP addresses.

MULTOPS Bandwidth Attack Detection. MULTOPS is a source-network defense mechanism. This solution postulates that if a network administrator (a victim) were able to detect an IP addresses that participate in a DDoS attack, then measures could be taken to block only these particular addresses. The solution is a heuristic one, and defines a data-structure that network devices (such as routers) can use to detect (and eliminate) DoS attacks. The Multi-Level Tree for Online Packet Statistics (MULTOPS) is a tree of nodes that contains packet rate statistics for subnet prefixes at different aggregation levels [12]. MULTOPS uses disproportional rates to or from hosts and subnets as a heuristic to detect (and potentially stop) attacks.

Firewalls. Firewalls are victim-network mechanisms. Most firewalls built today are designed to enable a form of protection against SYN floods. Firewalls are better suited to fight the attack because they tend to be designed to examine packets and maintain connection and state information of session traffic. As a countermeasure to DoS attacks, firewalls can be configured as a relay, as a semi-transparent gateway [13], or combine other techniques.

Active Monitoring. This category is of solutions is a victim network mechanisms, it consists of using software agents to continuously monitor TCP/IP traffic in a network at a given place (Router/Firewall). An agent can collect communication control information to generate a view of all connections that can be observed on a monitored network. Furthermore, active monitors can watch for certain conditions to arise and react appropriately. Examples for active monitors are synkill from COAST Laboratory [14], The Nozzle [15], and The SYNDEF [16].

Load Balancing. The last victim-network defense against DoS floods is to distribute the flood against as many hosts or network devices as possible. In the case of commercial web sites or corporate sites that are well known and have

considerable throughput the load balancing is probably already in place. A solution that is based on Class Based Routing mechanisms in the Linux kernel is proposed [13]. The solution is oriented to suite big sites that used load-balancing server and aimed to keep the web servers under attack responding to normal requests. The solution uses a number of configurable input queues on the load balancer and output queues on the web servers. If the traffic monitor in the load balancer detects a possible DoS attack, it slows the traffic from the origination IP address by assigning it to a slower queue or block it at the firewall.

Global Mechanisms. Clearly, Brute-force DDoS floods threaten the Internet as a whole, local solution to the problem become futile. Global solutions are seems better from a technological point of view. Global proposed solutions are:

Improving the security of the entire Internet. Improving the security of all computers linked to the Internet would prevent attackers from finding enough vulnerable computers to break into and plant daemon programs that would turn them into zombies.

Using globally coordinated filters. The strategy here is to prevent the accumulation of a critical mass of attacking packets in time. Once filters are installed throughout the Internet, a victim can send information that it has detected an attack, and the filters can stop attacking packets earlier along the attacking path, before they aggregate to lethal proportions. This method is effective even if the attacker has already seized enough zombie computers to pose a threat.

Tracing the source IP address. The goal of this approach is to trace the intruders' path back to the zombie computers and stop their attacks or, even better, to find the original attacker and take legal actions. If tracing is done promptly enough, it can help to abort the DDoS attack. Catching the attacker would deter repeat attacks. Examples of traceback mechanisms are [17], and [18]. However, two attacker techniques hinder tracing: IP spoofing that uses forged source IP addresses, and the hierarchical attacking structure that detaches the control traffic from the attacking traffic.

2.3 Mechanisms of fighting with Spam information on the Web Sites and Portals.

A spam filter is a program that is used to detect unsolicited and unwanted email and prevent those messages from getting to a user's inbox. Like other types of filtering programs, a spam filter looks for certain criteria on which it bases judgments. For example, the simplest and earliest versions (such as the one available with Microsoft's Hotmail) can be set to watch for particular words in the subject line of messages and to exclude these from the user's inbox. This method is not especially effective, too often omitting perfectly legitimate messages (these are called *false positives*) and letting actual spam through. More sophisticated programs, such as Bayesian filters or other heuristic filters, attempt to identify spam through suspicious word patterns or word frequency.

Considering matters technically - but also with common sense -- what is generally called "spam" is somewhat broader than the category "unsolicited commercial e-mail"; spam encompasses all the e-mail that we do not want and that is only very loosely directed at us. Such messages are not always commercial per se, and some push the limits of what it means to be solicited. For example, we do not want to get viruses (even from our unwary friends); nor do we generally want chain letters, even if they don't ask for money; nor proselytizing messages from strangers; nor outright attempts to defraud us. In any case, it is usually unambiguous whether a message is spam, and many, many people get the same such e-mails.

The spam filtering mechanism relies on the SMTP protocol for accepting unfiltered Internet Email and delivering filtered mail to a recipient mail server.

Unfiltered Internet Email works by having the source mail server identify the correct mail server for a domain and delivering messages to that mail server via

Fig. 2.2 How Email works without Spam filtering

SMTP. The DNS MX record is used to identify the correct mail server for a domain[19].

With Spam and Virus filtering the basic model remains the same except that a cluster of servers dedicated to spam and virus filtering is placed between the source mail server and your normal mail server.

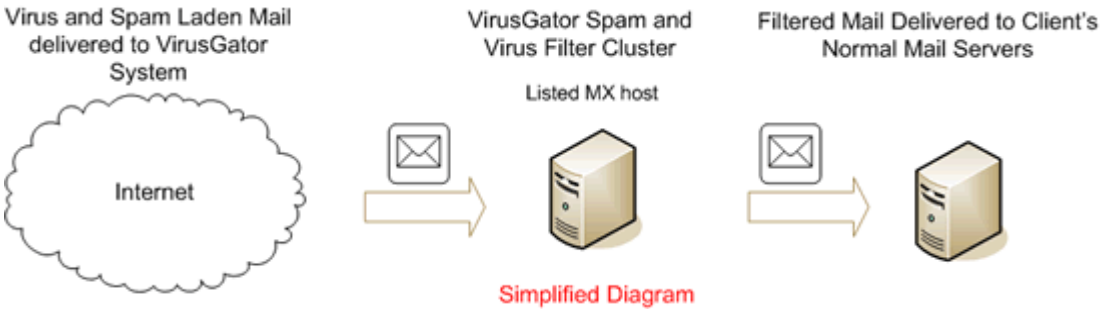


Fig. 2.3 Simplified Spam Filter diagram.

The spam filter servers communicate with both the source mail server and your normal mail server via SMTP. Nothing is changed on your normal mail server except that it should only talk to mail servers in the cluster. In reality, the topology used by our Spam filter is much more robust than the simplified diagram. Below is a more representative diagram of our spam filtering service system.

The placement of the spam filtering cluster servers between the source and your normal mail server is achieved by way of DNS MX records which may be updated through either your own DNS servers, our DNS servers, or your Domain Registrar.

The normal mail server is listed as the mail exchanger (MX) for a domain. Here is a sample MX record:

xyz.com. IN MX 10 mail.xyz.com.

This record says that for xyz.com, all mail should be sent to the host mail.xyz.com. All you need to do to enable Spam filtering for the domain is replace the host mail.xyz.com with the hostname of our spam filtering server cluster.

Here is a sample spam filter configured MX record : xyz.com. IN MX 10 chomp.virusgator.com.

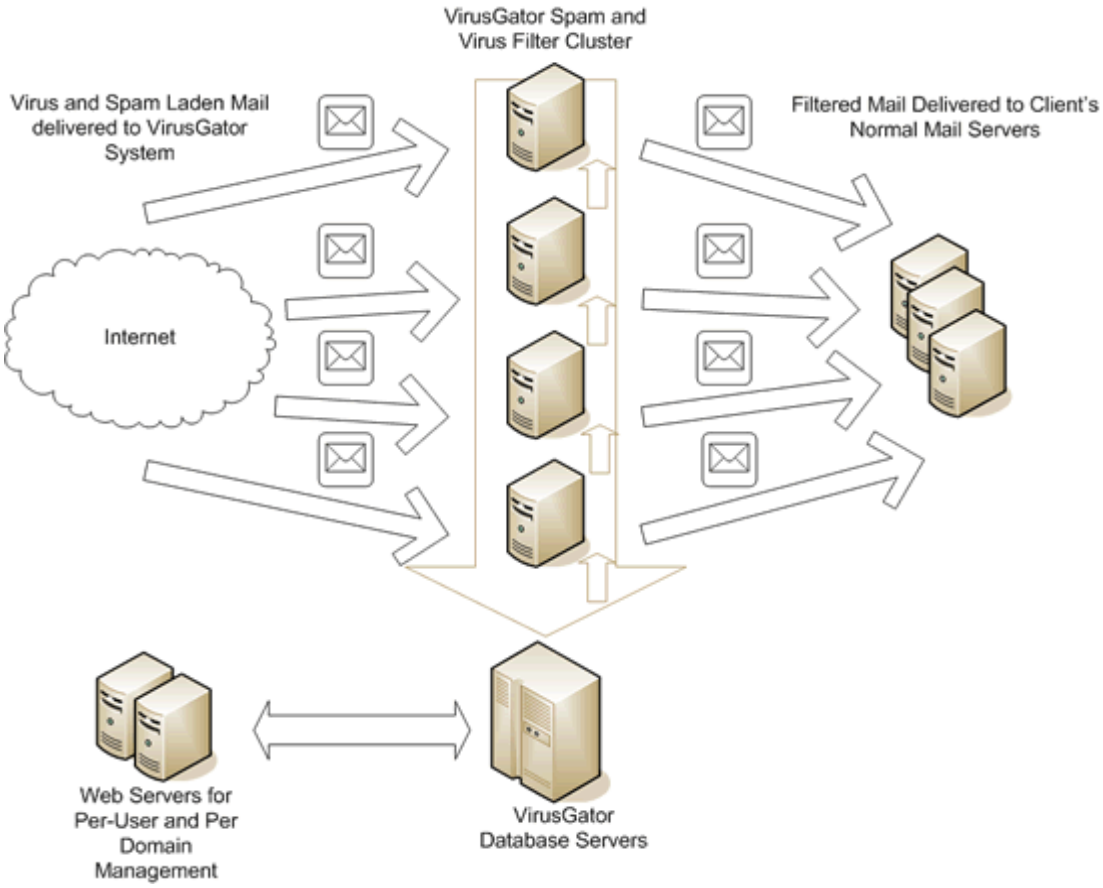


Fig. 2.4 Spam Filtering Server Cluster diagram

Once this is done, chomp.virusgator.com is the valid mail server for this domain and all mail for xyz.com should go to this logical server. Once the cluster servers have completed their work, mail is delivered to your normal mail server via SMTP.

Important: The above is a sample configuration. Our support staff will inform you regarding the host names for your new MX record. Do not use the above sample settings as they will not work.

Spam Filtering Logic. The precise spam filtering methodology used and logic involved in identifying viruses and spam is very complicated and would require many pages of illustrations and explanations to describe in detail. Below you will find a simplified diagram of the basic logic used in identifying spam and this logic is the basis used by the spam filters in VirusGator.

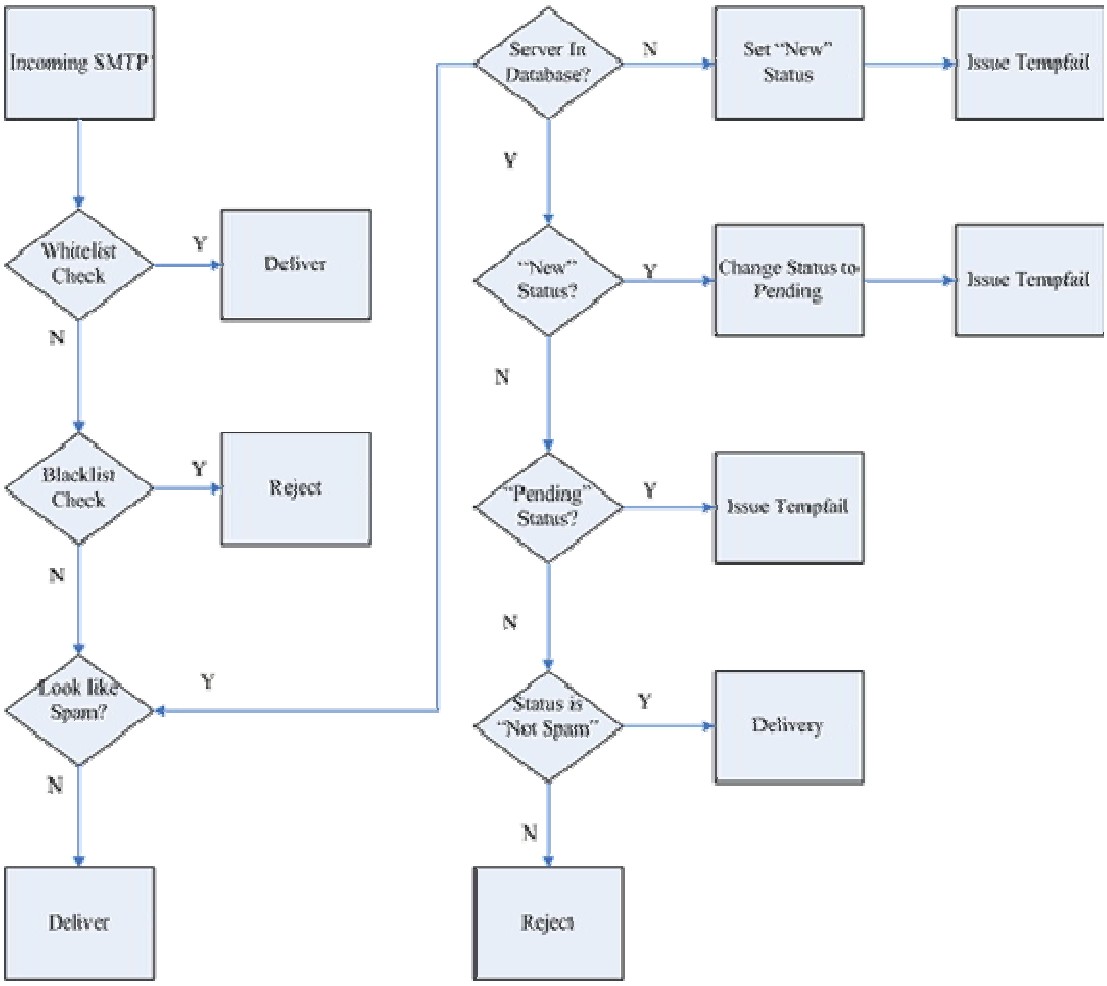


Fig. 2.5 Spam Filtering Logic Diagram

Hiding contact information. For some e-mail users, a reasonable, sufficient, and very simple approach to avoiding spam is simply to guard e-mail addresses closely. For these people, an e-mail address is something to be revealed only to selected, trusted parties. As extra precautions, an e-mail address can be chosen to avoid easily guessed names and dictionary words, and addresses can be disguised when posting to public areas. We have all seen e-mail addresses cutely encoded in forms like "<mertzHIDDEN@NOSPAM.gnosis.cx>" or "echo zregm@tabfvf.pk | tr A-Za-z N-ZA-Mn-za-m".

In addition to hiding addresses, a secretive e-mailer often uses one or more of the free e-mail services for "throwaway" addresses. If you need to transact e-mail with a semi-trusted party, a temporary address can be used for a few days, then abandoned along with any spam it might thereafter accumulate. The *real* "confidantes only" address is kept protected.

Looking at filtering software. This article looks at filtering software from a particular perspective. I want to know how well different approaches work in correctly identifying spam as spam and desirable messages as legitimate. For purposes of answering this question, I am not particularly interested in the details of configuring filter applications to work with various Mail Transfer Agents (MTAs). There is certainly a great deal of arcana surrounding the best configuration of MTAs such as Sendmail, QMail, Procmail, Fetchmail, and others. Further, many e-mail clients have their own filtering options and plug-in APIs. Fortunately, most of the filters I look at come with pretty good documentation covering how to configure them with various MTAs.

For purposes of my testing, I developed two collections of messages: spam and legitimate. Both collections were taken from mail I actually received in the last couple of months, but I added a significant subset of messages up to several years old to broaden the test. I cannot know exactly what will be contained in next month's e-mails, but the past provides the best clue to what the future holds. That sounds cryptic, but all I mean is that I do not want to limit the patterns to a few words, phrases, regular expressions, etc. that might characterize the very latest e-mails but fail to generalize to the two types.

In addition to the collections of e-mail, I developed training message sets for those tools that "learn" about spam and non-spam messages. The training sets are both larger and partially disjoint from the testing collections. The testing collections consist of slightly fewer than 2000 spam messages, and about the same number of good messages. The training sets are about twice as large.

A general comment on testing is worth emphasizing. False negatives in spam filters just mean that some unwanted messages make it to your inbox. Not a good thing, but not horrible in itself. False positives are cases where legitimate messages are misidentified as spam. This can potentially be *very* bad, as some legitimate messages are important, even urgent, in nature, and even those that are merely conversational are ones we do not want to lose. Most filtering software allows you to save rejected messages in temporary folders pending review -- but if you need to review a folder full of spam, the usefulness of the software is thereby reduced.

1. Basic structured text filters. The e-mail client I use has the capability to sort incoming e-mail based on simple strings found in specific header fields, the header in general, and/or in the body. Its capability is very simple and does not even include regular expression matching. Almost all e-mail clients have this much filtering capability.

Over the last few months, I have developed a fairly small number of text filters. These few simple filters correctly catch about 80% of the spam I receive. Unfortunately, they also have a relatively high false positive rate -- enough that I need to manually examine *some* of the spam folders from time to time. (I sort probable spam into several different folders, and I save them all to develop message corpora.) Although exact details will differ among users, a general pattern will be useful to most readers:

- Set 1: A few people or mailing lists do funny things with their headers that get them flagged on other rules. I catch something in the header (usually the From:) and whitelist it (either to INBOX or somewhere else).
- Set 2: In no particular order, I run the following spam filters:
  - Identify a specific bad sender.
  - Look for "<>" as the From: header.
  - Look for "@<" in the header (lots of spam has this for some reason).

- Look for "Content-Type: audio". Nothing I want has this, only virii (your mileage may vary).
- Look for "euc-kr" and "ks_c_5601-1987" in the headers. I can't read that language, but for some reason I get a *huge* volume of Korean spam (of course, for an actual Korean reader, this isn't a good rule).

- Set 3: Store messages to known legitimate addresses. I have several such rules, but they all just match a literal To: field.
- Set 4: Look for messages that have a legit address in the header, but that weren't caught by the previous To: filters. I find that when I am only in the Bcc: field, it's almost always an unsolicited mailing to a list of alphabetically sequential addresses (mertz1@..., mertz37@..., etc).
- Set 5: Anything left at this point is probably spam (it probably has forged headers to avoid identification of the sender).

2. Whitelist/verification filters. A fairly aggressive technique for spam filtering is what I would call the "whitelist plus automated verification" approach. There are several tools that implement a whitelist with verification: TDMA is a popular multi-platform open source tool; ChoiceMail is a commercial tool for Windows; most others seem more preliminary. (See Resources later in this article for links.)

A whitelist filter connects to an MTA and passes mail only from explicitly approved recipients on to the inbox. Other messages generate a special challenge response to the sender. The whitelist filter's response contains some kind of unique code that identifies the original message, such as a hash or sequential ID. This challenge message contains instructions for the sender to reply in order to be added to the whitelist (the response message must contain the code generated by the whitelist filter). Almost all spam messages contain forged return address information, so the challenge usually does not even arrive anywhere; but even those spammers who provide usable return addresses are unlikely to respond to a challenge. When a legitimate sender answers a challenge, her/his address is added

to the whitelist so that any future messages from the same address are passed through automatically.

Although I have not used any of these tools more than experimentally myself, I would expect whitelist/verification filters to be very nearly 100% effective in blocking spam messages. It is conceivable that spammers will start adding challenge responses to their systems, but this could be countered by making challenges slightly more sophisticated (for example, by requiring small human modification to a code). Spammers who respond, moreover, make themselves more easily traceable for people seeking legal remedies against them.

The problem with whitelist/verification filters is the extra burden they place on legitimate senders. Inasmuch as some correspondents may fail to respond to challenges -- for any reason -- this makes for a type of false positive. In the best case, a slight extra effort is required for legitimate senders. But senders who have unreliable ISPs, picky firewalls, multiple e-mail addresses, non-native understanding of English (or whatever language the challenge is written in), or who simply overlook or cannot be bothered with challenges, may not have their legitimate messages delivered. Moreover, sometimes legitimate "correspondents" are not people at all, but automated response systems with no capability of challenge response. Whitelist/verification filters are likely to require extra efforts to deal with mailing-list signups, online purchases, Web site registrations, and other "robot correspondences".

3. Distributed adaptive blacklists. Spam is almost by definition delivered to a large number of recipients. And as a matter of practice, there is little if any customization of spam messages to individual recipients. Each recipient of a spam, however, in the absence of prior filtering, must press his own "Delete" button to get rid of the message. Distributed blacklist filters let one user's Delete button warn millions of other users as to the spamminess of the message.

Tools such as Razor and Pyzor operate around servers that store digests of known spams. When a message is received by an MTA, a distributed blacklist

filter is called to determine whether the message is a known spam. These tools use clever statistical techniques for creating digests, so that spams with minor or automated mutations (or just different headers resulting from transport routes) do not prevent recognition of message identity. In addition, maintainers of distributed blacklist servers frequently create "honey-pot" addresses specifically for the purpose of attracting spam (but never for any legitimate correspondences). In my testing, I found *zero* false positive spam categorizations by Pyzor. I would not expect any to occur using other similar tools, such as Razor.

There is some common sense to this. Even those ill-intentioned enough to taint legitimate messages would not have samples of *my* good messages to report to the servers -- it is generally only the spam messages that are widely distributed. It is *conceivable* that a widely sent, but legitimate message such as the developer works newsletter could be misreported, but the maintainers of distributed blacklist servers would almost certainly detect this and quickly correct such problems.

As the summary table below shows, however, false negatives are far more common using distributed blacklists than with any of the other techniques I tested. The authors of Pyzor recommend using the tool in *conjunction* with other techniques rather than as a single line of defense. While this seems reasonable, it is not clear that such combined filtering will actually produce many more spam identifications than the other techniques by themselves.

In addition, since distributed blacklists require talking to a server to perform verification, Pyzor performed far more slowly against my test corpora than did any other techniques. For testing a trickle of messages, this is no big deal, but for a high-volume ISP, it could be a problem. I also found that I experienced a couple of network timeouts for each thousand queries, so my results have a handful of "errors" in place of "spam" or "good" identifications.

4. Rule-based rankings. The most popular tool for rule-based spam filtering, by a good margin, is *Spam Assassin*. There are other tools, but they are not as widely used or actively maintained. Spam Assassin (and similar tools)

evaluate a large number of patterns -- mostly regular expressions -- against a candidate message. Some matched patterns add to a message score, while others subtract from it. If a message's score exceeds a certain threshold, it is filtered as spam; otherwise it is considered legitimate.

Some ranking rules are fairly constant over time -- forged headers and auto-executing JavaScript, for example, almost timelessly mark spam. Other rules need to be updated as the products and scams advanced by spammers evolve. Herbal Viagra and heirs of African dictators might be the rage today, but tomorrow they might be edged out by some brand new snake-oil drug or pornographic theme. As spam evolves, Spam Assassin must evolve to keep up with it.

Spam Assassin runs much quicker than distributed blacklists, which need to query network servers. But it also runs much slower than even non-optimized versions of the below statistical models (written in interpreted Python using naive data structures).

5. Bayesian word distribution filters. Paul Graham wrote a provocative essay in August 2002. In "A Plan for Spam", Graham suggested building Bayesian probability models of spam and non-spam words. Graham's essay, or any general text on statistics and probability, can provide more mathematical background.

The general idea is that some words occur more frequently in known spam, and other words occur more frequently in legitimate messages. Using well-known mathematics, it is possible to generate a "spam-indicative probability" for each word. Another simple mathematical formula can be used to determine the overall "spam probability" of a novel message based on the collection of words it contains.

Graham's idea has several noteworthy benefits:

1. It can generate a filter automatically from corpora of categorized messages rather than requiring human effort in rule development.
2. It can be customized to individual users' characteristic spam and legitimate messages.
3. It can be implemented in a very small number of lines of code.

4. It works surprisingly well.

At first blush, it would be reasonable to suppose that a set of hand-tuned and laboriously developed rules like those in Spam Assassin would predict spam more accurately than a scattershot automated approach. It turns out that this supposition is dead wrong. A statistical model basically just works better than a rule-based approach. As a side benefit, a Graham-style Bayesian filter is also simpler and faster than Spam Assassin.

Within days -- perhaps hours -- of Graham's article being published, many people simultaneously started working on implementing the system. For purposes of my testing, I used a Python implementation created by a correspondent of mine named John Barham. I thank him for providing his implementation. However, the mathematics are simple enough that every other implementation is largely equivalent.

There are some issues of data structures and storage techniques that will effect operating speed of different tools. But the actual predictive accuracy depends on very few factors -- the main significant factor is probably the word-lexing technique used, and this matters mostly for eliminating spurious random strings. Barham's implementation simply looks for relatively short, disjoint sequences of characters in a small set (alphanumeric plus a few others).

6. Bayesian trigram filters. Bayesian techniques built on a word model work rather well. One disadvantage of the word model is that the number of "words" in e-mail is virtually unbounded. This fact may be counterintuitive -- it seems reasonable to suppose that you would reach an asymptote once almost all the English words had been included. From my prior research into full text indexing, I know that this is simply not true; the number of "word-like" character sequences possible is nearly unlimited, and new text keeps producing new sequences. This fact is particularly true of e-mails, which contain random strings in Message-IDs, content separators, UU and base64 encodings, and so on. There are

various ways to throw out words from the model (the easiest is just to discard the sufficiently infrequent ones).

I decided to look into how well a much more starkly limited model space would work for a Bayesian spam filter. Specifically, I decided to use trigrams for my probability model rather than "words". This idea was not invented whole cloth, of course; there is a variety of research into language recognition/differentiation, cryptographic unicity distances of English, pattern frequencies, and related areas, that strongly suggest trigrams are a good unit.

There were several decisions I made along the way. The biggest choice was deciding what a trigram is. While this is somewhat simpler than identifying a "word", the completely naive approach of looking at every (overlapping) sequence of three bytes is non-optimal. In particular, considering high-bit characters -- although occurring relatively frequently in multi-byte character sets (in other words, CJK) -- forces a much bigger trigram space on us than does looking only at the ASCII range. Limiting the trigram space even further than to low-bit characters produces a smaller space, but not better overall results.

For my trigram analysis, I utilized only the most highly differentiating trigrams as message categorizers. But I arrived at the chosen numbers of "spam" and "good" trigrams only by trial and error. I also picked the cutoff probability for spam rather arbitrarily: I made an interesting discovery that no message in the "good" corpus was assigned a spam probability above .0071 other than two false positives in the .99 range. Lowering my cutoff from an initial 0.9 to 0.1, however, allowed me to catch a few more message in the "spam" corpus. For purposes of speed, I select no more than 100 "interesting" trigrams from each candidate message -- changing that 100 to something else can produce slight variations in the results (but not in an obvious direction)[20].

III. Producing protecting system of Web Servers from internal and external effects

3.1 Implementation of the "Web Secure" protecting system.

My web secure protocol which I want to present is called "Web Secure". "Web Secure" CMS protecting system is flexible, powerful and intuitive system with minimum requirements for hosting, high levels of protection and is an excellent choice for the construction site of any complexity - from the normal home page with a hundred visits a day to a corporate portal with an integrated shop and a forum capable of taking Tens of thousands of visitors per day.

Fig. 3.1 Logotype of "Web Secure" protecting CMS

Administrator Panel. Administrators, Blocks and Banners, Categories, Comments, Database, Editor, more fields, groups, languages, Communications Module, ratings, RSS feeds, safety, configuration, File Editor, Community.

System Modules. Feedback, Content, File Catalog, News, Polls, Private Front Users, Backupdatabase, RSS informer, Recommend this site, Search, Editor, TinyMCE 3, SpawEditor 2, Editor, FCKeditor, top users.

Its purpose is to making web – sites and detecting them from some attacks such as spams, sql attacks and etc.

Minimum requirements for the proper functioning of the system are installed on your hosting or server programs: PHP 4.3 or higher, MySQL 4 or higher. Below there is given information to install and using this system.

1. Create a database on your hosting or server with codiering: utf8_general_ci
2. Set 666 CHMOD rights to the files config/config.php and config/config_global.php;

3. For the purposes of security, you can use the name of a standard file admin.php change;

4. Calling in the address bar of your browser: **Ошибка! Недопустимый объект гиперссылки.**, afterthat, on the browser you can see like this page and you can go on installing CMS.

Fig. 3.2 Installing time of the CMS

5. After installing the system, delete the directory setup/ and the file setup.php

After installing CMS there is shown administrator panel of the site, for example it is shown below:

Fig. 3.3 Administrator panel of the CMS

In this panel, you can work with all global functions, for example using “Edit Admins”, you can add, modify and delete extra administrators, moderators and others, also, you can manage all admins.

Using “Blocks” panel, you can manage all blocks for the site.

Categories panel is used to make categories for the site.

Using “Comments” panel, you can check all comments which are written on the site.

Using Database panel you can work with database for all time.

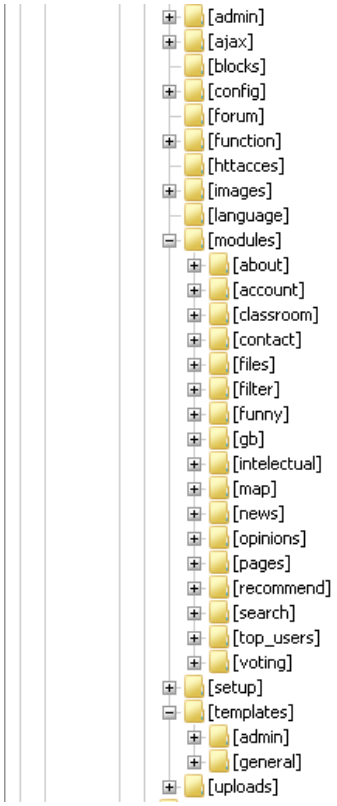
Using “AntiSpam filter” you can setup filter and check all information.

Using “Languages” panel you can add many kind of languages, and typical situation, there three languages such as English, Uzbek, Russian.

Using Security and References panel you can setup security modules and check status all time.

Besides that, in administrator panel, you can take all information about users of site and at the same time you can know any user, where, when, what did he/she or what is he/she doing?

Now, there is appear one question, how to make new site using this CMS. There is given main host structure of CMS.



According to main host structure, first folder which is called “admin” is only for administrators of the site.

All needed javascript files is saved in second folder “Ajax”.

Block files is located in “Blocks” folder.

All configuration files is saved in “Config” folder.

Important functions of the site, is situated in “function” folder

All images is saved in folder “image”

And all module files of the site is located in “modules” folder.

Fig. 3.4 Host of the site

If user wants to change site design, he should enter “templates” folder, in this folder you can see below following files:

It is important to mention that, to change design of site, user have to know base of HTML and PHP. If user knows these requirements to make site using my CMS will be very easy.

There, to make new design index.html is very essential file than others. That’s why it is suggested to edit this file, firstly.

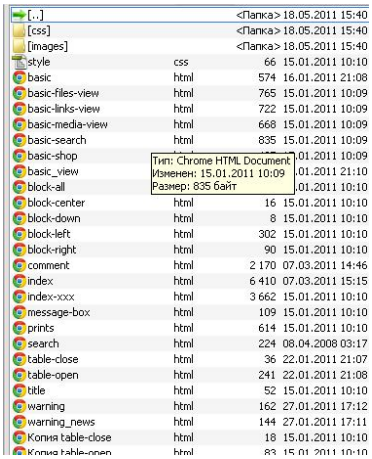


Fig. 3.5 Template part of the site

After changing site design, as example design will be show like this:

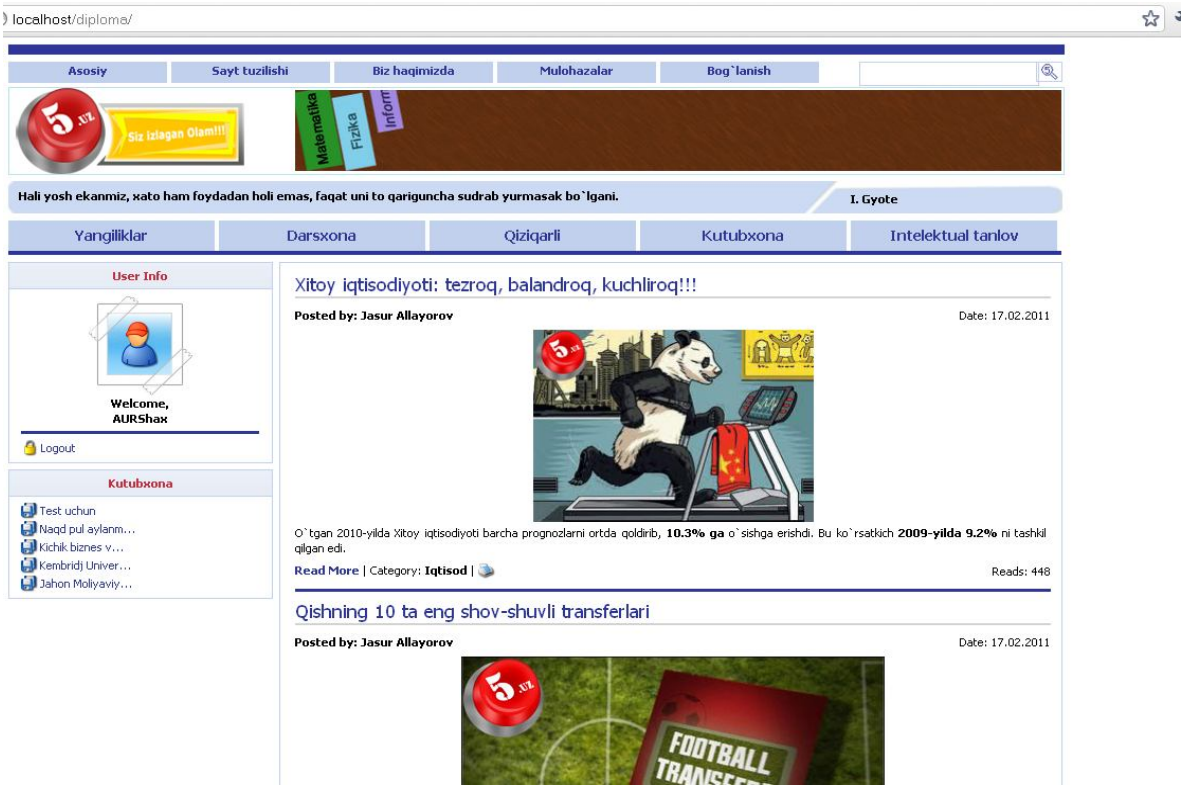


Fig. 3.6 New site which was built using the CMS

3.2 Typical features of "Web Secure" protecting system.

It is essential to say that, "Web Secure" system is built using PHP, MYSQL and a bit JavaScript. In this system, many kinds of issues were solved especially on the field of security. Some of the basic features available in most content management systems include

1. **Content Creation Capability:** The core feature of a content management system is the ability for users to create and manage content. Most content management systems provide this capability through an editing interface also known as a content input form (CIF). For enterprise class content management systems these CIF are configurable and would contain one or many form fields including fields for capturing content metadata such as authors name, date, and embedded editing interface for editors to create and edit content.

2. Content Acquisition/Consumption: This can be defined as the ability to consume structured and unstructured content from third party sources. For example ability to directly consume Microsoft Word documents, PDF documents etc.

3. **Content Transformation:** is typically referred to as transformation of XML, but can also mean the ability for on demand PDF conversions (on the fly), dynamic content assembly such as content from a database and file system assembled to provide a single unified view to the end-user etc., or in the world of digital asset management could mean resampling of image and audio files to transcoding of video to multiple formats.

4. **Library Services:** this includes capability such as check-in and check-out of content to ensure there are no conflicts during content editing, and multi-stage content approvals.

5. **Versions:** ability to store multiple versions of the same content object with capability to roll back to previous versions as and when required. Many CMS solutions also provide capabilities for visual comparison of versions.

6. **Workflow:** is the process used to manage content through its life cycle from content creation, authoring to content reviews (example editorial review, technical review, and legal review) through to publishing and ensuring the content is retired on its expiration date.

7. **Audit Trails:** ability to log any changes made to a content item such as edits, deletes, versions changes etc.

8. **Notifications:** these are triggers sent out based on predefined events such as content edits, or for notifying any workflow tasks.

9. **Search:** ability for users to search based on various attributes such as content title, metadata, author, publish date, content body etc.

10. **Metadata Management:** ability to appropriately tag a content item such that it can be easily classified and categorized. Extensible metadata management ensures easier content storage and retrieval.

11. **Taxonomy/Categorization:** ability to categorize content based on predefined attributes. There are several types of taxonomies but the typical ones are the hierarchal organization of content based on a parent child relationship. Some search engines too have capabilities to dynamically generate taxonomies based on occurrences of frequently used tags and keywords.

12. **Publishing:** is the process of delivering the content to the right end-point (or website) such that users may be able to view the content. The end-points in this case are also known as Content Delivery Applications (CDA). This is one of the core features of a content management system. High-end or enterprise class content management systems are capable of delivering content to multiple end-points/sites, while in the lower end ones the Content Management and Delivery happens in the same application. There are several checks and controls required when publishing involves physical delivery of content from a CDA to CMA. This includes capabilities such as guaranteed content delivery, automatic error handling, ability to configure publishing queues etc.

13. **Archival:** this is the ability to store the content after its expiration date to an archival database prior to its disposition.

14. **Disposition:** is the process of removing content objects from a datastore at on a predefined schedule after expiration. Legal/Compliance needs may dictate the need for an objective evidence of content disposition.

15. **Analytics:** there are several type of analytics such as content operational metrics to web analytics. Operational Metrics show the efficiencies of the content management process such as the cycle time from content creation to publishing clearly identifying bottlenecks in the process. Web Analytics shows statistics such as content usage, most used content; least used content, general traffic patterns, page views, number of hits etc. This is usually done with a web-analytics tool, and some of the high end tools have capabilities to narrow down collection of data for individual page elements/components, and may be used for marketing purposes such as A/B tests and multivariate testing etc.

16. **Security:** Content security has several connotations, but at a minimum it means the ability to ensure that only user with sufficient privileges are allowed to see content items that they are entitled to. Typically content management systems filter the content based on user privileges that are stored to database or LDAP repository.

So, below, I am going to explain especially security part of this system with the following examples:

For instance, in the function folder there given all functions. First of all , i am going to give example about authentification service of the site:

```
if ($confsp['login'] != "" && $confsp['password'] != "") {
        if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW'])) authenticate();
        if (!(($_SERVER['PHP_AUTH_USER'] == $confsp['login']) && ($_SERVER['PHP_AUTH_PW'] == $confsp['password']))) authenticate();
}

function authenticate() {
        global $confsp;
        Header('WWW-Authenticate: Basic realm="AURSlaed"');
        Header('HTTP/1.0 401 Unauthorized');
        get_exit($confsp['pass_msg'], 0);
}
```

These are authentification functions. There using php $_SERVER standart variables and Headers.

Secondly, there is given detecting admin of the site:

```
function is_admin_god() {
	global $prefix, $db, $conf;
	static $godtrue;
	$admin = (isset($_COOKIE[$conf['admin_c']])) ? explode(":", addslashes(base64_decode($_COOKIE[$conf['admin_c']]))) : false;
	if (isset($admin)) {
		if (!isset($godtrue)) {
			$id = intval(substr($admin[0], 0, 11));
			$name = htmlspecialchars(substr($admin[1], 0, 25));
			$pwd = htmlspecialchars(substr($admin[2], 0, 40));
			$ip = getip();
			if ($id && $name && $pwd && $ip) {
				list($aname, $apwd, $aip) = $db->sql_fetchrow($db->sql_query("SELECT name, pwd, ip FROM ".$prefix."_admins WHERE id='$id' AND super='1'"));
				if ($aname == $name && $aname != "" && $apwd == $pwd && $apwd != "" && $aip == $ip && $aip != "") {
					$godtrue = 1;
					return $godtrue;
				}
			}
			$godtrue = 0;
			return $godtrue;
		} else {
			return $godtrue;
		}
	} else {
		return 0;
	}
}
```

There this function check admin's login and password and know who is he. If this user is Admin, this function returns "true", but another situation returns 'false'.

Next function is about #Hack Report

Hack report

```
function hack_report($msg) {
	global $conf, $confu, $confs;
	$msg = htmlspecialchars(trim(substr($msg, 0, 500)), ENT_QUOTES);
	$url = htmlspecialchars(trim(getenv("REQUEST_URI")), ENT_QUOTES);
	$ip = getip();
	$agent = getagent();
	$date_time = date("d.m.y - H:i:s");
	if (isset($_COOKIE[$conf['user_c']])) {
```

```
                $user = $_COOKIE[$conf['user_c']];
                $user = explode(":", addslashes(base64_decode($user)));
                $user = substr(".$user[1].", 0, 25);
                $user_block = ".$user.",";
        } else {
                $user = substr($confu['anonym'], 0, 25);
        }
        if ($confs['block']) {
                $btime = time() + 86400;
                $content = file_get_contents("config/config_security.php");
                $content = str_replace("\$confs['blocker_ip'] = \".$confs['blocker_ip']."\";",
"\$confs['blocker_ip'] = \".$confs['blocker_ip']."|".$ip."|4|".md5($agent)."|".$btime."|"._HACK."||\";", $content);
                $fp = @fopen("config/config_security.php", "wb");
                fwrite($fp, $content);
                fclose($fp);
                setcookie($confs['blocker_cookie'], "block", $btime);
        }
        if ($confs['mail']) {
                $subject = ".$conf['sitename']." - "._SECURITY.";
                $msg = ".$conf['sitename']." - "._SECURITY."\n\n"
                ."._HACK.":      ".$msg."\n"._IP.":      ".$ip."\n"._USER.":      ".$user."\n"._URL.":
".$url."\n"._BROWSER.": ".$agent."\n"._STARTDATE.": ".$date_time."\n";
                $mheader = "MIME-Version: 1.0\n"
                ."Content-Type: text/plain; charset="._CHARSET."\n"
                ."Content-Transfer-Encoding: 8bit\n"
                ."Reply-To: \".$conf['adminmail']."\" <".$conf['adminmail'].">\n"
                ."From: \".$conf['adminmail']."\" <".$conf['adminmail'].">\n"
                ."Return-Path: <".$conf['adminmail'].">\n"
                ."X-Priority: 1\n"
                ."X-Mailer: Open SLAED Mailer\n";
                mail($conf['adminmail'], $subject, $msg, $mheader);
        }
        if ($confs['write_h']) {
                if ($fhandle = @fopen("config/logs/hack.txt", "a")) {
                        if (filesize("config/logs/hack.txt") > 1048576) unlink("config/logs/hack.txt");
                        fwrite($fhandle,      "._HACK.":      ".$msg."\n"._IP.":      ".$ip."\n"._USER.":
".$user."\n"._URL.": ".$url."\n"._BROWSER.": ".$agent."\n"._STARTDATE.": ".$date_time."\n---\n");
                        fclose($fhandle);
                }
        }
        setcookie($conf['user_c'], false);
        get_exit("._HACK."!", 1);
}
```

This function all time is checking all queries and if one's is hack, this function detects this query and IP address, then this IP will be blocked by this function.

Next function is master to get IP addresses:

```
function getip() {
        if (getenv("REMOTE_ADDR") && strcasecmp(getenv("REMOTE_ADDR"), "unknown")) {
                $ip = getenv("REMOTE_ADDR");
        } elseif (!empty($_SERVER['REMOTE_ADDR']) && strcasecmp($_SERVER['REMOTE_ADDR'],
"unknown")) {
                $ip = $_SERVER['REMOTE_ADDR'];
        } else {
```

```
                $ip = "unknown";
        }
        return $ip;
}
```

Below function does to detect spams:

```
function anti_spam($mail) {
        preg_match("/^(.*?)(@)(.*?)$/", $mail, $info);
        $content = "<script type=\"text/javascript\">
        String.prototype.AddMail = function (prefix, postfix) {
                hamper = prefix+\"@\"+postfix;
                document.write((hamper).link(\"mailto:\"+hamper));
        }
        </script>";
        $content .= "<script type=\"text/javascript\">\"mysi\".AddMail('".$info[1]."', '".$info[3]."');</script>"
        ."<noscript>".$info[1]."<!-- slaed --><span>&#64;</span><!-- slaed -->".$info[3]."</noscript>";
        return $content;
}
```

Next function's task is about to detect site users:

```
# Check user
function is_user($usr="") {
        global $prefix, $db, $user, $confu;
        static $usertrue;
        if (!isset($usertrue)) {
                $uid = intval(substr($user[0], 0, 11));
                $una = htmlspecialchars(substr($user[1], 0, 25));
                $pwd = htmlspecialchars(substr($user[2], 0, 40));
                $ip = getip();
                if ($uid != "" && $pwd != "") {
                        if ($confu['check'] == "0") {
                                list($pass) = $db->sql_fetchrow($db->sql_query("SELECT user_password
FROM ".$prefix."_users WHERE user_id='$uid' AND user_name='$una'"));
                                if ($pass == $pwd && $pass != "") {
                                        $usertrue = 1;
                                        return 1;
                                }
                        } else {
                                list($pass, $last_ip) = $db->sql_fetchrow($db->sql_query("SELECT
user_password, user_last_ip FROM ".$prefix."_users WHERE user_id='$uid' AND user_name='$una'"));
                                if ($pass == $pwd && $pass != "" && $last_ip == $ip && $last_ip != "") {
                                        $usertrue = 1;
                                        return 1;
                                }
                        }
                }
                $usertrue = 0;
                return 0;
        }
        if ($usertrue == 1) {
                return 1;
        } else {
                return 0;
        }
}
```

This part can detect user's orders. Next part is called URL filter:

```
# URL filter
function url_filter($url) {
	$url = strtolower($url);
	$url = (preg_match("#http\:\/\/#i", $url)) ? $url : "http://".$url."";
	$url = ($url == "http://") ? "" : text_filter($url);
	return $url;
}
```

Below, there is shown Email checking function:

```
# Mail check
function checkemail($mail) {
	global $stop;
	$mail = strtolower(text_filter($mail, 1));
	if ((!$mail) || ($mail=="") || (!preg_match("/^[_\.0-9a-z-]+@([0-9a-z][0-9a-z-]+\.)+[a-z]{2,6}$/",
$mail))) $stop = "".ERROR1."<br>"._ERROR2." (<b>email@domain.com</b>)";
	if ((strlen($mail) >= 4) && (substr($mail, 0, 4) == "www.")) $stop = "".ERROR1."<br>"._ERROR3."
(<b>www.</b>)";
	if (strrpos($mail, " ") > 0) $stop = "".ERROR1."<br>"._ERROR4.".";
	return $stop;
}
```

Next function works with Captcha, it can random word to put captcha images:

```
# Format captcha random
function captcha_random($id="") {
	global $conf;
	if ((extension_loaded("gd") && $id == 2) || (extension_loaded("gd") && !is_user())) {
		$content = "<div class=\"left\">"._SECURITYCODE.":</div><div class=\"center\"><img
src=\"index.php?captcha=1\" border=\"1\" title=\""._SECURITYCODE."\"
alt=\""._SECURITYCODE."\"></div>"
		."<div class=\"left\">"._TYPESECCODE.":</div><div class=\"center\"><input type=\"text\"
name=\"check\" size=\"10\" maxlength=\"6\" style=\"width: 75px;\" class=\"".$conf['style']."\"></div>";
		return $content;
	}
}
```

Next function also works with Captcha, it can check words which is obtained from captcha images:

```
# Format captcha check
function captcha_check($id="") {
	global $conf;
	if (($id == 2) || ($id == 1 && !is_user()) || ($_POST['posttype'] == "save" && !is_user())) {
		$code = substr(hexdec(md5(".date("F j").".$_SESSION['captcha'].".$conf['sitekey']."")), 2, 6);
		unset($_SESSION['captcha']);
		if (extension_loaded("gd") && $code != intval($_POST['check'])) {
			return 1;
		} else {
			return 0;
		}
	} else {
		return 0;
	}
}
```

Below, following function can write random words on the captcha images

```
# Format image key for captcha
switch(isset($_GET['captcha'])) {
        case "1":
        unset($_SESSION['captcha']);
        $random = gen_pass(10);
        $_SESSION['captcha'] = $random;
        $code = substr(hexdec(md5("".date("F j").""."$random.""."$conf['sitekey']."")), 2, 6);
        Header("Content-type: image/jpeg");
        $image = imagecreatefromjpeg(img_find("misc/code_bg"));
        $color = imagecolorallocate($image, 100, 100, 100);
        imagettftext($image, 14, rand(-3, 3), rand(5, 15), 18, $color, "config/font/".$conf['font'].".ttf", $code);
        imagejpeg($image, "", $conf['quality']);
        imagedestroy($image);
        exit;
        break;
}
```

This function is called filter and it checks all entered words to the site:

```
function filter($text, $no_acc_words)
{
        for($i = 0; $i <= sizeof($no_acc_words); $i++)
        {
         if(@$no_acc_words[$i] != '')
                {
                 $_pos=strpos(strtolower($text), $no_acc_words[$i]);
                 if($_pos !== false){ return 0; }
                 }
        }
}
```

Functions which are mentioned above are main parts of site security.

IV. SAFETY OF VITAL ACTIVITIES.

4.1 Establishing work places in Computer rooms.

The best computer chairs are those which take care of our body posture and prevent the development of back pain. There are many elements/factors which contribute to making a chair, perfect for comfortable seating. Computer chairs should allow proper blood circulation in the body. This is because, one has to sit on these chairs for a long time; proper blood flow is required for maintaining the health of our body. A computer chair shouldn't make us change the seating position now and then.

In today's office environment, involving sitting at the computer for long hours, ergonomics has become a necessity, especially in preventing conditions like carpal tunnel syndrome (CTS), headaches, shoulder pain, neck pain, back pain and so on due to badly designed furniture, like office chairs, and computer accessories like keyboards and mice.

Fig. 4.1 Computer room

The expressions 'office ergonomics' and 'computer ergonomics' are usually used interchangeably, and they generally refer to ergonomic office chairs, keyboards, keyboard trays, mice, desks, stools and so on. Hence, this has made the term 'ergonomics' an integral part of office nomenclature.

We can provide establishing workplaces in computer rooms with some steps as followedby:

1. **Ergonomic Office Chairs**
2. **Ergonomic Keyboards and Mouse**
3. **Ergonomics in the workplace**
4. **Fields of energomics**

One of the most important items of office ergonomics is the chair. Obviously, since people have to sit on the office chair every day, for most of the day, it has to not only be comfortable, but also ergonomically well designed. Ergonomic office chairs help to protect your lower back by providing support to the lumbar region. Plus, they are designed in such a way as to allow you to place your feet solidly on the floor while you sit on it.

Apart from adequate back support, ergonomic chairs for the office usually come with adjustable height and five legs for stability. The seat usually has a depth of about 17-20 inches, with a slight downward slope, so that there is no loss of circulation in the thighs, the front edge rounded, and the ability to tilt it 3 degrees forward or 4 degrees backwards. The lumbar support in the backrest is usually adjustable to 5-10 inches from the seat, retaining the spine's natural 'S' curve.

While seated fully back in an ergonomic chair, the lumbar region should make contact with the backrest, with the feet resting flat on the floor. The ideal sitting posture is upright, the hips positioned at a 90 degree angle, and the arms being able to hang naturally at the sides. The armrests are generally optional and are adjustable. If they impede close access to the desk, the armrests should be removed. While sitting on the chair, the hands should be in the same line as the forearms, so that the wrists are straight when the fingers are on the keyboard's front row.

Ergonomic executive chairs are usually designed with upholstered armrests and backrests. They usually also have a comparatively higher backrest.

Ergonomic Keyboards and Mouse. A keyboard and a mouse are used for most interactions with the computer. When the keyboard and mouse are used at long stretches of time, especially with the hands placed at an awkward posture, i.e., bent to the side, down or up, it can lead to musculoskeletal disorders that are painful, like carpal tunnel syndrome, tendosynovitis, and tendonitis. Hence, nowadays these devices are available in various ergonomic designs.

Ergonomic keyboard designs that are available are rotated, curved, and split. These ergonomically designed keyboards have adjustable height as well as tilt, so that the user can try out various heights and angles of tilt to find out which position is most comfortable.

As for the mouse, the important considerations to be taken into account are the shape as well as where the mouse is placed relative to the keyboard. And these days, there are several ergonomically designed mice that are available. Some mice are equipped with programmable button so that the mouse can be used to control a few computer functions. The following are some of the features that an ergonomic mouse should have:

- The shape and size of the mouse should fit comfortably in the hand.
- You should have the ability to hold the mouse in a neutral position, i.e., your hand should not be bent at any awkward position.
- The mouse should be placed in such a way so that it can be used with your upper arm comfortably relaxed, and as near your body as possible, and without you having to reach towards the side or forwards for it.

To enable this, the numeric pad is separate in some computer keyboards, thus allowing the mouse to be located nearer the keyboard. Some keyboards also have features like a touchpad, touchpoint, or trackball, which reduces the need for using the mouse as frequently.

Ergonomics in the workplace. Outside of the discipline itself, the term 'ergonomics' is generally used to refer to physical ergonomics as it relates to the workplace (as in for example ergonomic chairs and keyboards). Ergonomics in the workplace has to do largely with the safety of employees, both long and short-term. Ergonomics can help reduce costs by improving safety. This would decrease the money paid out in workers' compensation. For example, over five million workers sustain overextension injuries per year. Through ergonomics, workplaces can be designed so that workers do not have to overextend themselves and the manufacturing industry could save billions in workers' compensation.

Workplaces may either take the reactive or proactive approach when applying ergonomics practices. Reactive ergonomics is when something needs to be fixed, and corrective action is taken. Proactive ergonomics is the process of seeking areas that could be improved and fixing the issues before they become a large problem. Problems may be fixed through equipment design, task design, or environmental design. Equipment design changes the actual, physical devices used by people. Task design changes what people do with the equipment. Environmental design changes the environment in which people work, but not the physical equipment they use.

Fields of Ergonomics

Engineering psychology. *Engineering psychology* is an interdisciplinary part of ergonomics and studies the relationships of people to machines, with the intent of improving such relationships. This may involve redesigning equipment, changing the way people use machines, or changing the location in which the work takes place. Often, the work of an engineering psychologist is described as making the relationship more "user-friendly."

Engineering psychology is an applied field of psychology concerned with psychological factors in the design and use of equipment. Human factors is broader than engineering psychology, which is focused specifically on designing systems that accommodate the information-processing capabilities of the brain.

Macroergonomics. Macroergonomics is an approach to ergonomics that emphasizes a broad system view of design, examining organizational environments, culture, history, and work goals. It deals with the physical design of tools and the environment. It is the study of the society/technology interface and their consequences for relationships, processes, and institutions. It also deals with the optimization of the designs of organizational and work systems through the consideration of personnel, technological, and environmental variables and their interactions. The goal of macroergonomics is a completely efficient work system at both the macro- and micro-ergonomic level which results in improved

productivity, and employee satisfaction, health, safety, and commitment. It analyzes the whole system, finds how each element should be placed in the system, and considers all aspects for a fully efficient system. A misplaced element in the system can lead to total failure.

History. Macroergonomics, also known as organizational design and management factors, deals with the overall design of work systems. This domain did not begin to receive recognition as a sub-discipline of ergonomics until the beginning of the 1980s. The idea and current perspective of the discipline was the work of the U.S. Human Factors Society Select Committee on the Future of Human Factors, 1980-2000. This committee was formed to analyze trends in all aspects of life and to look at how they would impact ergonomics over the following 20 years. The developments they found include:

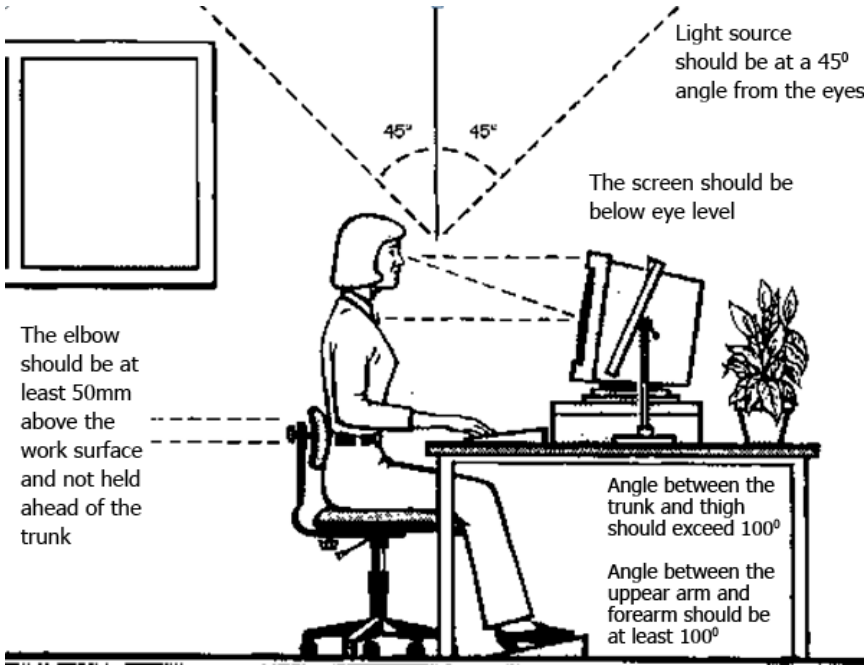


Fig. 4.2 The rules of sitting at the computer

1. Breakthroughs in technology that would change the nature of work, such as the desktop computer,

2. The need for organizations to adapt to the expectations and needs of this more mature workforce,

3. Differences between the post-World War II generation and the older generation regarding their expectations the nature of the new workplace,

4. The inability of solely microergonomics to achieve reductions in lost-time accidents and injuries and increases in productivity,

5. Increasing workplace liability litigation based on safety design deficiencies.

These predictions have become and continue to become reality. The macroergonomic intervention in the workplace has been particularly effective in establishing a work culture that promotes and sustains performance and safety improvements.

4.2 Emergency situations. Protecting from emergency circumstances and conquer with them in enterprises.

Emergency - situation in a certain area, created by the accident, the catastrophe that endangers the natural phenomena, natural or other disasters that may cause or resulted in human casualties, damage to human health or environment, considerable material losses and violation of conditions of life of people;

Protection of the population and territories from emergency situations – system of measures, methods, tools, a set of actions to prevent and emergency response;

Prevention of emergency situations - a complex of measures, carried out in advance and to the maximum possible reduce the risk of emergencies, as well as preservation of human health, reducing the size of damage to the natural environment and material losses in the event of their occurrence;

Fig. 4.3 Emergency situations

Liquidation of emergency situations - a complex emergency rescue and other urgent works carried out in the event emergencies and to save lives and preserve health, reducing the size of environmental damage and material

losses, as well as the localization of emergencies and Termination of the hazards[21].

Protection guidelines

- The basic principles - protecting the population and territories from emergencies are:
- humanism, priority of human life and health;
- publicity;
- timeliness and accuracy of the information;
- preventive measures to protect against emergencies.

The information in the field of protection from emergencies

- The information in the field of population and territories from emergencies is open, unless otherwise specified law.
- The bodies of state power and administration, bodies self-government and heads of enterprises, institutions and organizations are obliged to promptly and accurately inform the public through the media and other channels on the state protection population and territories from emergency situations and measures taken to ensure their safety, and appeared on the projected emergency situations, the methods and ways to protect the population against them.
- Concealment, untimely submission or submission officials of deliberately false information to the field of protection population and territories from emergency situations entails responsibility in accordance with the law.
- Procedure for providing information in the field of protection against emergency is defined by law.[22]

Classification

- technogenic emergencies
- natural
- environmental

Preamble

I. Technogenic emergencies

II. Emergency situations of natural character

III. Environmental emergencies

IV. Local, local, national and trans boundary emergency

Emergencies * classified for reasons (sources) of their occurrence and, depending on the number of people affected in these situations, the amount of material damage and the scope (the boundaries of zones **) emergencies are divided into local, local, national and trans boundary.

I. Emergencies Manmade.

1. Transportation accidents and disasters
2. Accidents on chemically dangerous objects
3. Accident at the fire and explosive objects
4. Accident on energy and utility systems
5. The sudden collapse of building structures
6. Accidents associated with the use or storage of radioactive and other hazardous and environmentally hazardous substances
7. Hydraulic engineering disaster and accident

1. Transportation accidents and disasters:

plane crash, resulting in the deaths of crew members and passengers, the complete destruction or severe damage to the aircraft and requiring search and rescue operations;

disaster and accident (crash) on railway transport, cause fires, explosions, destruction, rolling stock and resulted in the deaths of railway staff, passengers and people who were in the disaster area on railway platforms, stations, buildings and urban development, as well as contamination carried potent toxic substances (SPS) area adjacent to the accident site;

disasters and accidents of road transport, including road traffic accidents, accompanied by explosions, fires, destruction of vehicles, a manifestation of the corrosive properties of transported SPS and death (injuries, poisoning) of the people;

disaster, accident, fire stations and in tunnels underground, resulting in the deaths, injuries and poisoning of people, destruction of subway trains;

accident at the pipelines that caused the volley (emergency) release (bottled) gas, oil and petroleum products, fire open oil and gas fountains.

2. Accidents on chemically dangerous objects:

accidents, fires and explosions on chemically dangerous objects, accompanied volley (accidental) release into the environment of highly toxic substances and the release of damaging factors outside the sanitary protection area with a large excess of maximum permissible concentration (MPC), which may cause or has caused massive damage people, animals and plants.

3. Accident at the fire-hazardous locations

accidents, fires and explosions at facilities that use a process or having to store explosives, flammable, and other flammable substances and materials, resulting in the mechanical and thermal injuries, poisoning and death, the destruction of fixed assets, the violation of the production cycle and livelihoods within the zones of emergency situations;

accidents, fires and collapse of rock caused by the explosion of gas and dust in coal mines and the mining industry, resulting in injury, poisoning and loss of life and required to conduct search and rescue use of special equipment and respiratory protection.

4. Accident on energy and utility systems:

accidents and fires in the HPP, TPP, thermal power plants, district heating plants, power lines, boilers, compressor and gate stations and other points of supply, resulted in a power outage responsible consumers of industry and agriculture, and violation of public life;

accident on gas pipelines, water intakes, water, sewerage and other municipal facilities, resulting in a violation of life and threat to public health;

accident at the gas treatment plants, biological and other treatment facilities, which caused pollution of the atmosphere, soil, surface water and groundwater contaminants at concentrations that threaten human health.

5. The sudden collapse of building structures:

schools, hospitals, cinemas and other social facilities, as well as residential buildings, fires, gas explosions and other accidents associated with the loss of life and requiring immediate rescue operations and emergency medical assistance to victims.

6. Accidents associated swith the use or storage of radioactive and other hazardous and environmentally harmful substances:

accidents at facilities that use in the process of radioactive substances and caused their release outside the sanitary protection zone, which originated increased radioactivity may cause a threat to get people sverhdopustimyh doses; accident involving radioactive materials;

accident (break) on drives of radioactive waste, tailings, slurry tank that threaten the environment and human health;

loss of radioisotope products;

situations involving the diversion of biological agents in the environment or loss of research and other institutions engaged in the manufacture, storage and transportation of biological agents and drugs from them.

7. Hydraulic engineering disasters and accidents:

catastrophic flooding caused by failure of hydraulic structures on reservoirs, rivers and canals, a breakthrough high mountain lakes and resulted in human casualties, disruption of the functioning of industrial and agricultural projects, livelihoods of people in floodplains and requiring emergency helps.

II. Emergencies Natural Character

1. Geological hazards

2. Hydro meteorological hazards

3. Extraordinary epidemiologic, epizootic situation

1. **Geological hazards:**

earthquake which caused human casualties, destruction of varying degrees of administrative and industrial buildings, process equipment, power systems, transportation and infrastructure, buildings, social facilities and residential homes, disruption of production and livelihoods;

landslides, rock falls and other dangerous geological phenomena have caused or may cause loss of human life and requiring temporary relocation from the danger zone or relocation of people for permanent residence in the safe areas.

2. **Hydrometeorological hazards:**

floods, flash floods and mudslides, resulting in the deaths, flooding human settlements, individual industrial and agricultural facilities, destruction of infrastructure and transport communications, a violation of production and livelihoods of people and requiring emergency avalanches, severe (storm) winds, torrential rains and other dangerous hydro meteorological events that resulted or may result in injuries and deaths of residents of settlements, having a rest in sanatoriums, rest homes, camps, tourists and athletes.

3. **Extraordinary epidemiologic, epizootic situation**

especially dangerous infections caused by single plague, cholera, yellow fever; infectious disease in humans - epidemic typhus, Brill's disease, Q fever; zoonotic infections - anthrax, rabies;

viral infection - AIDS;

epidemic - infectious disease group of people that do not belong to a particularly dangerous infections, with one source of infection or transmission of the same factors, in one village - 50 persons or more;

disease of unknown etiology group - 20 people or more;

febrile illness of unknown diagnosis - 15 people or more;

A situation in which mortality or disease than the average of 3 times or more;

poisoning by toxic substances, the number of victims - 10 men, the death toll - 2 persons or more;

Mass food poisoning, the number of victims - 10 men, the death toll - 2 persons or more;

epizootic - the facts of mass illness or death of the animals;

epiphytoties - the facts of mass death of plants[23].

III. Emergencies Ecological character

1. Situations related to changes in state land (soil, subsoil);

2. Situations related to the change in the composition and properties of the atmosphere (Air);

3. Situations involving change of state of the hydrosphere;

1. **Situations related to changes in state land (Soil, subsoil):**

catastrophic subsidence - landslides, rock falls the earth's surface caused by production of mineral resources to mining and other human activities; contamination of soil and subsoil of the toxic chemicals of industrial origin, the presence of heavy metals, petroleum products, as well as pesticides and other chemicals used in agricultural production at concentrations that pose a threat to human health.

2. **Situations related to the change in the composition and properties atmosphere (air):**

1. extremely high air pollution ingredients: dioxide, sulfur dioxide and nitrogen oxide, carbon monoxide, dioxin, soot, dust and other harmful substances of human origin in concentrations that pose a threat to human health;
2. formation of vast areas and loss of large amount of acid rain;
3. elevated levels of radiation.

3. **Situations involving change of state of the hydrosphere:**

1. extremely high contamination of surface and groundwater discharges of industrial and agricultural production: oil, waste containing heavy metals, various pesticides and other harmful substances that have caused or may cause loss of people;
2. increased level of groundwater, which may cause or has caused the destruction of buildings, utilities and residential buildings;
3. drastic shortage of drinking water due to contamination by harmful substances water sources and water intakes.

IV. Locally, the local, national, and transboundary emergencies

1. Refers to a local emergency, which resulted in injured 10 people, or violated the conditions of life not more than 100 people, or damage to property amounts to no more than a thousand times the minimum wage on the day of an emergency, and the area of emergency does not go beyond Onsite production or social purpose.

2. Refers to the local emergency as a result of which affected more than 10 but not more than 500 people, or violated the conditions of life of more than 100 but not more than 500 people, or property damage of over 1 thousand but not more than 0.5 million minimum wages work on the day of an emergency, and emergency zone does not extend beyond the village, city, district or region.

3. By the Republican is an emergency situation, which resulted in more than 500 people injured or violated the conditions of life of over 500 persons, or damage to property amounts to more than 0.5 million times the minimum wage on the day of an emergency, and the zone of an emergency beyond the region.

4. Refers to cross-border emergency situation, the consequences of which are outside the country, or an emergency has occurred abroad and affects the territory of Uzbekistan.

5. In order to adequately respond to emergency situations of natural and environmental issues (heavy rain, hail, pollution of soil, subsoil and water with oil products, heavy metals, pesticides and other toxic chemicals, extreme pollution by

the harmful ingredients of the atmosphere above the MPC, etc.), quantitative indicators, which are hereby Regulations do not exist, establish ministries and departments to conduct surveillance and monitoring of the environment and the respective governing bodies of the State system of prevention and emergency response the Republic of Uzbekistan (GSCHS) depending on the specific impact on the lives, health and the natural environment.