Retrieve more information from slides and lectures

**DDOS**

Tips

1. Organize a DDoS Attack Response Plan. Don’t be caught blindsided by DDoS attacks; have a response plan ready in case of a security breach so your organization can respond as promptly as possible. Your plan should document how to maintain business operations if a DDoS attack is successful, any technical competencies and expertise that will be necessary, and a systems checklist to ensure that your assets have advanced threat detection.

Additionally, establish an incident response team in case the DDoS is successful and define responsibilities, such as notifying key stakeholders and ensuring communication throughout the organization.

2. Secure your Infrastructure with DDoS Attack Prevention Solutions. Equip your network, applications, and infrastructure with multi-level protection strategies. This may include prevention management systems that combine firewalls, VPN, anti-spam, content filtering and other security layers to monitor activities and identity traffic inconsistencies that may be symptoms of DDoS attacks.

If you’re looking for protection by leveraging cloud-based solutions, many providers allow for advanced protection resources for additional charges. Other options allow for businesses to go “full cloud,” entrusting sensitive data with a reputable cloud provider that offers heightened security protocols, both virtual and physical.

3. Perform a Network Vulnerability Assessment. Identify weakness in your networks before a malicious user does. A vulnerability assessment involves identifying security exposures so you can patch up your infrastructure to be better prepared for a DDoS attack, or for any cybersecurity risks in general.

Assessments will secure your network by trying to find security vulnerabilities. This is done by taking inventory of all devices on the network, as well as their purpose, system information, and any vulnerabilities associated with them, and including what devices need to be prepared for upgrades or future assessments. Doing so will help define your organization’s level of risk so you can optimize any security investments.

4. Identify Warning Signs of a DDoS Attack. If you can identify the symptoms of a DDoS attack as early as possible, you can take action and hopefully mitigate damage. Spotty connectivity, slow performance, and intermittent web crashes are all signs that your business may be coming under attack from a DDoS criminal. Educate your team on signs of DDoS attacks so everyone can be alert for warning signs.

Not all DDoS attacks are extensive and high volume; low-volume attacks that launch for short durations are just as common. These attacks can be particularly nefarious because they are more likely to go under the radar as just a random incident rather than a potential security breach. Low-volume DDoS attacks are likely distractions for damaging malware; while your IT security staff is distracted by a low-volume attack, malicious software like ransomware can infiltrate your network.

5. Adopt Cloud-Based Service Providers. There are several benefits to outsourcing DDoS attack prevention to the cloud.

Cloud providers who offer high levels of cybersecurity, including firewalls and threat monitoring software, can help protect your assets and network from DDoS criminals. The cloud also has greater bandwidth than most private networks, so it is likely to fail if under the pressure of increased DDoS attacks.

Additionally, reputable cloud providers offer network redundancy, duplicating copies of your data, systems, and equipment so that if your service becomes corrupted or unavailable due to a DDoS attack, you can switch to secure access on backed-up versions without missing a beat.

Prevention

Enable DDoS protection on network hardware: Ensure routers, load balancers and firewalls have specific protection policies enabled and are configured for high availability and redundancy.

Enable DDoS protection on network intrusion detection/prevention systems (IDS/IPS): Enable rules that protect against DDoS and block rules for unapproved protocols often used in these attacks, such as NTP and ICMP.

Enable DDoS protection on web application firewalls (WAFs): Enable anomaly detection and any client classification rules that detect or block known malicious or redirected traffic.

Update incident response plans with your ISPs: Ensure protections are in place with your ISPs and build a playbook for detection and response that includes contact numbers, service-level agreements (SLAs) and escalation paths.

Deploy DDoS protection tools/appliances: These tools can offer solutions for packet scrubbing, DNS redirection, DNSSEC, geographic blocking, web application firewall (WAF)/ filtering, caching and flow monitoring. For cloud environments, consider evaluating/implementing cloud DDoS protection such as AWS Shield or Akamai.

In addition to the aforementioned DDoS prevention steps, create an incident response plan that:

* Defines the resources, tools and procedures required to minimize the risk and costs of a DDoS incident.
* Includes critical topics such as risk assessment, roles and responsibilities, mitigation strategies, monitoring, attack recovery and communications planning.
* During the planning phase, you should also make sure you understand where single-points-of-failure are located and how you could mitigate threats to them. For high-impact assets, consider employing multiple ISPs.

Detection

A DDoS attack is a complex challenge for a business to face, because often, it’s difficult to determine whether a spike in traffic is legitimate or an attack, especially if the proper tools aren’t in place.

Detection can be automatic or manual. Manual detection usually occurs when people or customers complain about slow performance or inability to access resources, but there are also network monitoring tools from Cisco and SolarWinds that can automatically detect and alert that an attack is under way. Obviously, automatic detection is better because it occurs faster than a manual process ever could.

If you suspect an attack is in process, you can also:

* Try a ping request: If a TTL (time to live) count on a ping request times out, this may indicate a problem. The most obvious symptom of a DDoS attack is a site or service suddenly becoming slow or unavailable.
* Review the logs: If accessible, log tools may show large spikes in traffic. A single or range of IP addresses might make up a very high number of requests, which could indicate an attack.
* Review system errors: Look for problematic system responses. An example would be a web server that responds with a 503 error.
* User or customer complaints: User reports of slowness is still the No. 1 way many people find out about an attack.
* Any of these signs may be indicative of an attack in progress. Once confirmed, it’s time to invoke the incident response plan. The immediate goal is always to mitigate business impact and get systems available again as soon as possible.

Eradication, Containment and Recovery

Leverage business continuity: Identify alternative courses for business or network operations. For example, you may choose to coordinate with your business continuity group to fail over to another site. Having more than one ISP is critical.

* **Set up alternative networks**: Using a separate management network allows for greater security and possibly the ability to recover because it is a type of out-of-band access that may survive a failure in the production network. Using out-of-band networks or dial-up modems that provide dial-up access to key systems that may not be reachable on the network in the event of an outage can help you respond to and manage an incident. Make sure these devices or management networks are protected and have strong access control.
* **Deploy intelligent routing**: Many DDoS mitigation solutions such as Cloudflare or Akamai use intelligent routing to break traffic into manageable chunks, preventing denial-of-service.
* **Set up allow-list network access**: Create a list of priority IPs and services that must be allowed through and block everything else.
* **Coordinate with your ISP**: Don’t fail to take the very basic step of coordinating with your ISP. It is often better positioned to respond to and mitigate an attack.
* **Try data scrubbing**: With this method, the traffic destined for a particular IP address range is redirected to a data center, where the attack traffic is “scrubbed” or cleaned. Only clean traffic is then forwarded to the target destination. Although most scrubbing service providers offer strong DDoS mitigation capabilities, enterprises should evaluate the provider’s infrastructure capacity and service levels to ensure it will be sufficient. A good rule of thumb is to look for mitigation networks with at least two or three times the capacity of the largest attacks you’ve seen against your network.
* **Buy enough network bandwidth**: The most basic way to make an organization resilient to DDoS attacks is to have ample network bandwidth for very high loads right from the beginning. However, this is not always the most efficient way of dealing with these attacks, because it requires purchasing more bandwidth than you would need on average.

Mitigation

Usually, no single strategy alone will fully mitigate a DDoS attack. It’s important to take a multi-strategy approach based on your business risk. And the more strategies you use, the more important it is to test them to ensure they will work as expected when a real attack occurs.

Tabletop and other simulated exercises can help test both tools and processes, and verify they perform as expected. It’s also critical to learn from these activities and adjust the plans, as necessary.

Defense

DDoS attacks remain an ongoing threat for many companies, but with the right preparation and incident response processes, the business impact of these attacks can be mitigated. To ensure you have the best defenses in place, make sure you:

* Plan in advance: The more that can be done in the planning phase, the more smoothly the other phases will go. Make sure you understand the potential business impact of a DDoS attack and plan your security controls accordingly.
* Use multiple DDoS mitigation strategies: Strategies including traffic allow-listing and alternate ISPs can be used together to create a comprehensive DDoS response strategy.
* Fit your strategies to your specific risk profile and business: This helps avoid going overboard on spending.

**Data Exfiltration**

Common Techniques

*Social Engineering*

Social engineering is one of the most common methods of exfiltrating data. An adversary tricks a user into sharing sensitive data or credentials by posing as a legitimate employee or partner. For example, an adversary may pose as a help desk agent to ask a user for sensitive information, such as username and password.

One common type of social engineering attack is phishing. In phishing attacks, the attacker sends users an email that appears to be from a legitimate source, such as the human resources department. The email will ask the user to click on a link, which will send victims to a false site that looks exactly like the official human resources portal. This false site may be set up exclusively to harvest credentials, or the site’s code may contain a malicious script that installs a keylogger or other malware that will then be used to execute the next stage of the phishing attack.

*Human Error*

Careless insiders commonly download sensitive company data from their secure company-issued devices to personal devices that are not protected by their employers’ network security solutions or policies. Instead, the data is either entirely unprotected or protected only by the basic level of consumer security tools. In this situation, data exfiltration may not be limited to the movement of files — it could include photos of monitor screens taken with smartphones, recordings of conversations made with smartphones, etc.

Prevention

The most significant defensive practice a business can establish is also the most difficult: educate employees. Of course, many businesses already do this with regular mandatory security awareness training, but most employees continue to underestimate the likelihood that they’ll be targeted. Businesses need to foment a culture of security throughout the organization before they can have faith in their employees to act as the first line of defense.

Bring-your-own-device (BYOD) policies should be in place and made clear to all employees. Today in particular, with the shift to remote work, employees may be using any manner of personal device to access valuable data, from a kid’s gaming system to a Windows 8 tower. Monitoring the network to see who is logging on and which devices they are using is not only necessary to prevent a data breach today, but also to understand how users are interacting with the network in order to plan better for tomorrow.

To control insider threats, both benign and malicious, control privilege. That means only granting least privilege; dynamically controlling privilege so that when an employee’s reason to access a sensitive system is no longer valid, they no longer have access; and systematically revoking privilege for former employees from the moment their employment ends, rather than waiting a week or two to clean up old accounts.

**SQL Injection**

In computing, SQL injection is a code injection technique used to attack data-driven applications, in which malicious SQL statements are inserted into an entry field for execution (e.g. to dump the database contents to the attacker).[1][2] SQL injection must exploit a security vulnerability in an application's software, for example, when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database.

Actions a successful attacker may take on a compromised target include:

* Bypassing authentication
* Exfiltrating/stealing data
* Modifying or corrupting data
* Deleting data
* Running arbitrary code
* Gaining root access to the system itself

Common Injection Techniques

There are a wide variety of SQL injection vulnerabilities, attacks, and techniques, which arise in different situations. Some common SQL injection examples include:

* Retrieving hidden data, where you can modify an SQL query to return additional results.
* Subverting application logic, where you can change a query to interfere with the application's logic.
* UNION attacks, where you can retrieve data from different database tables.
* Examining the database, where you can extract information about the version and structure of the database.
* Blind SQL injection, where the results of a query you control are not returned in the application's responses.

Prevention

This conversation’s ultimate goal is to provide us with the context needed to prevent SQL injection whenever possible. The following are tips that assist in preventing SQL injection altogether. Some of these tips also help in reducing the scope of SQL injection so that exploiting security holes is more challenging, less lucrative, or impossible.

Limit Result Sets

Put limits on all result sets, such as:

* Implement a date range limit that ensures data is returned from a narrow date/time range
* Limit row counts processed or returned. This prevents reading or returning too much data. In addition to preventing large result sets, limiting data processed can ensure good performance, regardless of parameters
* Prevent blank searches. Allowing a user to search through everything with no filters or return all possible results will likely perform poorly and provide little value. If blank search criteria make no sense logically, then do not allow them
* By guiding users towards common use cases, we can improve performance and bolster security at the same time. Allowing users freedom to do what they want sounds noble, but ultimately leads to more bugs, security holes, and exploits. A user will be happier with a limited set of quality options than a massive set of buggy ones.

Additionally, Cleanse and Validate Freeform User Input

This is one of the most important steps to preventing SQL injection. Any data that a user can provide, whether via a web form, file, API, or other application needs to be cleansed and validated. This process will check user input for invalid characters, unacceptable length, or any other abnormalities prior to processing or storing it on any production systems.

Mitigation

An SQL injection is a well known attack and easily prevented by simple measures. After an apparent SQL injection attack on TalkTalk in 2015, the BBC reported that security experts were stunned that such a large company would be vulnerable to it.[19]

Object relational mappers

Developers can use ORM frameworks such as Hibernate[20] to create database queries in a safe and developer-friendly way. Since database queries are no longer constructed as strings, there is no danger of an injection vulnerability.[21]

Web application firewalls

While WAF products such as ModSecurity CRS[22] cannot prevent SQL injection vulnerabilities from creeping into a codebase, they can make discovery and exploitation significantly more challenging to an attacker.

Parameterized statements

Main article: Prepared statement

With most development platforms, parameterized statements that work with parameters can be used (sometimes called placeholders or bind variables) instead of embedding user input in the statement. A placeholder can only store a value of the given type and not an arbitrary SQL fragment. Hence the SQL injection would simply be treated as a strange (and probably invalid) parameter value. In many cases, the SQL statement is fixed, and each parameter is a scalar, not a table. The user input is then assigned (bound) to a parameter.[23]

Easily put, using parameterized queries can definitely prevent SQL injection. This mainly means that your variables aren't query strings that would accept arbitrary SQL inputs, however, some parameters of given types are definitely necessary. Parameterized queries require the developer to define all the code. Therefore, without parameterized queries, anyone could put any kind of SQL code into the field, and have the database erased. But if the parameters were to set to '@username' then the person would only be able to put in a username without any kind of code.[24]

Enforcement at the coding level

Using object-relational mapping libraries avoids the need to write SQL code. The ORM library in effect will generate parameterized SQL statements from object-oriented code.

Escaping

A popular, though error-prone, way to prevent injections is to attempt to escape all characters that have a special meaning in SQL. The manual for an SQL DBMS explains which characters have a special meaning, which allows creating a comprehensive blacklist of characters that need translation. For instance, every occurrence of a single quote (') in a parameter must be replaced by two single quotes ('') to form a valid SQL string literal. For example, in PHP it is usual to escape parameters using the function mysqli\_real\_escape\_string(); before sending the SQL query:

Pattern check

Integer, float or boolean, string parameters can be checked if their value is valid representation for the given type. Strings that must follow some strict pattern (date, UUID, alphanumeric only, etc.) can be checked if they match this pattern.

Database permissions

Limiting the permissions on the database login used by the web application to only what is needed may help reduce the effectiveness of any SQL injection attacks that exploit any bugs in the web application.

For example, on Microsoft SQL Server, a database logon could be restricted from selecting on some of the system tables which would limit exploits that try to insert JavaScript into all the text columns in the database.

Examples

* In February 2002, Jeremiah Jacks discovered that Guess.com was vulnerable to an SQL injection attack, permitting anyone able to construct a properly-crafted URL to pull down 200,000+ names, credit card numbers and expiration dates in the site's customer database.[28]
* On November 1, 2005, a teenaged hacker used SQL injection to break into the site of a Taiwanese information security magazine from the Tech Target group and steal customers' information.[29]
* On January 13, 2006, Russian computer criminals broke into a Rhode Island government website and allegedly stole credit card data from individuals who have done business online with state agencies.[30]
* On March 29, 2006, a hacker discovered an SQL injection flaw in an official Indian government's tourism site.[31]
* On June 29, 2007, a computer criminal defaced the Microsoft UK website using SQL injection.[32][33] UK website The Register quoted a Microsoft spokesperson acknowledging the problem.
* On September 19, 2007 and January 26, 2009 the Turkish hacker group "m0sted" used SQL injection to exploit Microsoft's SQL Server to hack web servers belonging to McAlester Army Ammunition Plant and the US Army Corps of Engineers respectively.[34]

**XSS**

Cross-site scripting is a type of security vulnerability that can be found in some web applications. XSS attacks enable attackers to inject client-side scripts into web pages viewed by other users. A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same-origin policy.

Cross site scripting (XSS) is an attack in which an attacker injects malicious executable scripts into the code of a trusted application or website. Attackers often initiate an XSS attack by sending a malicious link to a user and enticing the user to click it. If the app or website lacks proper data sanitization, the malicious link executes the attacker’s chosen code on the user’s system. As a result, the attacker can steal the user’s active session cookie.

*Prevention*

How to prevent XSS attacks. To prevent XSS attacks, your application must validate all the input data, make sure that only the allowlisted data is allowed, and ensure that all variable output in a page is encoded before it is returned to the user.

What is the main measure of preventing XSS vulnerabilities?

HTML Sanitization

Output encoding here will prevent XSS, but it will break the intended functionality of the application. The styling will not be rendered. In these cases, HTML Sanitization should be used. HTML Sanitization will strip dangerous HTML from a variable and return a safe string of HTML.

Safe Sinks¶

Security professionals often talk in terms of sources and sinks. If you pollute a river, it'll flow downstream somewhere. It’s the same with computer security. XSS sinks are places where variables are placed into your webpage.

Thankfully, many sinks where variables can be placed are safe. This is because these sinks treat the variable as text and will never execute it. Try to refactor your code to remove references to unsafe sinks like innerHTML, and instead use textContent or value.

Other controls

Cookie Attributes - These change how JavaScript and browsers can interact with cookies. Cookie attributes try to limit the impact of an XSS attack but don’t prevent the execution of malicious content or address the root cause of the vulnerability.

Content Security Policy - An allowlist that prevents content being loaded. It’s easy to make mistakes with the implementation so it should not be your primary defense mechanism. Use a CSP as an additional layer of defense and have a look at the cheatsheet here.

Web Application Firewalls - These look for known attack strings and block them. WAF’s are unreliable and new bypass techniques are being discovered regularly. WAFs also don’t address the root cause of an XSS vulnerability. In addition, WAFs also miss a class of XSS vulnerabilities that operate exclusively client-side.

**Email Spoofing**

Email spoofing is a type of cyberattack that targets businesses by using emails with forged sender addresses. Because the recipient trusts the alleged sender, they are more likely to open the email and interact with its contents, such as a malicious link or attachment.

For example, a spoofed email may pretend to be from a well-known shopping website, asking the recipient to provide sensitive data, such as a password or credit card number. Alternatively, a spoofed email may include a link that installs malware on the user's device if clicked.

**Phishing Attacks**

Phishing is when attackers send malicious emails designed to trick people into falling for a scam. Typically, the intent is to get users to reveal financial information, system credentials or other sensitive data.

**Virus malware**

**(look at pdf)**

Hypertext Transfer Protocol Secure

Hypertext Transfer Protocol Secure (https) is a combination of the Hypertext Transfer Protocol (HTTP) with the Secure Socket Layer (SSL)/Transport Layer Security (TLS) protocol. TLS is an authentication and security protocol widely implemented in browsers and Web servers.