**Authentication vulnerabilities (ช่องโหว่การตรวจสอบสิทธิ์)**

**What is authentication?** การตรวจสอบความถูกต้องคืออะไร

การตรวจสอบความถูกต้อง คือ กระบวนการตรวจสอบตัวตนของผู้ใช้หรือลูกค้า เว็บไซต์อาจเปิดเผยต่อใครก็ตามที่เชื่อมต่อกับอินเทอร์เน็ต ซึ่งทำให้กลไกการตรวจสอบความถูกต้องที่แข็งแกร่งเป็นส่วนสำคัญของการรักษาความปลอดภัยบนเว็บที่มีประสิทธิภาพ

การตรวจสอบความถูกต้องมีสามประเภทหลัก

* สิ่งที่รู้ เช่น รหัสผ่านหรือคำตอบสำหรับคำถามด้านความปลอดภัย "knowledge factors"
* สิ่งที่มี physical object เช่น โทรศัพท์มือถือหรือโทเค็นความปลอดภัย "possession factors"
* สิ่งที่เป็นหรือทำ biometrics or patterns of behavior "inherence factors"

**What is the difference between authentication and authorization?**

Authentication การพิสูจน์ตัวตน คือ กระบวนการยืนยันว่าผู้ใช้เป็นใคร

Authorization การอนุญาต คือ การยืนยันว่าผู้ใช้ได้รับอนุญาตให้ทำบางสิ่งหรือไม่

ตัวอย่างเช่น การพิสูจน์ตัวตนจะกำหนดว่าบุคคลที่พยายามเข้าถึงเว็บไซต์โดยใช้ชื่อผู้ใช้ Carlos123 เป็นคนเดียวกับที่สร้างบัญชีนั้นจริงหรือไม่

เมื่อ Carlos123 ผ่านการพิสูจน์ตัวตนแล้ว การอนุญาตจะกำหนดว่าบุคคลนั้นได้รับอนุญาตให้ทำอะไร พวกเขาอาจได้รับอนุญาตให้เข้าถึงข้อมูลส่วนบุคคลเกี่ยวกับผู้ใช้รายอื่น เช่น การลบบัญชีของผู้ใช้รายอื่น

**How do authentication vulnerabilities arise?**

ช่องโหว่ส่วนใหญ่ในกลไกการยืนยันตัวตนเกิดขึ้นได้สองวิธีดังต่อไปนี้

* กลไกการยืนยันตัวตนอ่อนแอ เนื่องจากไม่สามารถป้องกันการโจมตีแบบ brute-force ได้อย่างเหมาะสม
* ข้อบกพร่องทางตรรกะหรือการเขียนโค้ดที่ไม่ดีในการใช้งานทำให้ผู้โจมตีสามารถข้ามกลไกการยืนยันตัวตนได้ทั้งหมด เรียกว่า "broken authentication"

ในหลายพื้นที่ของการพัฒนาเว็บ ข้อบกพร่องทางตรรกะทำให้เว็บไซต์ทำงานผิดปกติ ซึ่งอาจเป็นปัญหาความปลอดภัยหรือไม่ก็ได้ อย่างไรก็ตาม เนื่องจากการยืนยันตัวตนมีความสำคัญอย่างยิ่งต่อความปลอดภัย จึงมีความเป็นไปได้สูงที่ตรรกะการยืนยันตัวตนที่บกพร่องจะทำให้เว็บไซต์เผชิญกับปัญหาความปลอดภัย

**What is the impact of vulnerable authentication?**

ผลกระทบของช่องโหว่ในการตรวจสอบสิทธิ์อาจร้ายแรง หากผู้โจมตีสามารถหลีกเลี่ยงการตรวจสอบสิทธิ์หรือใช้วิธีการ brute-force เข้าสู่บัญชีผู้ใช้รายอื่น ผู้โจมตีจะสามารถเข้าถึงข้อมูลและฟังก์ชันต่างๆ ของบัญชีที่ถูกบุกรุกได้ทั้งหมด หากผู้โจมตีสามารถเจาะระบบบัญชีที่มีสิทธิ์ใช้งานสูง เช่น ผู้ดูแลระบบได้ ผู้โจมตีจะสามารถควบคุมแอปพลิเคชันทั้งหมดได้อย่างสมบูรณ์และอาจเข้าถึงโครงสร้างพื้นฐานภายในได้

**What is the impact of vulnerable authentication?**

แม้ว่าบัญชีที่มีสิทธิ์ใช้งานต่ำอาจทำให้ผู้โจมตีเข้าถึงข้อมูลที่ไม่ควรเข้าถึงได้ เช่น ข้อมูลทางธุรกิจที่ละเอียดอ่อนทางการค้า ถึงบัญชีจะไม่สามารถเข้าถึงข้อมูลที่ละเอียดอ่อนได้ แต่ผู้โจมตียังสามารถเข้าถึง additional pages ซึ่งจะเป็นช่องทางการโจมตีเพิ่มเติม การโจมตีที่มีความรุนแรงสูงมักไม่สามารถเกิดขึ้นได้จาก publicly accessible pages แต่อาจเกิดขึ้นได้จาก internal page

**Vulnerabilities in password-based login**

สำหรับเว็บไซต์ที่ใช้กระบวนการเข้าสู่ระบบโดยใช้รหัสผ่าน ผู้ใช้จะต้องลงทะเบียนเพื่อสร้างบัญชีของตนเองหรือได้รับบัญชีจากผู้ดูแล ซึ่งบัญชีนี้จะถูกเชื่อมโยงกับชื่อผู้ใช้และรหัสผ่านลับเฉพาะ เมื่อเข้าสู่ระบบผู้ใช้จะต้องกรอกข้อมูลเหล่านี้ในแบบฟอร์มเพื่อยืนยันตัวตน

ในกรณีนี้ การที่ผู้ใช้รู้รหัสผ่านลับนั้นถือเป็นหลักฐานเพียงพอในการยืนยันตัวตนของผู้ใช้ ซึ่งหมายความว่าความปลอดภัยของเว็บไซต์อาจถูกทำลายได้หากผู้โจมตีสามารถดึงข้อมูลหรือเดารหัสผ่านของผู้ใช้อื่นได้

วิธีการที่ผู้โจมตีอาจใช้ในการโจมตีมีหลายรูปแบบ ซึ่งบทความต่อไปนี้จะแสดงให้เห็นถึงวิธีการใช้การโจมตีแบบเดารหัสผ่าน (brute-force attack) และข้อบกพร่องในกระบวนการป้องกันการโจมตีแบบเดารหัส นอกจากนี้ยังอธิบายเกี่ยวกับช่องโหว่ของการยืนยันตัวตนแบบ HTTP basic authentication

**Brute-force attacks**

การโจมตีแบบเดารหัสผ่าน (Brute-force attack) คือ การที่ผู้โจมตีใช้วิธีการลองผิดลองถูกเพื่อเดาชื่อผู้ใช้และรหัสผ่านที่ถูกต้อง โดยการโจมตีประเภทนี้มักจะทำผ่านระบบอัตโนมัติที่ใช้รายการคำศัพท์ที่มีชื่อผู้ใช้และรหัสผ่านอยู่แล้ว การใช้กระบวนการอัตโนมัตินี้ โดยเฉพาะการใช้เครื่องมือเฉพาะทาง ทำให้ผู้โจมตีสามารถพยายามเข้าสู่ระบบจำนวนมากได้อย่างรวดเร็ว

การโจมตีแบบเดารหัสไม่ใช่การเดาสุ่มที่ไม่มีหลักการเสมอไป ผู้โจมตีอาจใช้หลักการพื้นฐานหรือข้อมูลที่มีอยู่สาธารณะในการช่วยเดาชื่อผู้ใช้และรหัสผ่าน ทำให้การโจมตีแบบเดารหัสมีประสิทธิภาพมากยิ่งขึ้น ซึ่งทำให้เว็บไซต์ที่ใช้การเข้าสู่ระบบด้วยรหัสผ่านเพียงอย่างเดียวอาจมีความเสี่ยงสูง หากไม่มีการป้องกันการโจมตีแบบเดารหัสที่เพียงพอ

**Brute-forcing usernames**

ชื่อผู้ใช้ (Username) มักเป็นสิ่งที่เดาได้ง่ายหากมีรูปแบบที่สามารถจดจำได้ เช่น อีเมล โดยเฉพาะในธุรกิจต่าง ๆ ที่มักใช้รูปแบบเช่น firstname.lastname@somecompany.com อย่างไรก็ตาม ถึงแม้ว่าจะไม่มีรูปแบบชัดเจน บางครั้งบัญชีที่มีสิทธิ์เข้าถึงสูง (high-privileged accounts) ก็ถูกสร้างด้วยชื่อผู้ใช้ที่คาดเดาได้ เช่น admin หรือ administrator

ในระหว่างการตรวจสอบความปลอดภัย ควรตรวจสอบว่าเว็บไซต์เปิดเผยชื่อผู้ใช้ที่เป็นไปได้หรือไม่ เช่น สามารถเข้าถึงโปรไฟล์ของผู้ใช้ได้โดยไม่ต้องเข้าสู่ระบบหรือไม่ แม้ว่าจะไม่สามารถเห็นเนื้อหาของโปรไฟล์ แต่ชื่อที่ใช้ในโปรไฟล์อาจเป็นชื่อเดียวกับชื่อผู้ใช้ที่ใช้เข้าสู่ระบบ นอกจากนี้ ควรตรวจสอบการตอบสนองของ HTTP เพื่อดูว่ามีการเปิดเผยที่อยู่อีเมลหรือไม่ บางครั้งการตอบสนองอาจมีที่อยู่อีเมลของผู้ใช้ที่มีสิทธิ์สูง เช่น ผู้ดูแลระบบหรือฝ่ายสนับสนุนด้านไอที

**Brute-forcing passwords**

การโจมตีแบบเดารหัสผ่าน (Brute-forcing passwords) สามารถใช้กับรหัสผ่านได้เช่นกัน ซึ่งระดับความยากง่ายในการเดาจะขึ้นอยู่กับความแข็งแกร่งของรหัสผ่าน หลายเว็บไซต์จึงมีนโยบายการตั้งรหัสผ่านที่บังคับให้ผู้ใช้สร้างรหัสผ่านที่มีความซับซ้อนสูง เพื่อให้ยากต่อการโจมตีแบบเดารหัสผ่านเพียงอย่างเดียว โดยปกติแล้วนโยบายนี้จะกำหนดให้รหัสผ่านต้องมี

- จำนวนอักขระขั้นต่ำ

- การผสมผสานระหว่างตัวอักษรเล็กและใหญ่

- อักขระพิเศษอย่างน้อยหนึ่งตัว

**Brute-forcing passwords - Continued**

แม้ว่ารหัสผ่านที่มีความซับซ้อนสูง (high-entropy) จะยากต่อการเดาด้วยการใช้คอมพิวเตอร์เพียงอย่างเดียว แต่เรายังสามารถใช้ความรู้พื้นฐานเกี่ยวกับพฤติกรรมมนุษย์ในการค้นหาช่องโหว่ที่ผู้ใช้อาจสร้างโดยไม่รู้ตัว แทนที่จะสร้างรหัสผ่านที่แข็งแกร่งด้วยการสุ่มอักขระ ผู้ใช้มักเลือกใช้รหัสผ่านที่จดจำง่ายและพยายามปรับให้ตรงตามนโยบายรหัสผ่าน เช่น หากใช้ "mypassword" ไม่ได้ ผู้ใช้อาจลองใช้ "Mypassword1!" หรือ "Myp4$$w0rd" แทน

ในกรณีที่นโยบายกำหนดให้ผู้ใช้ต้องเปลี่ยนรหัสผ่านอย่างสม่ำเสมอ ผู้ใช้มักจะเปลี่ยนเพียงเล็กน้อยและเป็นรูปแบบที่เดาได้ เช่น เปลี่ยนจาก "Mypassword1!" เป็น "Mypassword1?" หรือ "Mypassword2!"

ความรู้เกี่ยวกับรูปแบบรหัสผ่านที่น่าจะใช้และการเปลี่ยนแปลงที่คาดการณ์ได้นี้ทำให้การโจมตีแบบเดารหัสผ่านสามารถทำได้อย่างซับซ้อนและมีประสิทธิภาพมากขึ้นกว่าการลองทุกการรวมกันของอักขระ

**Username enumeration**

การระบุชื่อผู้ใช้ (Username enumeration) คือ กระบวนการที่ผู้โจมตีสามารถสังเกตการเปลี่ยนแปลงของพฤติกรรมเว็บไซต์เพื่อดูว่าชื่อผู้ใช้ที่กรอกนั้นมีอยู่จริงหรือไม่

การระบุชื่อผู้ใช้มักเกิดขึ้นในหน้าล็อกอิน เช่น เมื่อผู้ใช้กรอกชื่อผู้ใช้ที่ถูกต้องแต่รหัสผ่านผิด หรือในแบบฟอร์มลงทะเบียน เมื่อผู้ใช้กรอกชื่อผู้ใช้ที่มีคนใช้แล้ว ซึ่งทำให้ผู้โจมตีสามารถสร้างรายการชื่อผู้ใช้ที่ถูกต้องได้อย่างรวดเร็ว ลดเวลาและความพยายามที่จำเป็นในการโจมตีแบบเดารหัสผ่าน

ในระหว่างการโจมตีแบบเดารหัสผ่านบนหน้าล็อกอิน คุณควรให้ความสนใจกับความแตกต่างดังต่อไปนี้:

1. **รหัสสถานะ (Status codes):** ในการโจมตีแบบเดารหัสผ่าน รหัสสถานะ HTTP ที่ถูกส่งกลับมามักจะเหมือนกันในกรณีส่วนใหญ่ เนื่องจากชื่อผู้ใช้หรือรหัสผ่านส่วนใหญ่จะไม่ถูกต้อง หากมีการทายที่ส่งคืนรหัสสถานะที่แตกต่างไป อาจเป็นสัญญาณบ่งชี้ว่าชื่อผู้ใช้นั้นถูกต้อง หลักปฏิบัติที่ดีคือให้เว็บไซต์ส่งรหัสสถานะเดียวกันเสมอ ไม่ว่าผลลัพธ์จะเป็นอย่างไร แต่บางเว็บไซต์ไม่ได้ปฏิบัติตามหลักการนี้
2. **ข้อความแสดงข้อผิดพลาด (Error messages):** บางครั้ง ข้อความแสดงข้อผิดพลาดจะแตกต่างกัน ขึ้นอยู่กับว่าทั้งชื่อผู้ใช้และรหัสผ่านไม่ถูกต้องหรือมีแค่รหัสผ่านที่ไม่ถูกต้อง หลักปฏิบัติที่ดีคือให้เว็บไซต์ใช้ข้อความที่เหมือนกันในทั้งสองกรณี แต่บางครั้งอาจมีข้อผิดพลาดเล็กน้อยในการพิมพ์ ทำให้ข้อความทั้งสองต่างกัน แม้แต่การพิมพ์ผิดเพียงหนึ่งตัวอักษรก็ทำให้ข้อความแสดงผลแตกต่างกันได้ แม้ว่าตัวอักษรนั้นจะไม่แสดงบนหน้าเว็บก็ตาม
3. **เวลาตอบสนอง (Response times):** หากการร้องขอส่วนใหญ่ถูกประมวลผลด้วยเวลาตอบสนองที่คล้ายกัน แต่บางคำขอมีเวลาที่แตกต่างไป อาจแสดงว่ามีบางสิ่งที่แตกต่างเกิดขึ้นเบื้องหลัง เช่น เว็บไซต์อาจตรวจสอบความถูกต้องของรหัสผ่านก็ต่อเมื่อชื่อผู้ใช้ถูกต้อง ซึ่งอาจทำให้เวลาตอบสนองเพิ่มขึ้นเล็กน้อย วิธีนี้อาจละเอียดอ่อน แต่ผู้โจมตีสามารถทำให้ความล่าช้านี้ชัดเจนขึ้นได้โดยการป้อนรหัสผ่านที่ยาวมากจนเว็บไซต์ใช้เวลามากขึ้นในการประมวลผล

**Flawed brute-force protection** การป้องกันการโจมตีแบบ Brute-Force ที่มีข้อบกพร่อง

การโจมตีแบบ Brute-Force มักเกี่ยวข้องกับการลองเดารหัสผ่านผิดจำนวนมากก่อนที่จะสามารถเข้าถึงบัญชีได้สำเร็จ การป้องกัน Brute-Force มีจุดมุ่งหมายเพื่อทำให้การโจมตีอัตโนมัติทำได้ยากขึ้น และชะลอความเร็วในการพยายามเข้าสู่ระบบของผู้โจมตี

วิธีที่พบได้บ่อยในการป้องกัน Brute-Force มี 2 วิธีหลัก

1. **ล็อกบัญชี** หากผู้ใช้งานระยะไกลพยายามเข้าสู่ระบบผิดหลายครั้งเกินไป
2. **บล็อกที่อยู่ IP** ของผู้ใช้งานระยะไกลที่พยายามเข้าสู่ระบบหลายครั้งในระยะเวลาอันสั้น

ทั้งสองวิธีมีระดับการป้องกันที่แตกต่างกันไป แต่ก็ไม่ได้สมบูรณ์แบบ

ตัวอย่างเช่น บางระบบอาจบล็อก IP ของคุณหากคุณเข้าสู่ระบบผิดหลายครั้ง และบางกรณีตัวนับจำนวนครั้งที่พลาดจะถูกรีเซ็ตหากเจ้าของ IP เข้าสู่ระบบสำเร็จ นั่นหมายความว่าผู้โจมตีสามารถเข้าสู่ระบบบัญชีของตนเองทุกๆ ครั้งหลังจากลองผิด เพื่อรีเซ็ตตัวนับ ทำให้ระบบป้องกันนี้แทบไร้ประโยชน์

กรณีนี้ ผู้โจมตีสามารถใส่ข้อมูลเข้าสู่ระบบของตนเองลงในรายการคำที่ใช้เดารหัสผ่าน (wordlist) เป็นระยะๆ เพื่อหลีกเลี่ยงการถูกบล็อกจากระบบได้อย่างง่ายดาย

**User Rate Limiting** การจำกัดอัตราการใช้งานของผู้ใช้

เป็นวิธีที่เว็บไซต์ใช้ป้องกันการโจมตีแบบ Brute-Force โดยการจำกัดจำนวนคำขอเข้าสู่ระบบที่สามารถทำได้ในช่วงเวลาสั้น ๆ หากมีคำขอมากเกินไป จะทำให้ IP ของผู้ใช้งานถูกบล็อก โดยทั่วไป IP จะถูกปลดบล็อกได้ในวิธีดังนี้:

1. ปลดบล็อกอัตโนมัติหลังจากผ่านช่วงเวลาหนึ่ง
2. ปลดบล็อกด้วยการดำเนินการของผู้ดูแลระบบ
3. ปลดบล็อกโดยผู้ใช้ผ่านการแก้ CAPTCHA

การจำกัดอัตราการใช้งานเป็นที่นิยมมากกว่าการล็อกบัญชี เนื่องจากลดความเสี่ยงของปัญหาการระบุชื่อผู้ใช้งาน (Username Enumeration) และการโจมตีแบบ Denial of Service (DoS) อย่างไรก็ตาม วิธีนี้ยังไม่ปลอดภัยอย่างสมบูรณ์ เนื่องจากผู้โจมตีสามารถหลีกเลี่ยงการบล็อกได้โดยการปลอม IP ของตน หรือใช้วิธีการเดารหัสผ่านหลายรายการในคำขอเดียว (Single Request) เพื่อลดการตรวจจับจากระบบบล็อกนี้

**HTTP Basic Authentication** เป็นวิธีการยืนยันตัวตนที่ยังคงพบได้ในบางกรณี เนื่องจากความเรียบง่ายและการติดตั้งที่สะดวก กระบวนการนี้ทำงานโดยให้ฝั่งไคลเอนต์รับโทเค็นการยืนยันตัวตนจากเซิร์ฟเวอร์ ซึ่งโทเค็นนี้ถูกสร้างขึ้นจากการรวมชื่อผู้ใช้และรหัสผ่านเข้าด้วยกัน จากนั้นเข้ารหัสในรูปแบบ Base64 เบราว์เซอร์จะเก็บโทเค็นนี้ไว้และจัดการโดยอัตโนมัติ ในทุกคำขอถัดไป เบราว์เซอร์จะเพิ่มโทเค็นนี้ลงในส่วนหัวของคำขอ HTTP

โดยมีรูปแบบเป็น Authorization: Basic base64(username: password)

อย่างไรก็ตาม เนื่องจาก Base64 เป็นเพียงวิธีการเข้ารหัสแบบง่ายที่สามารถถอดกลับได้ง่าย การใช้งาน HTTP Basic Authentication จึงควรอยู่ภายใต้การเชื่อมต่อ HTTPS เพื่อเพิ่มความปลอดภัยของข้อมูล แม้จะใช้งานง่ายแต่โดยทั่วไปถือว่าเป็นวิธีการยืนยันตัวตนที่ไม่ปลอดภัยด้วยเหตุผลหลายประการ

* วิธีนี้ต้องส่งข้อมูลชื่อผู้ใช้และรหัสผ่านซ้ำ ๆ ในทุกคำขอ ซึ่งหากเว็บไซต์ไม่ได้ใช้ **HSTS (HTTP Strict Transport Security)** ข้อมูลการเข้าสู่ระบบของผู้ใช้อาจถูกดักจับระหว่างทางผ่านการโจมตีแบบ **Man-in-the-Middle (MitM)**
* การใช้งาน HTTP Basic Authentication มักขาดการป้องกันการโจมตีแบบ Brute Force เนื่องจากโทเค็นที่ใช้ประกอบด้วยค่าคงที่ ทำให้ตกเป็นเป้าหมายของการเดารหัสผ่านได้ง่าย
* HTTP Basic Authentication ยังเสี่ยงต่อการโจมตีที่เกี่ยวข้องกับเซสชัน เช่น **Cross-Site Request Forgery (CSRF)** ซึ่งไม่มีการป้องกันในตัว

แม้บางครั้งการเจาะระบบผ่านช่องโหว่ของ HTTP Basic Authentication อาจทำให้เข้าถึงหน้าเว็บที่ดูเหมือนไม่สำคัญ แต่การเปิดเผยข้อมูลการยืนยันตัวตนอาจนำไปสู่การโจมตีเพิ่มเติม หรือการนำข้อมูลที่ได้ไปใช้ในบริบทอื่นที่มีความสำคัญและเป็นความลับมากกว่า

**Vulnerabilities in Multi-Factor Authentication** ช่องโหว่ในระบบการยืนยันตัวตนแบบหลายปัจจัย

แม้ว่าการยืนยันตัวตนแบบหลายปัจจัยจะช่วยเพิ่มความปลอดภัยให้มากกว่าการใช้การยืนยันตัวตนแบบปัจจัยที่อาศัยเพียงรหัสผ่านเพียงอย่างเดียว แต่ระบบนี้ก็ยังมีช่องโหว่ที่สามารถถูกโจมตีได้

เว็บไซต์หลายแห่งเริ่มใช้ MFA เพื่อเพิ่มความมั่นใจในการพิสูจน์ตัวตน โดยต้องการให้ผู้ใช้งานยืนยันตัวตนผ่านหลายปัจจัย เช่น รหัสผ่าน และปัจจัยเพิ่มเติม เช่น รหัส OTP, การยืนยันผ่านอุปกรณ์เฉพาะ หรือไบโอเมตริกซ์ (เช่น ลายนิ้วมือหรือการสแกนใบหน้า)

อย่างไรก็ตาม การโจมตีช่องโหว่ใน MFA อาจเกิดขึ้นได้ในหลายรูปแบบ

* **การดักจับโค้ด OTP (One-Time Password):** ผู้โจมตีอาจใช้เทคนิคฟิชชิ่งหรือมัลแวร์เพื่อหลอกล่อให้ผู้ใช้เปิดเผยโค้ด
* **ช่องโหว่ในระบบสำรอง:** เช่น หากระบบมีทางเลือกในการข้าม MFA (เช่น การตอบคำถามเพื่อกู้คืนบัญชี) ผู้โจมตีอาจใช้วิธีนี้เพื่อหลีกเลี่ยง MFA
* **การโจมตีแบบ Man-in-the-Middle (MitM):** ผู้โจมตีสามารถดักจับข้อมูลระหว่างผู้ใช้และเซิร์ฟเวอร์ในระหว่างกระบวนการยืนยันตัวตน

แม้ MFA จะเพิ่มระดับความปลอดภัยได้ แต่การออกแบบและการใช้งานระบบอย่างไม่รอบคอบอาจสร้างจุดอ่อนที่ผู้โจมตีสามารถใช้ประโยชน์ได้

แม้ว่าการยืนยันตัวตนด้วยไบโอเมตริกซ์จะไม่ใช่ตัวเลือกที่เหมาะสมสำหรับเว็บไซต์ส่วนใหญ่ แต่การยืนยันตัวตนแบบสองปัจจัย (2FA) ซึ่งผสมผสานระหว่าง "สิ่งที่คุณรู้" และ "สิ่งที่คุณมี" กำลังได้รับความนิยมเพิ่มขึ้นเรื่อย ๆ วิธีนี้มักให้ผู้ใช้กรอกรหัสผ่านแบบดั้งเดิมร่วมกับรหัสยืนยันชั่วคราวที่ได้จากอุปกรณ์ทางกายภาพที่อยู่ในครอบครอง

ข้อดีของ 2FA คือ แม้ผู้โจมตีจะได้ข้อมูลปัจจัยหนึ่ง เช่น รหัสผ่าน แต่การเข้าถึงปัจจัยที่สอง เช่น อุปกรณ์หรือรหัส OTP ที่ส่งมาจากช่องทางอื่น (Out-of-Band) จะเป็นเรื่องที่ยากมากขึ้น ทำให้ 2FA มีความปลอดภัยสูงกว่าการยืนยันตัวตนแบบปัจจัยเดียว

อย่างไรก็ตาม ความปลอดภัยของ 2FA ขึ้นอยู่กับการออกแบบและการนำไปใช้ หากมีการติดตั้งอย่างไม่เหมาะสม ผู้โจมตีอาจสามารถหลบเลี่ยงหรือเจาะผ่านระบบได้ เช่นเดียวกับการยืนยันตัวตนแบบปัจจัยเดียว

สิ่งสำคัญอีกประการคือ ประโยชน์สูงสุดของ MFA (Multi-Factor Authentication) จะเกิดขึ้นก็ต่อเมื่อมีการตรวจสอบปัจจัยที่แตกต่างกันจริง ๆ การตรวจสอบปัจจัยเดียวกันในรูปแบบที่ต่างกัน เช่น 2FA ที่ใช้รหัสผ่านและรหัสยืนยันผ่านอีเมล อาจดูเหมือนมี 2 ปัจจัย แต่แท้จริงแล้วกลับอาศัยแค่ "สิ่งที่คุณรู้" (รหัสผ่านและข้อมูลเข้าสู่อีเมล) ซึ่งไม่ได้เพิ่มระดับความปลอดภัยอย่างแท้จริง

**Two-Factor Authentication Tokens** โทเค็นการยืนยันตัวตนแบบสองปัจจัย

รหัสยืนยันตัวตน (Verification Code) มักถูกอ่านจากอุปกรณ์ทางกายภาพบางอย่างที่ผู้ใช้เป็นเจ้าของ เว็บไซต์ที่ให้ความสำคัญกับความปลอดภัยสูงมักจะมอบอุปกรณ์เฉพาะ เช่น **RSA Token** หรืออุปกรณ์คีย์แพด สำหรับการสร้างรหัสยืนยันโดยตรง อุปกรณ์เหล่านี้ถูกออกแบบมาเพื่อความปลอดภัยโดยเฉพาะ และช่วยลดความเสี่ยงจากการดักจับข้อมูลระหว่างการส่ง

อีกวิธีที่ได้รับความนิยมคือการใช้แอปพลิเคชันเฉพาะ เช่น **Google Authenticator** ซึ่งทำงานในลักษณะคล้ายกัน โดยสร้างรหัสยืนยันจากอุปกรณ์ของผู้ใช้โดยตรง

ในทางกลับกัน บางเว็บไซต์เลือกที่จะส่งรหัสยืนยันไปยังโทรศัพท์มือถือของผู้ใช้ผ่านทาง SMS แม้ว่าวิธีนี้จะยังถือว่าเป็นการตรวจสอบปัจจัย "สิ่งที่คุณมี" แต่กลับมีช่องโหว่ที่อาจถูกโจมตีได้ เช่น:

* **การดักจับรหัสผ่าน SMS:** เนื่องจากรหัสถูกส่งผ่านเครือข่าย SMS แทนที่จะสร้างจากอุปกรณ์ของผู้ใช้โดยตรง ทำให้มีความเสี่ยงที่จะถูกดักจับระหว่างทาง
* **การโจมตีด้วยการสวมสิทธิ์ SIM (SIM Swapping):** ผู้โจมตีอาจใช้ข้อมูลปลอมเพื่อขอรับซิมการ์ดที่มีหมายเลขโทรศัพท์ของเหยื่อ ทำให้สามารถรับข้อความ SMS รวมถึงรหัสยืนยันที่ส่งถึงเหยื่อได้

ดังนั้น แม้การส่งรหัสยืนยันผ่าน SMS จะสะดวก แต่ก็มีความเสี่ยงด้านความปลอดภัยที่ควรระมัดระวังและหลีกเลี่ยงในกรณีที่ต้องการความปลอดภัยสูงสุด

**Bypassing Two-Factor Authentication** การข้ามกระบวนการยืนยันตัวตนแบบสองปัจจัย

ในบางกรณี การติดตั้งระบบยืนยันตัวตนแบบสองปัจจัย (2FA) อาจมีช่องโหว่ที่ทำให้ผู้โจมตีสามารถข้ามกระบวนการนี้ได้ทั้งหมด

ตัวอย่างหนึ่งของช่องโหว่นี้คือ เมื่อผู้ใช้ต้องกรอกรหัสผ่านก่อน จากนั้นจึงกรอกรหัสยืนยันตัวตนในหน้าถัดไป หากระบบอนุญาตให้เข้าถึงสถานะ "ล็อกอินสำเร็จ" หลังจากผ่านขั้นตอนแรก (การกรอกรหัสผ่าน) โดยไม่ได้ตรวจสอบว่าผู้ใช้กรอกรหัสยืนยันตัวตนหรือไม่ ผู้โจมตีอาจใช้ประโยชน์จากช่องโหว่นี้ได้

วิธีการตรวจสอบช่องโหว่ดังกล่าวคือ ทดลองข้ามไปยังหน้า "สำหรับผู้ใช้ที่ล็อกอินแล้ว" โดยตรง หลังจากกรอกรหัสผ่านสำเร็จในขั้นตอนแรก โดยไม่ผ่านการยืนยันในขั้นตอนที่สอง บางครั้งระบบอาจโหลดหน้าเหล่านั้นโดยไม่ได้ตรวจสอบว่าขั้นตอนการยืนยันตัวตนเพิ่มเติมเสร็จสมบูรณ์หรือไม่

การออกแบบระบบ 2FA ที่ไม่รอบคอบเช่นนี้ เป็นช่องโหว่ที่ผู้โจมตีสามารถใช้เพื่อเลี่ยงการป้องกันของระบบได้อย่างง่ายดาย

**Account locking**

การล็อกบัญชีเป็นหนึ่งในวิธีที่เว็บไซต์ใช้เพื่อป้องกันการโจมตีแบบเดาสุ่มรหัสผ่าน (brute force) โดยจะล็อกบัญชีเมื่อมีความพยายามเข้าสู่ระบบล้มเหลวตามเกณฑ์ที่กำหนด เช่น จำนวนครั้งที่กำหนดไว้

การตอบสนองจากเซิร์ฟเวอร์ที่บ่งบอกว่าบัญชีถูกล็อก อาจช่วยให้ผู้โจมตีใช้เป็นข้อมูลในการตรวจสอบชื่อผู้ใช้งานได้

การล็อกบัญชีช่วยเพิ่มการป้องกันการโจมตีแบบ brute force ที่มุ่งเป้าหมายบัญชีเฉพาะเจาะจง แต่ไม่สามารถป้องกันการโจมตีแบบ brute force ที่พยายามเข้าถึงบัญชีใดก็ได้แบบสุ่มได้อย่างมีประสิทธิภาพ

ตัวอย่างวิธีที่สามารถหลีกเลี่ยงการป้องกันการล็อกบัญชีได้ มีดังนี้:

1. สร้างรายการชื่อผู้ใช้ที่น่าจะมีอยู่จริง  
   เช่น จากการสำรวจข้อมูล username หรือใช้รายชื่อผู้ใช้ที่พบได้บ่อย
2. เลือกชุดรหัสผ่านที่สั้นที่สุดที่คิดว่าผู้ใช้อาจใช้  
   โดยจำนวนรหัสผ่านที่เลือกต้องไม่เกินจำนวนครั้งที่ระบบอนุญาตให้ลอง เช่น ถ้าระบบอนุญาตให้ลอง 3 ครั้ง ต้องเลือกไม่เกิน 3 รหัสผ่าน
3. ใช้เครื่องมือ เช่น Burp Intruder  
   ลองใช้รหัสผ่านที่เลือกกับชื่อผู้ใช้ทั้งหมดในรายการ โดยการกระจายความพยายามไปยังบัญชีทั้งหมด วิธีนี้จะช่วยหลีกเลี่ยงการล็อกบัญชีเพราะไม่ได้ลองเกินจำนวนที่กำหนดในแต่ละบัญชี

หากมีเพียงผู้ใช้คนใดคนหนึ่งที่ใช้หนึ่งในรหัสผ่านที่เลือก ก็จะสามารถเข้าถึงบัญชีนั้นได้สำเร็จ

การล็อกบัญชีไม่สามารถป้องกันการโจมตีแบบ **Credential Stuffing** ได้ ซึ่งการโจมตีประเภทนี้ใช้พจนานุกรมขนาดใหญ่ที่ประกอบด้วยคู่ชื่อผู้ใช้และรหัสผ่านจริงจากข้อมูลที่รั่วไหล (Data Breaches) โดยอาศัยข้อเท็จจริงที่ว่าผู้ใช้หลายคนมักใช้ชื่อผู้ใช้และรหัสผ่านเดียวกันในหลายเว็บไซต์ ทำให้มีโอกาสที่ข้อมูลที่ถูกขโมยจะสามารถใช้งานได้ในเว็บไซต์เป้าหมาย

เนื่องจากการโจมตี **Credential Stuffing** จะลองใช้ข้อมูลแต่ละชื่อผู้ใช้เพียงครั้งเดียว การล็อกบัญชีจึงไม่สามารถป้องกันได้ เพราะไม่ได้กระตุ้นให้ระบบล็อกบัญชีตามเงื่อนไขที่ตั้งไว้ การโจมตีนี้อันตรายอย่างมากเพราะสามารถทำให้ผู้โจมตีเข้าถึงบัญชีจำนวนมากได้ในคราวเดียว โดยใช้กระบวนการที่เป็นอัตโนมัติ