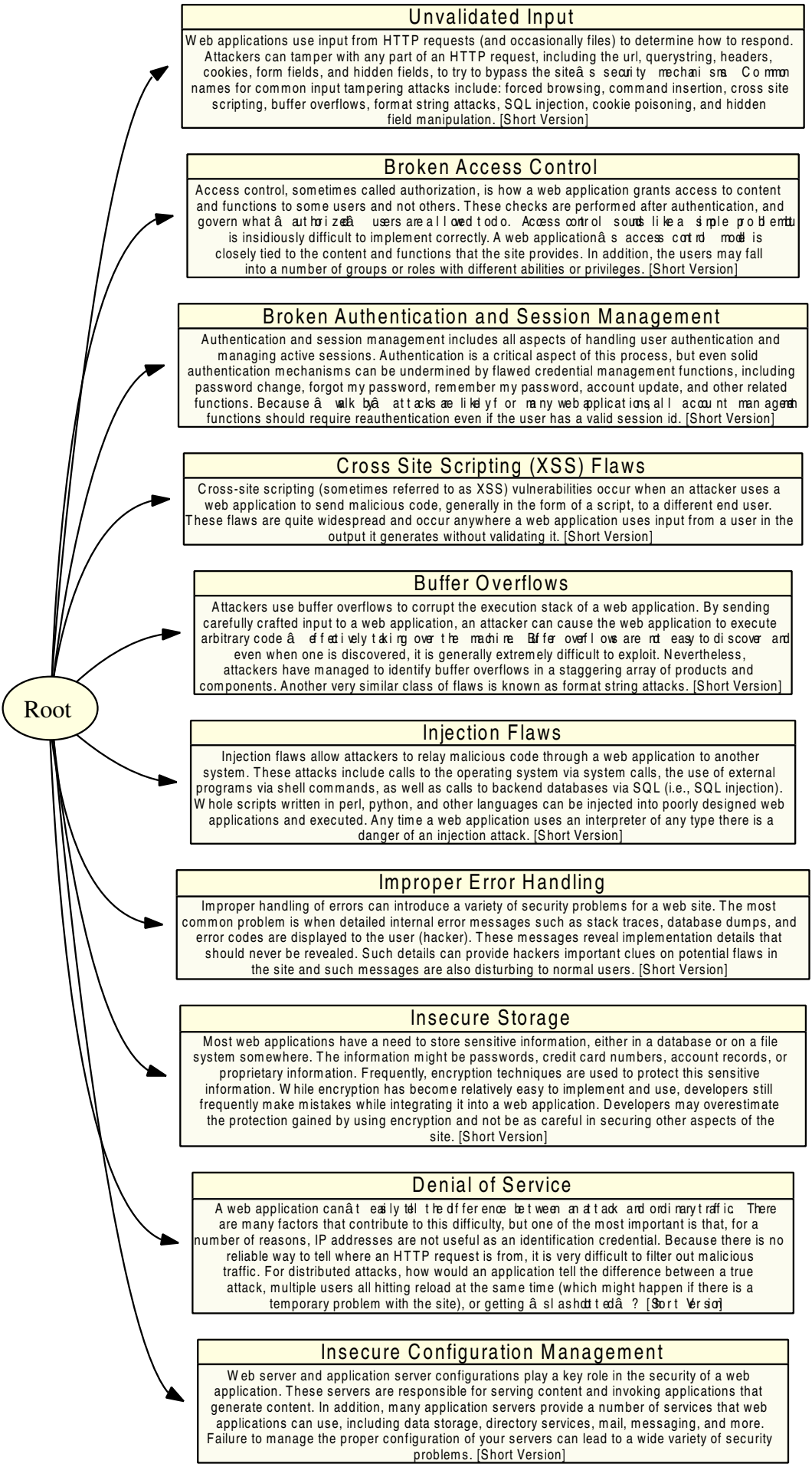
Root
Unvalidated Input
Web applications use input from HTTP requests (and occasionally files) to determine how to respond. Attackers can tamper with any part of an HTTP request, including the url, querystring, headers, cookies, form fields, and hidden fields, to try to bypass the siteâ s security mechani sms. Common names for common input tampering attacks include: forced browsing, command insertion, cross site scripting, buffer overflows, format string attacks, SQL injection, cookie poisoning, and hidden field manipulation. [Short Version]
Broken Access Control
Access control, sometimes called authorization, is how a web application grants access to content and functions to some users and not others. These checks are performed after authentication, and govern what â authorizedâ users are allowed todo. Access control sounds like a simple problembu is insidiously difficult to implement correctly. A web applicationâ s access control model is closely tied to the content and functions that the site provides. In addition, the users may fall into a number of groups or roles with different abilities or privileges. [Short Version]
Broken Authentication and Session Management
Authentication and session management includes all aspects of handling user authentication and managing active sessions. Authentication is a critical aspect of this process, but even solid authentication mechanisms can be undermined by flawed credential management functions, including password change, forgot my password, remember my password, account update, and other related functions. Because â walk byâ attacks are likelyf or many web applications, all account management functions should require reauthentication even if the user has a valid session id. [Short Version]
Cross Site Scripting (XSS) Flaws
Cross-site scripting (sometimes referred to as XSS) vulnerabilities occur when an attacker uses a web application to send malicious code, generally in the form of a script, to a different end user. These flaws are quite widespread and occur anywhere a web application uses input from a user in the output it generates without validating it. [Short Version]
Buffer Overflows
Attackers use buffer overflows to corrupt the execution stack of a web application. By sending carefully crafted input to a web application, an attacker can cause the web application to execute arbitrary code â effectively taking over the machine. Buffer overflows are not easy to discover and even when one is discovered, it is generally extremely difficult to exploit. Nevertheless, attackers have managed to identify buffer overflows in a staggering array of products and components. Another very similar class of flaws is known as format string attacks. [Short Version]
Injection Flaws
Injection flaws allow attackers to relay malicious code through a web application to another system. These attacks include calls to the operating system via system calls, the use of external programs via shell commands, as well as calls to backend databases via SQL (i.e., SQL injection). Whole scripts written in perl, python, and other languages can be injected into poorly designed web applications and executed. Any time a web application uses an interpreter of any type there is a danger of an injection attack. [Short Version]
Improper Error Handling
Improper handling of errors can introduce a variety of security problems for a web site. The most common problem is when detailed internal error messages such as stack traces, database dumps, and error codes are displayed to the user (hacker). These messages reveal implementation details that should never be revealed. Such details can provide hackers important clues on potential flaws in the site and such messages are also disturbing to normal users. [Short Version]
Insecure Storage
Most web applications have a need to store sensitive information, either in a database or on a file system somewhere. The information might be passwords, credit card numbers, account records, or proprietary information. Frequently, encryption techniques are used to protect this sensitive information. While encryption has become relatively easy to implement and use, developers still frequently make mistakes while integrating it into a web application. Developers may overestimate the protection gained by using encryption and not be as careful in securing other aspects of the site. [Short Version]
Denial of Service
A web application canâ t easily tell the difference between an attack and ordinary traffic. There are many factors that contribute to this difficulty, but one of the most important is that, for a number of reasons, IP addresses are not useful as an identification credential. Because there is no reliable way to tell where an HTTP request is from, it is very difficult to filter out malicious traffic. For distributed attacks, how would an application tell the difference between a true attack, multiple users all hitting reload at the same time (which might happen if there is a temporary problem with the site), or getting â slashdottedâ ? [Short Version]
Insecure Configuration Management
Web server and application server configurations play a key role in the security of a web application. These servers are responsible for serving content and invoking applications that generate content. In addition, many application servers provide a number of services that web applications can use, including data storage, directory services, mail, messaging, and more. Failure to manage the proper configuration of your servers can lead to a wide variety of security problems. [Short Version]