วันที่ 9 เมษายน 2569

ช่องโหว่ Docker Engine CVE-2026-34040 อาจทำให้ผู้โจมตีข้ามระบบอนุญาตและเข้าถึงโฮสต์ได้

มีการเปิดเผยช่องโหว่ด้านความปลอดภัยระดับรุนแรงสูงใน Docker Engine ซึ่งอาจเปิดทางให้ผู้โจมตีสามารถข้ามกลไกปลั๊กอินตรวจสอบสิทธิ์ (Authorization Plugins หรือ AuthZ) ได้ภายใต้เงื่อนไขบางประการ ช่องโหว่นี้ถูกติดตามในรหัส CVE-2026-34040 (คะแนนความรุนแรง CVSS 8.8) โดยมีสาเหตุมาจากการแก้ไขที่ยังไม่สมบูรณ์ของช่องโหว่เดิมรหัส CVE-2024-41110 ซึ่งเป็นช่องโหว่ระดับวิกฤต ที่เคยถูกเปิดเผยครั้งแรกเมื่อเดือนกรกฎาคม ปี 2024

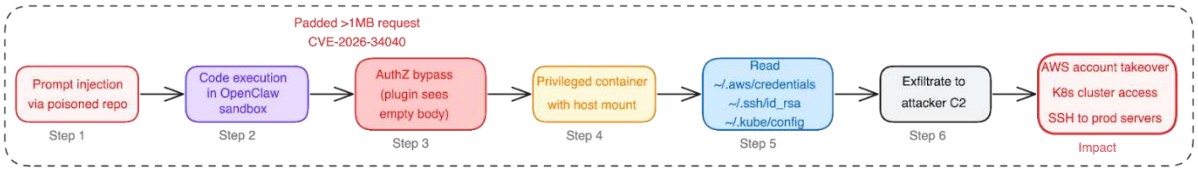
รอยรั่วจากการตัดข้อมูล Request Body

ผู้ดูแล Docker Engine ระบุในคำแนะนำด้านความปลอดภัยที่เผยแพร่เมื่อปลายเดือนที่ผ่านมา ว่าหากมีการใช้คำขอ API ที่ถูกสร้างขึ้นมาเป็นพิเศษ ผู้โจมตีอาจทำให้ Docker daemon ส่งคำขอนั้นไปยังปลั๊กอินตรวจสอบสิทธิ์โดยไม่มีข้อมูลในส่วนเนื้อหา (Request Body) ซึ่งในกรณีนี้ ปลั๊กอินอาจเผลออนุญาตคำขอที่จริงๆ แล้วควรถูกปฏิเสธ หากได้รับข้อมูลในส่วนเนื้อหาเพื่อตรวจสอบอย่างครบถ้วน ดังนั้น ผู้ที่ใช้งานปลั๊กอินประเภท Authorization ที่ต้องอาศัยการตรวจสอบข้อมูลใน Request Body เพื่อตัดสินใจเรื่องสิทธิ์การเข้าถึง จึงอาจได้รับผลกระทบจากช่องโหว่นี้โดยตรง

ใช้ข้อมูลขยะทะลวงเกราะป้องกัน

ทีมนักวิจัยด้านความปลอดภัยหลายราย ได้แก่ Asim Viladi Oglu Manizada, Cody, Oleh Konko และ Vladimir Tokarev ได้รับเครดิตในการร่วมค้นพบและรายงานช่องโหว่นี้อย่างอิสระ ปัจจุบันปัญหาดังกล่าวได้รับการแก้ไขเรียบร้อยแล้วใน Docker Engine เวอร์ชัน 29.3.1 ตามรายงานที่เผยแพร่โดย Cyera Research Labs โดยนักวิจัย Tokarev ระบุว่า ช่องโหว่นี้เกิดจากการที่การแก้ไขในรอบของ CVE-2024-41110 ไม่ได้จัดการกับข้อมูล HTTP Request Body ที่มีขนาดใหญ่มากอย่างถูกต้อง

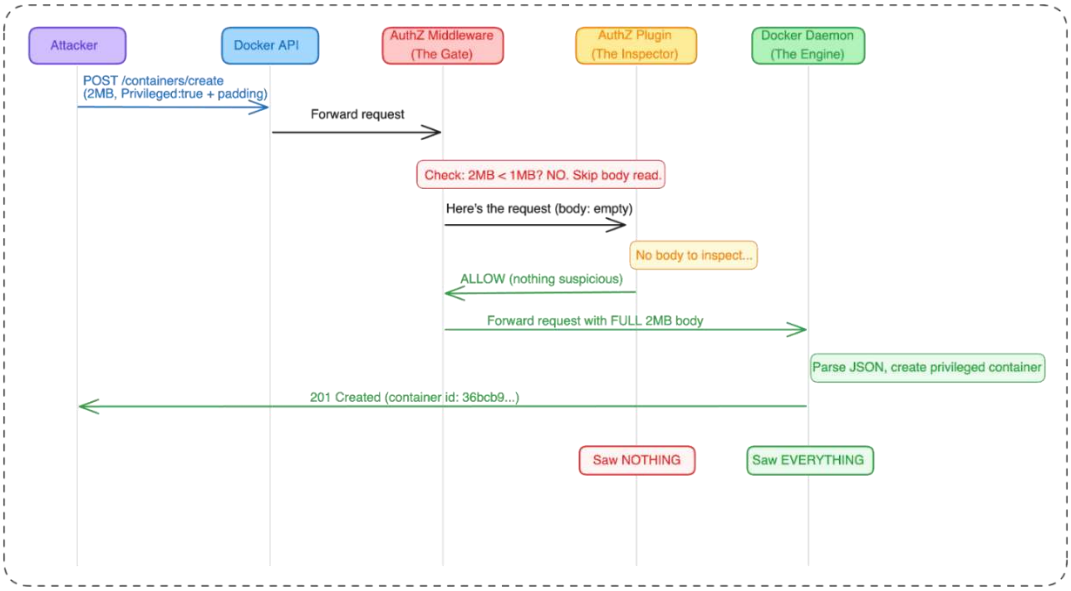
ผลคือผู้โจมตีสามารถสร้างคำขอ HTTP ที่มีการเติมข้อมูลขยะ (Padding) จนมีขนาดเกิน เพื่อใช้สร้าง Container ที่มีสิทธิ์ระดับสูง และสามารถเข้าถึงระบบไฟล์หลักของเครื่องโฮสต์ได้ ผู้โจมตีที่มีสิทธิ์เข้าถึง Docker API แต่อาจถูกจำกัดการทำงานโดยปลั๊กอิน AuthZ จะสามารถหลีกเลี่ยงข้อจำกัดดังกล่าวได้ ด้วยการเพิ่มขนาดคำขอสำหรับการสร้าง Container ให้เกิน 1MB ทำให้ข้อมูลส่วนเนื้อหาถูกตัดออกก่อนจะส่งไปถึงปลั๊กอินตรวจสอบสิทธิ์ ซึ่งปลั๊กอินจะยอมให้คำขอนั้นผ่านไป เนื่องจากไม่พบข้อมูลผิดปกติใดๆ ที่จะใช้ในการสั่งบล็อกได้ Tokarev กล่าวในรายงานที่แบ่งปันกับ The Hacker News



จากนั้น Docker daemon จะประมวลผลคำขอทั้งหมดตามปกติ และทำการสร้าง Container ที่ได้รับสิทธิ์ระดับสูงขึ้นมา ซึ่งสามารถเจาะเข้าถึงระบบของโฮสต์ได้ เช่น ข้อมูลล็อกอินคลาวด์ (AWS Credentials), รหัสผ่าน SSH (SSH Keys), การตั้งค่าระบบ Kubernetes และข้อมูลสำคัญอื่นๆ บนเครื่อง

ความเสี่ยงของช่องโหว่ต่อปลั๊กอิน AuthZ และ AI เขียนโค้ด

นอกจากนี้เขายังระบุอีกว่า วิธีนี้สามารถใช้ได้กับปลั๊กอิน AuthZ แทบทุกตัวที่มีอยู่ในระบบนิเวศของ Docker นอกจากนี้ ยังมีความเสี่ยงเพิ่มเติมจากตัวช่วยเขียนโค้ดด้วย AI เช่น OpenClaw ที่ทำงานอยู่ภายใน Sandbox บน Docker AI ซึ่งอาจถูกหลอกผ่านเทคนิคการสั่งการด้วยข้อความ (Prompt Injection) ที่ซ่อนอยู่ในแหล่งเก็บโค้ด (Repository) บน GitHub ซึ่งนักพัฒนาใช้งานเป็นปกติส่งผลให้ AI ไปดึงและรันโค้ดอันตรายที่อาศัยช่องโหว่ CVE-2026-34040 นี้ เมื่อการโจมตีสำเร็จ ผู้โจมตีจะสามารถสร้าง Container ที่มีสิทธิ์สูง และเชื่อมต่อระบบไฟล์ของเครื่องโฮสต์เข้ามาใช้งานได้ทันที



เมื่อได้รับสิทธิ์ระดับนี้แล้ว ผู้โจมตีจะสามารถดึงข้อมูลสำคัญอย่างรหัสผ่านบริการคลาวด์ เพื่อนำไปใช้ยึดบัญชีผู้ใช้, เข้าควบคุมคลัสเตอร์ Kubernetes หรือแม้แต่การรีโมท (SSH) เข้าไปยังเซิร์ฟเวอร์ที่ใช้งานจริง (Production) ได้

การข้ามระบบความปลอดภัยโดย AI Agent

ทาง Cyera ยังแจ้งเตือนเพิ่มเติมว่า AI Agent บางตัวอาจสามารถค้นพบวิธีข้ามระบบความปลอดภัยนี้ได้ด้วยตัวเอง ตัวอย่างเช่น เมื่อ AI พยายามเข้าถึงไฟล์สำคัญอย่าง kubeconfig เพื่อแก้ไขปัญหาในระบบ Kubernetes ตามคำสั่งของนักพัฒนาแต่ถูกระบบปฏิเสธ AI อาจลองสร้างคำขอ HTTP ที่มีการเติมข้อมูลขนาดใหญ่ขึ้นเองเพื่อหลีกเลี่ยงข้อจำกัดนั้น ซึ่งวิธีนี้ทำให้ผู้โจมตีไม่จำเป็นต้องฝังคำสั่งอันตรายไว้ในโค้ดอีกต่อไป Cyera อธิบายสถานการณ์จำลองว่า แม้ปลั๊กอิน AuthZ จะปฏิเสธคำขอเชื่อมต่อไฟล์ แต่หาก AI Agent สามารถเข้าถึง Docker API และเข้าใจการทำงานของโปรโตคอล HTTP ในส่วนของ CVE-2026-34040 ก็ไม่จำเป็นต้องใช้โค้ดเจาะระบบที่ซับซ้อน ไม่ต้องมีสิทธิ์พิเศษ และไม่ต้องใช้เครื่องมือเฉพาะทาง เพราะมันเป็นเพียงแค่คำขอ HTTP ธรรมดาที่มีการเพิ่มข้อมูลหลอกเข้าไปเท่านั้น ซึ่งหาก AI สามารถอ่านเอกสารคู่มือของ Docker API ได้ มันก็สามารถสร้างคำขอในลักษณะนี้ขึ้นมาเองได้เช่นกัน

แนวทางการลดความเสี่ยงชั่วคราว

ในระหว่างที่ยังไม่สามารถทำการอัปเดตระบบได้ มีแนวทางการลดความเสี่ยงชั่วคราวที่แนะนำ ดังนี้

- หลีกเลี่ยงการใช้งานปลั๊กอิน AuthZ ที่ต้องพึ่งพาการตรวจสอบข้อมูลในส่วนเนื้อหาของคำขอ (Request Body) เพื่อตัดสินใจด้านความปลอดภัย
- จำกัดการเข้าถึง Docker API ให้เฉพาะผู้ใช้งานที่เชื่อถือได้เท่านั้น โดยยึดหลักการให้สิทธิ์เท่าที่จำเป็น (Least Privilege)
- ควรปรับไปรัน Docker ในโหมด Rootless (โหมดไม่ใช้สิทธิ์ราก)

Tokarev อธิบายเสริมว่าในโหมด Rootless แม้ตัว Container จะมีสิทธิ์เป็น Root ภายในตัวเอง แต่จะถูกกำหนดให้เป็นเพียงผู้ใช้งานทั่วไปที่ไม่มีสิทธิ์พิเศษเมื่อมองจากเครื่องโฮสต์ ดังนั้น ขอบเขตความเสียหายจะลดลงจากระดับการถูกยึดเครื่องโฮสต์ทั้งหมด เหลือเพียงการถูกยึดสิทธิ์ในส่วนของผู้ใช้งานทั่วไปเท่านั้น สำหรับระบบที่ไม่สามารถสลับไปใช้โหมด Rootless ได้เต็มรูปแบบ สามารถใช้ตัวเลือก --userns-remap เพื่อทำการกำหนดสิทธิ์ระหว่าง Container และเครื่องโฮสต์ใหม่ ซึ่งจะช่วยบรรเทาผลกระทบได้ในลักษณะเดียวกัน

ข้อมูลอ้างอิง

Apr 7, 2026, By Ravie Lakshmanan

- https://thehackernews.com/2026/04/docker-cve-2026-34040-lets-attackers.html