Security in HTTP

HTTP is used for communication over the Internet, so application programmers, information providers, and users should be aware of the protection limitations in HTTP / 1.1.

HTTP is used for communication over the Internet, so application programmers, information providers, and users should be aware of the protection limitations in HTTP / 1.1. This discussion chapter will not include clear solutions to the issues mentioned here, but it does provide some suggestions for reducing risks.

The leakage of personal information

Clients often keep a large amount of personal information such as user name, location, mail address, encryption keys, . So you should be very careful to prevent leakage This information passes HTTP protocols to other sources.

All confidential information should be stored in the Server in the encrypted form.

Exploring the server's own software version may allow the Server device to become more vulnerable when attacked by software known as security vulnerabilities.

Authorized stations that serve as a gateway through a network firewall should take special precautions to the transmission of the Header information that identifies the hosts behind the firewall.

Information sent in the "From" field may conflict with the user's personal rights or the site's privacy policy, and therefore, should not be transmitted without user supervision. Use to disallow, allow or edit the contents of the school.

Clients should not include a Referer field in an HTTP (unsafe) request, if the page being directed is spread with a security protocol.

The authors of the service that use the HTTP protocol should not use GET-based patterns to accept sensitive data, because it will make the data encrypted in the Request-URI.

Attacks based on Path and File names

Documentation should be limited to documents that are returned by HTTP requests to only those documents that are intended by the Server manager.

For example, UNIX, Microsoft and other operating systems use `.` as a transmission component to indicate a directory level above the current directory. On such a system, a Server **MUST** not allow any build in the Request-URI, otherwise it will allow access to a source outside these directories to be accessible via the Server. .

DNS deception (DNS Spoofing)

Clients using HTTP are primarily based on **Domain Name Service** (DNS), and are therefore vulnerable to security attacks based on the deliberate link deletion of IP addresses and DNS names. So the Client should pay attention while assuming that the ongoing validity of a link between the IP / DNS domain name.

If the clients write to the memory hiding the results of host name lookups to achieve performance improvement, they must monitor TTl information reported by DNS. If the clients do not follow this rule, they can be fooled when a previously accessed **server** 's IP address changes.

Position Headers and deception

If a single Server supports multiple organizations without trusting each other, then it **MUST** check the values ?? of the **Location** and **Content Location** fields in the responses that are generated under the control of the prompted organizations. come to ensure that they do not attempt to take over invalid resources through which they do not have authorization.

Verification verification

Existing Clients and user agents have the characteristic of recording obscure verification information. HTTP / 1.1 does not provide a method for the Server to direct clients directly to remove cached credentials that are a major security risk.

There is some work around to parts of this problem, and so it is recommended to use password protection in screensaver, free time, and some other methods that do Reduce inherent safety issues in this regard.

Authorizations and caching

HTTP authorizations are an intermediate server, and respectively opportunities for intermediate attacks. The credentials have access to relevant confidential information, personal information about each user and organizations, and the proprietary information of the user and the content provider.

Authorized operators should tell the systems that credentials run on, because they will protect any system that contains or transmits sensitive information.

Writing to memory hides credentials that create additional vulnerabilities, since the contents of the hidden memory represent an attractive target for malicious exploitation. Therefore, hidden memory content must be protected as sensitive information.

According to Tutorialspoint

Previous post: URL encoding in HTTP

Next article: Example of Message in HTTP