|  |  |
| --- | --- |
| **Presenter:** | **Mohammad Tawalbeh** |
| **​​Topic:​** | **Password Cracking with Brute Force Algorithm and Dictionary Attack Using Parallel Programming** |
| **Publishing Date:** | **12 May 2023** |
| **Authors:** | **Ibrahim Alkhwaja , Mohammed Albugami, Ali Alkhwaja, Mohammed Alghamdi, Hussam Abahussain, Faisal Alfawaz, Abdullah Almurayh and Nasro Min-Allah** |
| **Problem:** | **Password Cracking with Brute Force Attacks** |
| **Solution:** | **Implement stronger password policies, utilize multi-factor authentication, and conduct regular security audits to protect sensitive information from password-cracking techniques.** |
| **Draw Backs:** | **While implementing stronger password policies, multi-factor authentication, and regular security audits can enhance security, drawbacks may include user inconvenience, increased administrative overhead, and potential resistance to change.** |

**Brute Force**

**Exploring password-cracking methods serves as a cornerstone in the realm of information security, shedding light on the weaknesses inherent in feeble passwords and the imperative for bolstered security protocols to safeguard critical data. Although both strategies aim at uncovering passwords, they diverge in their methodologies. The brute force algorithm exhaustively generates every conceivable combination of characters within a defined range and length, whereas the dictionary attack sifts through a predetermined list of words.**

**This investigation delves into the efficacy of these approaches using parallel implementations in Python, C++, and Hashcat. The findings underscore that the NVIDIA GeForce GTX 1050 Ti, leveraging CUDA, significantly outpaces the Intel(R) HD Graphics 630 GPU in password cracking, exhibiting a remarkable speedup of 11.5× and 10.4× for passwords with and without special characters, respectively. Notably, the inclusion of special characters prolongs the password-cracking endeavor, intensifying the challenge.**

**Our experimentation reveals that parallel processing markedly enhances the efficiency of password-cracking techniques. The brute force algorithm achieves a speedup of 1.9× with six cores, while the dictionary attack demonstrates a speedup of 4.4× with eight-core static scheduling. Through the lens of password-cracking methodologies, the necessity for fortified security measures to shield sensitive data becomes glaringly evident, underscoring the vulnerability inherent in weak passwords.**

**SQL INJECTION**

|  |  |
| --- | --- |
| **Presenter:** | **Mohammad Basandi** |
| **​​Topic:​** | **A STUDY ON SQL INJECTION TECHNIQUES** |
| **Publishing Date:** | **12.11.2016** |
| **Authors:** | **Rubidha Devi, Ramasamy Venkatesan, Raghuraman Koteeswaran** |
| **Problem:** | **SQL Injection vulnerability** |
| **Solution:** | **Implementing robust input validation and parameterized queries can mitigate the risk of SQL injection attacks, bolstering database security effectively.** |
| **Draw Backs:** | **The complexity and maintenance overhead of implementing robust input validation and parameterized queries may pose challenges, and they might not fully eliminate the risk of SQL injection.** |

**In today's era, the proliferation of cyber threats poses a significant risk to the integrity and security of personal and organizational data across various industries. Safeguarding databases against security breaches has become imperative worldwide. Among the array of threats, SQL injection stands out as a prevalent and highly exploitable vulnerability, capable of compromising the entirety of an organization's database, regardless of whether it operates in the private or public sector. This technique involves injecting malicious code into web pages to target data-driven applications. By manipulating SQL statements to satisfy an "ALWAYS TRUE" condition, attackers can gain unauthorized access to sensitive information.**

**This research paper aims to provide a comprehensive overview of SQL injection, covering fundamental concepts, different types, and recent attack scenarios. However, a thorough examination of this issue would be incomplete without delving into the underlying algorithms utilized to exploit vulnerabilities in our interconnected digital landscape. Tautology SQL injection, a prominent code injection technique, is frequently employed in data-driven attacks, as documented in security literature, inflicting severe damage to organizational data repositories and exposing confidential information.**

**Cross site scripting (XSS)**

|  |  |
| --- | --- |
| **Presenter:** | **Shahed Al-Azzam** |
| **​​Topic:​** | **Detection of XSS Attacks in Web Applications: A Machine**  **Learning Approach** |
| **Publishing Date:** | **January 2021** |
| **Authors:** | **Bronjon Gogoi, Tasiruddin Ahmed, Hemanta Kumar Saikia** |
| **Problem:** | **XSS Attacks in Web applications** |
| **Solution:** | **Using machine learning, the solution detects XSS attacks in web apps through data collection, model training, and evaluation. It deploys effective models to enhance web security.** |
| **Draw Backs:** | **using machine learning for XSS attack detection is the risk of computational overhead, biases in training data, susceptibility to adversarial attacks, and challenges with interpretability, potentially leading to false positives/negatives.** |

**With the internet's widespread adoption, web applications and sites have become increasingly prevalent. Consequently, cyber-attacks targeting these platforms have surged as well. Among these, XSS (Cross-Site Scripting) attacks stand out as particularly prevalent and troublesome. In fact, they rank as the second most significant web application security risk according to OWASP's Top 10 list.**

**Traditional defense mechanisms, such as rule-based and signature-based web application firewalls, have been commonly used to mitigate XSS threats. However, these defenses can be circumvented by obfuscating attack payloads, rendering them ineffective against sophisticated attacks designed to bypass such measures.**

**To address this challenge, this paper proposes leveraging machine learning (ML) techniques for XSS attack detection. By employing various ML algorithms, we aim to evaluate their effectiveness in identifying XSS attacks within web applications and sites. This comparative analysis seeks to shed light on the most robust ML-based approach for bolstering web security against XSS threats.**

**Local file inclusion**

|  |  |
| --- | --- |
| **Presenter:** | **Abdallah Al-Shorman** |
| **​​Topic:​** | **Deep Dive into Directory Traversal and File Inclusion Attacks leads to Privilege Escalation** |
| **Publishing Date:** | **16 May 2021** |
| **Authors:** | **Mrunalsinh Chawda, Dr. Priyanka Sharma, Mr. Jatin Patel** |
| **Problem:** | **Directory Traversal and File Inclusion Attacks(Vulnerability)** |
| **Solution:** | 1. **Enforce rigorous input validation to sanitize user inputs and prevent malicious traversal attempts.** 2. **Implement strict access controls to restrict file access and minimize the impact of successful attacks.** 3. **Regularly update software, conduct security audits, and educate personnel to maintain robust defenses against directory traversal vulnerabilities.**   **Top of Form** |
| **Draw Backs:** | **Implementing measures against directory traversal vulnerabilities may hinder user experience with overly stringent input validation and add complexity to system management through strict access controls. Additionally, the need for regular updates and security audits can disrupt workflow, impacting productivity. Balancing security with usability is crucial to effectively manage these drawbacks.** |

**In the realm of modern web applications, directory traversal vulnerabilities loom as a formidable threat, potentially affording attackers access to a broad spectrum of files, both arbitrary and sensitive. Through the exploitation of these vulnerabilities or any associated misconfigurations, malicious actors can ascend to elevated privileges, including the coveted root access. Safeguarding against such exploits mandates a vigilant approach during the developmental stages of web applications, ensuring that no arbitrary files are left vulnerable to public access via the production server, thus curtailing the avenues for exploitation.**

**Directory traversal vulnerabilities capitalize on the dynamic file inclusion mechanisms intrinsic to programming frameworks. Local file inclusion ensues when unregulated user inputs, such as form values or headers, are manipulated to construct inclusion paths for files.**

**Exploiting directory traversal attacks on web servers empowers adversaries to orchestrate a multitude of nefarious deeds. By seamlessly integrating these attacks with code injection techniques, malefactors can infiltrate a web server, precipitating the desecration of websites. Path-traversal exploits exploit vulnerable website parameters to embed URL references to remotely hosted malicious code, thereby catalyzing remote code execution and opening the door to privilege escalation assaults.**