การประเมินและปรับปรุงระบบสืบค้น MITRE ATT&CK

จากชุดข้อมูลสังเคราะห์สู่ CTI จริง และการแก้ปัญหาการจัดอันดับ
branch `eval/real-cti-dataset` · ปรับปรุงล่าสุด 13 สิงหาคม 2569

สรุปสำหรับผู้บริหาร

งานนี้เริ่มจากข้อทักท้วงของอาจารย์ที่ปรึกษาว่า การใช้ LLM สร้างชุดข้อมูลประเมินผลเองนั้น ไม่น่าเชื่อถือ จึงได้สร้างชุดข้อมูลใหม่จากรายงานภัยคุกคามสาธารณะที่มีผู้เชี่ยวชาญกำกับ แล้วนำมาแปลเป็นสำนวนคดีภาษาไทยจำนวน 100 สำนวน

ระหว่างการประเมินพบข้อผิดพลาดในเครื่องมือวัดสองจุด ซึ่งทำให้คะแนนต่ำกว่าความเป็นจริงมาก และเมื่อแก้ไขแล้วจึงพบว่าจุดอ่อนที่แท้จริงอยู่ที่ การจัดลำดับผลลัพธ์ ไม่ใช่ความสามารถในการค้นหา การแก้ไขวิธีเรียงลำดับเพียงจุดเดียวทำให้ตัวชี้วัดด้านอันดับดีขึ้นกว่าเท่าตัว โดยไม่มีตัวชี้วัดใดแย่ลง

ตัวชี้วัด	ก่อนแก้	หลังแก้	เปลี่ยนแปลง
MRR (อันดับของคำตอบถูกตัวแรก)	0.262	0.554	+111%
Hit@1 (อันดับ 1 ถูกหรือไม่)	0.050	0.460	+820%
NDCG@15	0.292	0.423	+45%
StepCoverage@15 (ความครอบคลุม)	0.511	0.528	+3%

1. ที่มาของปัญหา

ชุดข้อมูลประเมินผลเดิมสร้างโดยให้ LLM แต่งสำนวนคดีขึ้นจากชื่อเทคนิค ATT&CK ที่สุ่มมาจากฐานข้อมูลกราฟของระบบเอง วิธีนี้มีปัญหาสองชั้นซ้อนกัน

ชั้นแรก — คำถามสืบทอดคำศัพท์จากคำตอบ เนื่องจาก LLM เห็นชื่อเทคนิคก่อนเขียนโจทย์ คำที่ปรากฏในสำนวนจึงใกล้เคียงกับชื่อเทคนิคปลายทาง ระบบสืบค้นเพียงจับคู่คำก็ตอบได้ โดยไม่จำเป็นต้องเข้าใจพฤติกรรมที่บรรยาย

ชั้นที่สอง — วนกลับมาที่ตัวเอง เทคนิคที่ใช้เป็นเฉลยถูกสุ่มจากฐานข้อมูลกราฟชุดเดียวกับที่ ระบบใช้ค้นหา จึงรับประกันล่วงหน้าว่าคำตอบอยู่ในคลังเสมอ ซึ่งไม่ตรงกับการใช้งานจริง

2. ชุดข้อมูลใหม่จาก CTI จริง

เลือกแหล่งข้อมูลที่มีการกำกับเทคนิค ATT&CK โดยผู้เชี่ยวชาญภายนอก และเผยแพร่สาธารณะพร้อมสัญญาอนุญาตชัดเจน

แหล่งข้อมูล	ลักษณะ	ได้มา	สัญญาอนุญาต
CTID Adversary Emulation Library	แผนจำลองการโจมตี มีลำดับขั้นตอนและเทคนิคกำกับรายขั้น	80 ห่วงโซ่	Apache 2.0

CISA Cybersecurity Advisories	รายงานเหตุการณ์จริง มีการกำกับเทคนิคในเนื้อความ	55 ห่วงโซ่	CC-BY-4.0
รวมหลังคัดเลือก	กระจายทั่ว 32 เอกสารต้นทาง	100 ห่วงโซ่ / 347 ขั้นตอน	—

จุดสำคัญของ CISA คือส่วน TECHNICAL DETAILS กำกับรหัสเทคนิคไว้ท้ายประโยคที่บรรยายพฤติกรรม เรียงตามลำดับเวลา เมื่อตัดรหัสออก ประโยคที่เหลือคือคำบรรยายเชิงสังเกตที่ใช้เป็นโจทย์ได้ทันที จึงได้ทั้งลำดับขั้นตอนและเฉลยจากตัวบทเดียวกัน

การแปลเป็นสำนวนคดีไทย

รายงาน CTI เขียนด้วยภาษานักวิเคราะห์ซึ่งระบุชื่อเทคนิคไว้ในตัวบทอยู่แล้ว หากแปลตรงตัวจะเท่ากับใส่คำตอบลงในคำถาม จึงต้องถอดใหม่เป็นมุมมองของผู้สังเกตการณ์ เช่นเดียวกับที่ปรากฏในสำนวนคดีจริง

ต้นฉบับ (CISA)	สำนวนไทยที่ใช้เป็นโจทย์
the actors use open source brute force tools to gain access [T1110]	คนร้ายล็อกอินเข้ามาจากภายนอก ผ่านช่องทางเชื่อมต่อระยะไกล โดยใช้เพียงชื่อผู้ใช้และรหัสผ่านชั้นเดียว

ทุกสำนวนผ่านเครื่องตรวจอัตโนมัติสี่ข้อ ได้แก่ ข้อความอ้างอิงต้องปรากฏตรงตัวอักษรในสำนวน ห้ามมีรหัส ATT&CK หลุดในตัวบท ข้อความแบบบรรยายพฤติกรรมห้ามมีชื่อเทคนิคภาษาอังกฤษ และเฉลยต้องตรงกับต้นทางเสมอ โดยระบบคัดลอกเฉลยให้เองไม่ได้พิมพ์ด้วยมือ

ผลคือ 347 ขั้นตอน แบ่งเป็นแบบบรรยายพฤติกรรม 286 ขั้นตอน และแบบระบุชื่อเทคนิค 61 ขั้นตอน สัดส่วนนี้สะท้อนสำนวนคดีจริงซึ่งบรรยายสิ่งที่พบมากกว่าระบุศัพท์เทคนิค

3. ข้อผิดพลาดของเครื่องมือวัดที่พบระหว่างทาง

ก่อนจะสรุปว่าระบบทำงานได้ดีเพียงใด พบว่าเครื่องมือวัดเองมีข้อผิดพลาดสองจุด
ซึ่งกดคะแนนให้ต่ำกว่าความเป็นจริงอย่างมีนัยสำคัญ

3.1 เทียบเฉลยกับคำตอบคนละระดับชั้น

ATT&CK มีโครงสร้างสองระดับ คือเทคนิคหลักและเทคนิคย่อย ฐานข้อมูลกราฟของระบบเก็บเฉพาะเทคนิคหลัก 299 รายการ ขณะที่ฐานข้อมูลเวกเตอร์เก็บทั้งสองระดับรวม 2,195 รายการ

เมื่อสร้างชุดข้อมูล ได้ยุบเฉลยจากเทคนิคย่อยขึ้นเป็นเทคนิคหลักแล้ว แต่ไม่ได้ยุบฝั่งคำตอบด้วย
ผลคือเมื่อระบบตอบเทคนิคย่อยที่ถูกต้องและละเอียดกว่า กลับถูกนับว่าผิด

```
gold  = T1053         (           )
        = T1053.005     (                    )
       →        =
        →         T1053           =
```

การแก้ไขคือยุบรหัสฝั่งคำตอบขึ้นเป็นเทคนิคหลักก่อนเทียบ โดยใช้วิธีแทนที่แล้วตัดตัวซ้ำ เพื่อให้หนึ่งผลลัพธ์ยังคงกินหนึ่งอันดับ ตัวชี้วัดที่ระดับ K จึงยังมีความหมายเท่าเดิม นี่ไม่ใช่การเข้าข้างระบบ แต่เป็นการทำให้ทั้งสองฝั่งอยู่หน่วยเดียวกัน

ผลกระทบ (เส้นทาง production, k=15)	ก่อนแก้	หลังแก้
ความครอบคลุมแบบระบุชื่อเทคนิค	0.568	0.833
ความครอบคลุมแบบบรรยายพฤติกรรม	0.434	0.452
ความครอบคลุมรวม	0.449	0.511

3.2 คะแนนของตัวจัดอันดับถูกบีบช่วง

ฟังก์ชัน predict ของ cross-encoder ผ่านฟังก์ชัน sigmoid ให้แล้ว แต่โค้ดนำไปผ่าน sigmoid ซ้ำอีกครั้ง
ทำให้ช่วงคะแนนทั้งหมดถูกบีบจาก 0 ถึง 1 เหลือเพียง 0.500 ถึง 0.731

ลำดับไม่เปลี่ยน เพราะ sigmoid เป็นฟังก์ชันทางเดียว แต่ค่าเกณฑ์ที่ใช้คัดกรองตารางผลลัพธ์ปลายทาง
ถูกตั้งบนสเกลที่ผิดนี้ และเป็นที่มาของหมายเหตุในไฟล์ตั้งค่าที่ระบุว่า ตัวจัดอันดับอีมตัวที่ 0.500

หมายเหตุ ยังไม่ได้ปรับค่าเกณฑ์ใหม่ เนื่องจากกระทบพฤติกรรมของระบบที่ให้บริการอยู่ และควรทำเป็นการทดลองแยกต่างหาก

4. วิธีเดิม การรวมผลแบบวนรอบตามลำดับ

เส้นทางที่ระบบใช้จริงจะแตกสำนวนคดีออกเป็นคำถามย่อยรายเทคนิคด้วย LLM แล้วค้นทีละคำถาม จากนั้นนำผลมารวมกันแบบวนรอบ เพื่อให้ทุกคำถามย่อยมีที่นั่งในผลลัพธ์สุดท้าย ไม่ถูกตัดทิ้งทั้งกลุ่ม

```
for rank in range(depth):
    for vecs in per_query_vectors:      # ←               decompose
        merged_vector.append(vecs[rank])
```

แนวคิดเรื่องการันตีที่นั่งนั้นถูกต้อง ปัญหาอยู่ที่ลำดับการหยิบภายในรอบเดียวกัน ซึ่งเดินตามลำดับที่ตัวแตกคำถามส่งออกมา ลำดับนั้นไม่ได้บอกอะไรเกี่ยวกับความมั่นใจของผลลัพธ์เลย อันดับ 1 จึงเป็นผลของคำถามย่อยตัวแรกเสมอ ไม่ว่าคะแนนจะสูงหรือต่ำ

ตัวอย่างจริงจากการติดตามการทำงาน

สำนวนคดีธนาคาร กรณี rcti_009 ถูกแตกออกเป็นคำถามย่อย 7 ข้อ ผลลัพธ์ที่ได้เป็นดังนี้

อันดับ	ชนิด	ชื่อ	คะแนน	เป็นเฉลย
1	Relationship	Axiom	0.2491	
2	Technique	Exploitation for Privilege Escalation	0.7249	
3	Technique	OS Credential Dumping	0.0475	ใช่
4	Tactic	Credential Access	0.9322	
5	Technique	Ingress Tool Transfer	0.1510	ใช่
6	Technique	Lateral Tool Transfer	0.9903	ใช่

ผลลัพธ์ที่ตรงที่สุดในรายการได้คะแนน 0.9903 แต่ไปอยู่อันดับ 6 ขณะที่อันดับ 1 ได้เพียง 0.2491 คำตอบถูกตัวแรกจึงไปปรากฏที่อันดับ 3 ทั้งที่ควรอยู่อันดับ 1

อาการนี้อธิบายผลการวัดในภาพรวมได้ตรงกัน กล่าวคือเส้นทางที่มีการแตกคำถามมีความครอบคลุมสูงกว่ามาก แต่กลับมีค่า MRR ต่ำกว่าการค้นด้วยเวกเตอร์อย่างเดียว เพราะหาของเจอแต่จัดอันดับไม่ดี

5. วิธีใหม่ เรียงตามคะแนนภายในรอบ

แก้เฉพาะลำดับการหยิบภายในรอบเดียวกัน โดยเรียงตามคะแนนก่อนปล่อยออก
ทุกคำถามย่อยยังได้ที่นั่งในรอบแรกเหมือนเดิม และรายการที่รอดการตัดยังเป็นชุดเดิม เปลี่ยนเพียงลำดับ

```
for rank in range(depth):
    round_hits = [vecs[rank] for vecs in per_query_vectors if rank < len(vecs)]
    round_hits.sort(key=lambda vr: vr.score, reverse=True)   # ←
    for vr in round_hits:
        merged_vector.append(vr)
```

ข้อจำกัดที่ต้องระบุ คะแนนของแต่ละผลลัพธ์มาจากการให้คะแนนเทียบกับคำถามย่อยคนละข้อ
จึงไม่สามารถเทียบกันได้อย่างเคร่งครัดในเชิงทฤษฎี วิธีนี้จึงเป็นการประมาณลำดับที่ดีกว่าเดิม ไม่ใช่การจัดอันดับที่ถูกต้องสมบูรณ์
ได้บันทึกข้อจำกัดนี้ไว้ในโค้ดแล้ว

6. ผลการวัด

ประเมินบนสำนวนคดีไทย 100 สำนวน ผ่านเส้นทางเดียวกับที่ระบบให้บริการจริง ใช้แบบจำลอง Claude Haiku
ในการแตกคำถาม เปรียบเทียบกับผลก่อนแก้ที่วัดด้วยเครื่องมือที่แก้ไขแล้วเช่นกัน

ตรวจสอบว่าเงื่อนไขการทดลองเทียบเคียงกันได้

เงื่อนไข	ก่อนแก้	หลังแก้
จำนวนคำถามย่อยเฉลี่ยต่อสำนวน	7.20	7.13
จำนวนผลลัพธ์เวกเตอร์ที่รวมได้	14.92	14.94
จำนวนกราฟย่อย	5.47	5.49
จำนวนครั้งที่การแตกคำถามล้มเหลว	0	0

ตัวชี้วัดด้านการจัดอันดับ

ตัวชี้วัด	ก่อนแก้	หลังแก้	ผลต่าง
MRR	0.262	0.554	+0.292
MAP	0.174	0.313	+0.139
Hit@1	0.050	0.460	+0.410
NDCG@5	0.172	0.338	+0.166
NDCG@15	0.292	0.423	+0.131

ตัวชี้วัดด้านความครอบคลุมขั้นตอน

ตัวชี้วัด	ก่อนแก้	หลังแก้	ผลต่าง
StepCoverage@1	0.024	0.143	+0.119
StepCoverage@5	0.235	0.320	+0.085
StepCoverage@15	0.511	0.528	+0.017

StepCoverage@50	0.646	0.661	+0.015
แบบบรรยายพฤติกรรม @15	0.452	0.467	+0.015
แบบระบุชื่อเทคนิค @15	0.833	0.872	+0.039

ไม่มีตัวชี้วัดใดแย่ลง ระหว่างการติดตามการทำงานรายกรณีเคยพบว่ามีเฉลยหนึ่งตัวหลุดออกจาก 15 อันดับแรก จึงไม่สรุปจากกรณีเดียวและวัดใหม่ทั้งชุด ผลในภาพรวมพบว่าความครอบคลุมเพิ่มขึ้นทีทุกค่า K

7. การเปรียบเทียบสถาปัตยกรรมการค้นหา

วัดบนชุดข้อมูลเดียวกัน เพื่อยืนยันว่าความซับซ้อนของระบบให้ผลตอบแทนจริง (ตัวเลขก่อนการแก้ไขในหัวข้อที่ 5)

สถาปัตยกรรม	Hit@5	NDCG@5	MRR	StepCov@15	เวลา
เวกเตอร์อย่างเดียว	0.630	0.235	0.478	0.312	0.6 วิ
กราฟอย่างเดียว (คำสำคัญ)	0.110	0.029	0.067	0.081	0.3 วิ
ผสมเวกเตอร์กับกราฟ	0.580	0.147	0.240	0.346	3.6 วิ
ผสม พร้อมแตกคำถาม (ระบบจริง)	0.560	0.172	0.262	0.511	112 วิ

การแตกคำถามให้ความครอบคลุมสูงสุดอย่างชัดเจน ส่วนการค้นด้วยเวกเตอร์อย่างเดียวมีอันดับดีที่สุด ซึ่งเป็นที่มาของการแก้ไขในหัวข้อที่ 5 ส่วนการค้นบนกราฟด้วยคำสำคัญได้ผลเกือบเป็นศูนย์ เนื่องจากใช้การจับคู่สตริงภาษาอังกฤษกับสำนวนภาษาไทย

8. ข้อค้นพบเพิ่มเติมและข้อจำกัด

8.1 ไม่ใช่ปัญหาข้ามภาษา

ทดลองเปรียบเทียบแบบจำลองฝังข้อมูลสามแบบ พบว่าเมื่อยิงคำถามภาษาอังกฤษที่มีเฉลยชุดเดียวกัน คะแนนต่างจากภาษาไทยเพียง 0.02 ถึง 0.04 เท่านั้น จึงตัดสมมติฐานว่าแบบจำลองอ่านภาษาไทยไม่ได้ออกไป ปัญหาอยู่ที่การจับคู่พฤติกรรมกับเทคนิคซึ่งยากในทุกภาษา

นอกจากนี้ยังพบว่าการใช้เวกเตอร์แบบหนาแน่นอย่างเดียวให้ผลดีกว่าการผสมกับเวกเตอร์แบบกระจาย ซึ่งเป็นค่าที่ระบบใช้อยู่ ต่างกัน 0.034 ในค่า NDCG@5 และยังไม่ได้นำไปแก้ไข

8.2 ข้อจำกัดของตัวชี้วัด precision

แต่ละสำนวนมีเฉลยเฉลี่ยเพียง 3.59 รายการ ขณะที่วัดที่ 15 อันดับ เพดานทางคณิตศาสตร์ของ precision จึงอยู่ที่ 0.239 ค่าที่วัดได้ 0.116 คิดเป็นร้อยละ 48 ของเพดาน ตัวชี้วัดนี้จึงไม่เหมาะกับงานลักษณะนี้ ควรใช้ความครอบคลุมและ MRR เป็นหลัก

8.3 งานที่ยังค้าง

รายการ	สถานะ
ตรวจทานสำนวนไทยทั้ง 100 สำนวนโดยผู้เชี่ยวชาญ	ยังไม่ดำเนินการ
ปรับค่าเกณฑ์คัดกรองตารางผลลัพธ์ใหม่หลังแก้สเกลคะแนน	ยังไม่ดำเนินการ
สลับผลจากกราฟเข้ามาแทนการต่อท้าย	ยังไม่ดำเนินการ
ประเมินคุณภาพคำตอบที่ระบบสร้าง	รอตรวจสำนวนก่อน
เทคนิค T1562 หายจากฐานข้อมูลกราฟ	ตรวจพบแล้ว ยังไม่แก้

หมายเหตุเรื่องเวลาประมวลผล การวัดครั้งหลังใช้เวลาเฉลี่ย 14.2 วินาทีต่อสำนวน เทียบกับ 112.4 วินาที ในการวัดครั้งก่อน การเรียงลำดับไม่สามารถอธิบายส่วนต่างนี้ได้ สาเหตุน่าจะมาจากการวัดครั้งก่อน รันสถาปัตยกรรมสี่แบบต่อกันในกระบวนการเดียวบนหน่วยความจำกราฟิกที่จำกัด จึงไม่นับเป็นผลของการแก้ไขครั้งนี้