URL Shortener - Security & Scalability Documentation

Security Concerns and Solutions

1. Malicious URL Redirection

Concern: ผู้ใช้อาจส่ง URL อันตรายเพื่อหลอกให้ผู้อื่นคลิกลิงก์ที่นำไปสู่เว็บไซต์ที่มีมัลแวร์หรือฟิชชิ่ง เนื่องจากผู้ใช้มักจะไม่สามารถเห็น URL ปลายทางที่แท้จริงเมื่อเห็นเพียง URL สั้น

Solution:

- สร้าง URL Blacklist service ไว้สำหรับตรวจสอบว่ามี URL ใดที่ห้ามใช้งานยกตัวต่างเช่น iplogger, etc
- ตรวจสอบ URL input ผ่าน regex และ validation เพื่อยืนยันว่าเป็น URL ที่ถูกต้องตามมาตรฐาน
- เชื่อมต่อกับ URL safety services เช่น Google Safe Browsing API เพื่อตรวจสอบว่า URL ที่ผู้ใช้ต้องการย่อไม่ได้อยู่ในรายการเว็บไซต์อันตราย
- แสดงหน้า warning ก่อน redirect ในกรณีที่มีความเสี่ยง โดยแสดง URL ปลายทางให้ผู้ใช้ยืนยันก่อนเข้าถึง

2. URL Enumeration / Information Leakage

Concern: ผู้โจมตีอาจทำ brute force กับ key เพื่อค้นหา URLs ทั้งหมด ซึ่งอาจเข้าถึงข้อมูลส่วนตัวหรือข้อมูลที่ไม่ได้มีไว้เผยแพร่

Solution:

- ใช้ algorithm สร้าง key ที่คาดเดายาก แทนการใช้ key ที่มีรูปแบบง่ายต่อการคาดเดา (ในproject นี้เลือกใช้ nanoid)
- จำกัดจำนวนการเข้าถึงด้วย rate limiting โดยจำกัดจำนวนการเรียกใช้งาน API จากแต่ละ IP address

3. SQL Injection

Concern: การโจมตีผ่านการใส่คำสั่ง SQL ในช่อง input เพื่อเข้าถึงหรือเปลี่ยนแปลงข้อมูลในฐานข้อมูล

Solution:

- ใช้ ORM Service ที่น่าเชื่อถือ
- ใช้ parameterized queries กับทุกคำสั่ง SQL เพื่อแยกข้อมูลผู้ใช้ออกจากคำสั่ง SQL
- ทำ input sanitization โดยกรองหรือแปลงอักขระพิเศษที่อาจใช้ในการโจมตี
- จำกัดสิทธิ์ของ database user ให้มีสิทธิ์เฉพาะที่จำเป็นต่อการทำงาน

Scalability Concerns and Solutions

1. Database Performance

Concern: การเติบโตของข้อมูลทำให้ประสิทธิภาพลดลง เมื่อจำนวน URL เพิ่มขึ้น การค้นหาและดึงข้อมูลจะช้าลง

Solution:

- ใช้ cache เพื่อลดการเข้าถึง database โดยเก็บข้อมูลที่เข้าถึงบ่อยไว้ในหน่วยความจำ
- สร้าง index ที่เหมาะสม เพื่อเพิ่มความเร็วในการค้นหาข้อมูล
- Sharding database ตามช่วงของ key เพื่อกระจายข้อมูลไปยังหลายเซิร์ฟเวอร์

2. High Traffic Handling

Concern: การรองรับผู้ใช้จำนวนมากพร้อมกัน ซึ่งอาจทำให้ระบบทำงานช้าลงหรือไม่สามารถให้บริการได้

Solution:

- ใช้ load balancer กระจายการทำงาน เพื่อแบ่งภาระงานไปยังหลายเซิร์ฟเวอร์
- Horizontal scaling ด้วย containerization เพื่อเพิ่มหรือลดจำนวนเซิร์ฟเวอร์ตามปริมาณการใช้งาน
- Implement rate limiting และ connection pooling เพื่อจัดการทรัพยากรระบบอย่างมีประสิทธิภาพ

3. Geographic Distribution

Concern: ความล่าช้าเนื่องจากระยะทางทางภูมิศาสตร์ ผู้ใช้ในภูมิภาคที่ห่างไกลจากเซิร์ฟเวอร์อาจพบความล่าช้าในการใช้งาน

Solution:

- ใช้ CDN (Content Delivery Network) สำหรับ static content เพื่อกระจายเนื้อหาไปยังเซิร์ฟเวอร์ทั่วโลก
- กระจาย servers ไปตามภูมิภาคต่างๆ เพื่อลดระยะทางระหว่างผู้ใช้และเซิร์ฟเวอร์
- เพิ่ม edge caching เพื่อเก็บข้อมูลที่เข้าถึงบ่อยไว้ใกล้กับผู้ใช้

4. Cleanup and Maintenance

Concern: URLs ที่หมดอายุสะสมจำนวนมาก ทำให้ฐานข้อมูลมีขนาดใหญ่เกินความจำเป็น

Solution:

- สร้าง background job หรือ Batch Job สำหรับลบ URLs ที่หมดอายุ โดยทำงานอัตโนมัติตามตารางเวลาที่กำหนด