תבניות עיצוב לחיבור AI Agents לפייפליין סטרימינג

AWS Summit Tel Aviv 2026 ,סיכום סשן — ANT302

עדי אבני ואופר דוברובסקי, Solutions Architects, AWS | אלחנן, Senior Staff AI Scientist, Intuit

10 בספטמבר 2026 | משך: כ-42 דקות | הופק מהקלטת הסשן

סיכום מאת קובי אלמוג

על המסמך הזה

המסמך מסכם את הסשן ANT302 מתוך AWS Summit Tel Aviv 2026, במסלול Data & Analytics. הסשן חולק לשלושה חלקים: תבניות עיצוב לחיבור AI agents לפייפליין סטרימינג (AWS), דוגמה מלאה מקצה לקצה על חברת משלוחים פיקטיבית (AWS), ומקרה אמיתי מ-Intuit בתחום מניעת השתלטות על חשבונות. חותמות הזמן בגוף המסמך מפנות לנקודה המדויקת בהקלטה.

איך לקרוא את זה

- **פרק 1 — תקציר מנהלים:** חמש דקות קריאה. מה נטען בסשן ומה שווה לבדוק בארגון.
- **פרקים 2–4 — התבניות:** שכבת הדאטה, המעבר מאג'נט בודד ל-multi-agent, וחיבור לארכיטקטורות EDA קיימות.
- **פרקים 5–6 — הדוגמה:** תרחיש צי המשלוחים, מהאלרט ועד הפעולות שהאג'נט ביצע, וההשוואה לפני ואחרי.
- **פרקים 7–9 — Intuit:** עולם ה-Account Takeover, הארכיטקטורה בפועל, האתגרים והתוצאות.
- **פרקים 10–11 — הסיכום:** מה לוקחים מכאן, נקודות להמשך ומונחון.

הערת מקור המסמך הופק מתמלול אוטומטי של ההקלטה. התמלול מדויק בקטעים המקצועיים, אך שמות ומונחים לועזיים מופיעים בו בשיבוש והושבו לצורתם האנגלית. הציטוטים אומתו מול ההקלטה בחותמת הזמן המצוינת, ומספרים צוינו רק כשהופיעו על סלייד או נאמרו במפורש. שם המשפחה של הדובר מ-Intuit הופיע בתמלול בשיבוש ולכן לא צוין. בחלק הראשון של הסשן לא נלכדו סליידים בהקלטה, ולכן הוא מתואר בטקסט בלבד.

1. תקציר מנהלים

הטענה המרכזית של הסשן היא שהמידע הדרוש לקבלת החלטות מורכבות כבר זורם בפייפליין הסטרימינג של רוב הארגונים, ומה שחסר הוא שכבת אינטליג'נס שתדע לפעול עליו. שלושת הדוברים לא הציגו ארכיטקטורה חדשה אלא נקודת חיבור: להשאיר את הצנרת הקיימת כמות שהיא, ולהכניס אג'נטים רק אחרי סינון, בנקודות שבהן ההחלטה באמת דורשת שיקול דעת.

- **האג'נט נכנס אחרי הפילטר, לא לפניו.** הכלל שחזר בכל שלושת החלקים: קודם stream processing עם חוקים או מודל מסורתי, ורק על מה שעבר את הסף מפעילים LLM. "הפעלנו את ה-LLM או את החלק האג'נטי, אך ורק אחרי שביצענו איזשהו פילטרינג בסטרים פרוססינג" [12:18].
- **פיצול לאג'נטים קטנים הוא החלטה תפעולית.** הנימוק שניתן למעבר מאג'נט בודד ל-multi-agent אינו איכות התשובה אלא סקיילינג נפרד, בידוד תקלות ו-prompt ממוקד לכל שלב [10:42].
- **העלות נמוכה כי הקריאות מעטות.** בדוגמת צי המשלוחים רק מיעוט ההודעות מגיע לאג'נטים — "אנחנו נקרא להם אולי מאות פעמים ביום" [22:56] — בעוד הערך מגיע ממניעת נזק של מאות עד אלפי דולרים לאירוע.
- **Intuit מדווחת על שיפור של פי 64 בזמן טיפול.** בחלק מהתהליכים בחקירת השתלטות על חשבונות, לצד יכולת חסימה בזמן אמת [40:00]. המערכות הקיימות — Kafka, Flink, מודל SageMaker — לא הוחלפו.
- **מה שחוזר על עצמו צריך לחזור לקוד.** שני דוברים נפרדים הציגו לולאת משוב: זיהוי דפוסים דטרמיניסטיים אצל האג'נט והמרתם לחוקים או לקוד, כדי לחסוך טוקנים ולטנסי [12:18], [38:26].

רלוונטיות לארגון פיננסי

שני מתוך שלושת המקרים שהוצגו הם מקרי הונאה: זיהוי אנומליות בעסקאות והשתלטות על חשבונות. התבנית זהה בשניהם — מערכת חוקים או מודל מסמנים חשד בזמן אמת, והאג'נט נכנס רק בשלב ההסקה וההמלצה לחוקר האנושי, שהוא צוואר הבקבוק האמיתי. זהו בדיוק המבנה של מערכות ניטור בבנק, כולל האילוץ שמדובר בתוספת למערכת קיימת ולא בהחלפה שלה.

2. הבסיס: שכבת הדאטה

עדי אבני, מנהל צוות Solutions Architects ב-AWS, פתח בסקירת use cases שבהם דאטה בזמן אמת פוגש אג'נטים: ניטור מכונות בייצור שמזהה תקלה לפני שהיא קורית, חוויית לקוח שמזהה כשל בכניסה לאפליקציה עוד לפני שהלקוח פונה, אוטומציה של משימות, ואנליטיקה שבה אג'נט מנתח דשבורד לפני שאדם מסתכל עליו [1:32]. המשותף לכולם, לדבריו, הוא הצורך ב-Data Foundation חזק לפני שמפעילים אג'נטים.

שכבת הדאטה חולקה לשלוש: Amazon MSK — Stream Storage ו-Amazon Kinesis Data Streams, שקולטים אירועים בקצב גבוה ומאפשרים replay; Movement — Amazon Data Firehose ו-MSK Connect, הצנרת אל ה-Data Lake; ו-Processing — עיבוד זרם שמייצר פיצ'רים, שומר state בחלונות זמן ומבצע deduplication, כדי שאפשר יהיה להריץ מעליו analytics ו-AI [3:04].

3. התבנית הבסיסית: אנומליה ואז אג'נט

התבנית הראשונה שהוצגה היא זיהוי הונאות. האירועים נכנסים ל-Kinesis או ל-MSK ועוברים ל-Amazon Managed Service for Apache Flink, שמריץ עליהם זיהוי אנומליות מבוסס חוקים — למשל חמישים עסקאות מגאו-לוקיישנים שונים בחלון זמן קצוב, או חריגה בסכומי העסקה [4:34]. רק מה שסומן כאנומליה ממשיך לשכבה האג'נטית.

האג'נט מופעל על Amazon Bedrock AgentCore. עדי מנה את הרכיבים: runtime מנוהל להרצת האג'נטים בפרודקשן בלי לנהל תשתית, memory לטווח קצר וארוך, שימוש בכלים כמו browser use, identity ו-policies שמגדירים מה מותר לאג'נט לעשות, guardrails, ומעל הכול observability ו-evaluation [6:05]. מודל החיוב הוא לפי שימוש בפועל.

האג'נט שנקרא Anomaly Response Agent קורא את אירועי האנומליה מ-Flink, מבצע קורלציה מול data points ו-knowledge bases, בודק דפוסים היסטוריים כדי להבין אם המקרה מוכר או חדש, ומכריע אם מדובר בהונאה ומה תוכנית הפעולה — קריאה ל-API, פנייה לחוקר או חסימת גישה [7:37].

מתי אג'נט אחד כבר לא מספיק

הנימוק לפיצול היה מעשי: כל אחת מהמשימות דורשת גישה למערכת חיצונית אחרת, עם policy משלה, "וגם זה מייצר לנו איזשהו פרומפט שהולך ומסתבך ונהיה יותר מורכב בגלל שיש לנו מגוון רחב של פעולות באותו קונטקסט" [7:37]. במצב כזה מפצלים לאג'נטים עם סקופ צר ואחריות מוגדרת.

אג'נט	אחריות	מקורות מידע
Anomaly Interpreter	מקבל את אירוע האנומליה מ-Flink ומבין מה קרה ומה ההיסטוריה שלו	אירועי הסטרים
Risk Scoring	נותן ציון להערכת הסיכון — האם זו הונאה ובאיזו סבירות	data points ודפוסים היסטוריים
Response	מקבל את כל הפענוח הקודם ומחליט על הפעולה	פלט שני האג'נטים הקודמים

בארכיטקטורה שהוצגה, Flink מזהה אנומליה ושולח ל-SQS; Lambda מטריגרת את האג'נט הראשון שרץ על AgentCore, ומשם דרך תור נוסף ל-Risk Scoring ולבסוף ל-Response [9:11]. הרווח מהפיצול, בלשונו: "אנחנו יכולים לבודד את הבעיה ולעשות דיבאגינג לכל אחד מהאג'נטים לפי הצורך" [10:42], לצד סקיילינג נפרד ו-business logic ממוקד יותר לכל חלק.

4. חיבור למערכות Event Driven קיימות

החלק המעשי ביותר בפרק התבניות עסק במערכות שכבר קיימות. בארגון טיפוסי יש Kafka topics או תורים שמזינים מיקרו-שירותים עם business logic. הדרך להוסיף אינטליג'נס אינה לשכתב אלא לצרוך את אותן הודעות מאותם topics בצרכן נוסף, להוסיף agentic services, ולהריץ אותם במקביל לקיים או במקומו בהדרגה: "זה מאפשר לנו לעבור ל-agentic Event Driven Architecture, בלי לעשות זעזועים משמעותיים במערכת, ולעשות את זה בצורה מדורגת והדרגתית" [10:42].

שיקולים שהוצגו

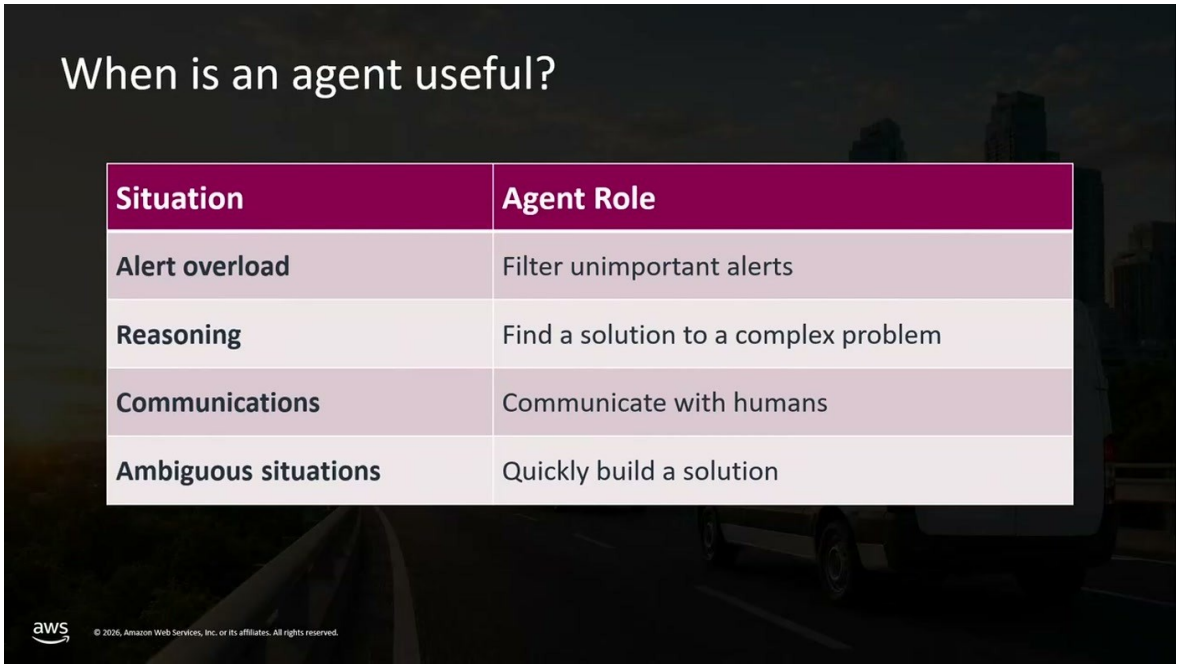
- **Latency מול אינטליג'נס:** האם לוקחים את ה-LLM המתקדם ביותר, או שלטנסי חשוב יותר ואפשר להסתפק בחוקים או במודל מסורתי [10:42].
- **פילטור לפני הפעלה:** הפעלת החלק האג'נטי רק על מה שעבר סינון בזרם — חיסכון בטוקנים וב-latency [12:18].
- **לולאת משוב לחוקים:** זיהוי דפוסים דטרמיניסטיים שהאג'נט מגלה, והחזרתם כחוקים לחלק ההתחלתי של המערכת [12:18].
- **Caching:** כשתשובות חוזרות על עצמן — חיסכון בעלות ובזמן תגובה [12:18].
- **Decoupling והרצה מקבילית:** SQS בין האג'נטים, והפעלה מקבילית היכן שאפשר [12:18].
- **Observability ו-cost awareness:** מעקב שוטף אחרי צריכת הטוקנים והעלות, כיכולת built-in בפלטפורמה [12:18].

5. הדוגמה: צי משלוחים

החברה הפיקטיבית שעליה נבנתה הדוגמה: אלפי רכבי משלוחים עם טלמטריית IoT

אופר דוברובסקי, Senior Solutions Architect ב-AWS, לקח את התבניות והמחיש אותן על חברה בדיונית בשם AnyDelivery: אלפי רכבים, מערכת סטרימינג שאוספת IoT מכל רכב, דאטהבייס של הזמנות, ואפליקציה לנהגים [13:48].

הפייפליין הקיים קלאסי: כל רכב שולח הודעת IoT כל 30 שניות דרך AWS IoT Core אל Kinesis Data Streams; זרם שני נושא את הודעות הנהגים; Flink מסתכל על חלונות זמן, מושך מהדאטהבייס נתוני משלוחים וכללי geofence, ומייצר אלרטים שיוצאים דרך Amazon SNS אל הסדרנים [15:19]. חלק מהאלרטים אינם דורשים כלום, וחלק דורשים התערבות מורכבת.



ארבע הסיטואציות שבהן אג'נט משתלם, והתפקיד שהוא ממלא בכל אחת

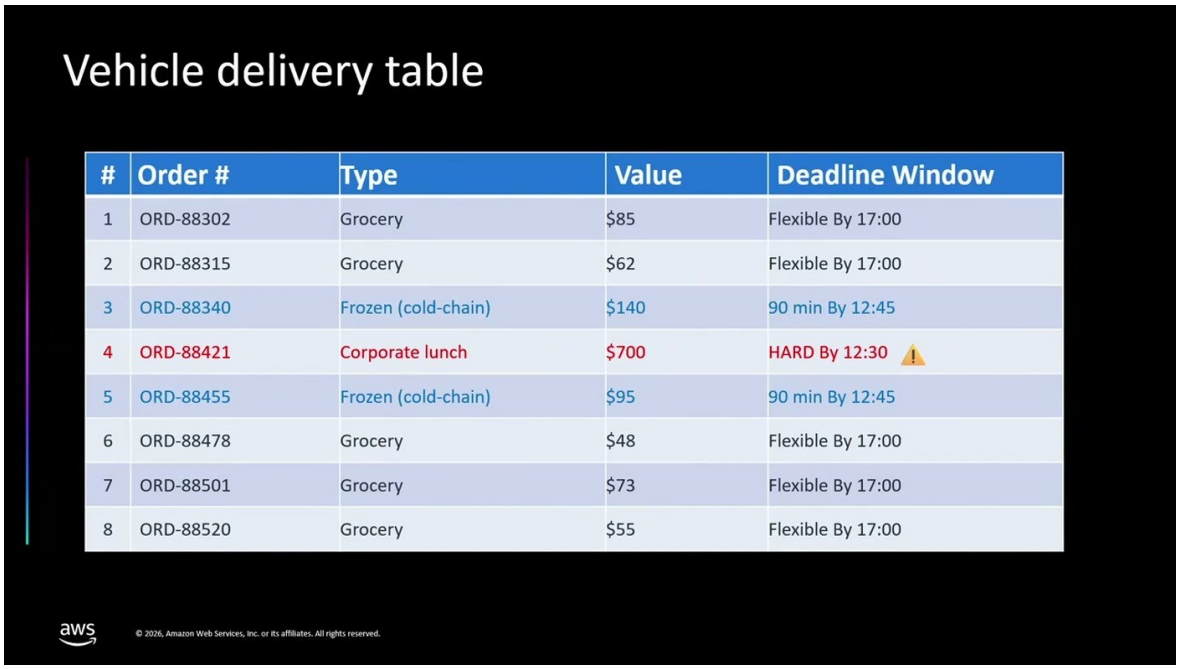
הסלייד ממפה ארבעה מצבים: עומס אלרטים, שבו תפקיד האג'נט לסנן את הלא-חשובים; reasoning, למציאת פתרון לבעיה מורכבת; תקשורת עם בני אדם; ומצבים מעורפלים שקשה לכתוב להם אוטומציה. על הראשון נאמר: "בהרבה

מקרים כאלה אנחנו מוצאים שהאג'נט יכול להקטין ב-95% את הרעש ולתת לאנשים להתמקד בדברים החשובים"
[15:19].

6. התרחיש: נהג עם תקר

נקודת הפתיחה של התרחיש שהודגם על הבמה

נהג A נתקע עם תקר. אחרי שמונה דקות, Flink מזהה 16 הודעות IoT מאותו מיקום, מצליב מול הדאטהבייס ורואה שהמיקום אינו כתובת מסירה — ומוציא geofence alert: רכב סטטי, לא כתובת מסירה. אחרי שתי דקות נוספות מגיעה הודעת הנהג: תקר, נקרא שירות דרך, הבטיחו להגיע תוך 45 דקות. השעה 11:32 [16:50].



שמונה ההזמנות שעל הרכב, ממוינות לפי דחיפות — השורה האדומה היא ארוחת הצהריים להנהלה

הטבלה היא הלב של הדוגמה. על הרכב שמונה הזמנות. אחת מהן, ORD-88421, היא ארוחת צהריים לישיבת הנהלה בשווי 700 דולר עם hard stop ב-12:30 — ותוכננה להגיע ב-12:00. שתי הזמנות נוספות דורשות שרשרת קירור עם חלון של 90 דקות. השאר גמישות עד 17:00. בשעה 11:32, במצב הקיים, אין סיכוי שההזמנה הקריטית תגיע [18:20].

Flink מייצר enriched message: לוקח את האלרט הקודם, מוסיף את הודעת הנהג, מושך את ההזמנות מהדאטהבייס, ומסמן שיש time critical delivery בשווי 700 דולר [18:20]. ואז מגיעה הנקודה שאופר ביקש לשים

לב אליה: "בנקודה הזאת, המערכת כבר יודעת שההזמנה הקריטית תאחר, וגם ההזמנות הקרות" [18:20]. כל הנתונים נמצאים במערכת — מה שחסר הוא מי שיפעל עליהם בזמן.

> **"זאת בעיה שהיא מושלמת לאג'נט" [19:54]** הנימוק לכך שזו בעיה מתאימה לאג'נט, כפי שהוצג: היא חוזרת מספיק פעמים בשבוע כדי להצטבר לסכומים שנתיים משמעותיים; קשה מאוד לתכנת אותה מראש כי כל פעם היא נראית אחרת; היא דורשת reasoning — שקלול ערך ההזמנות, מרחק הרכבים והשפעה על לקוחות; ובסוף היא דורשת ביצוע פעולות ולא רק שליחת אלרט [19:54].

הפתרון שהאג'נט מבצע

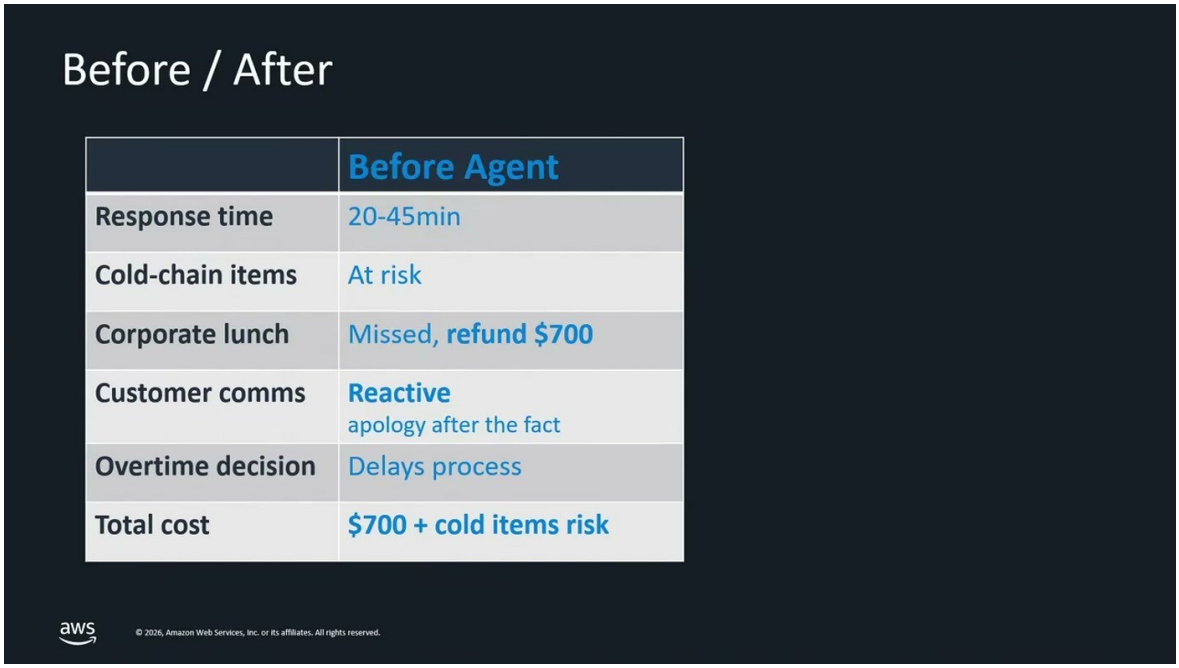
- **נהג A** ממשיך עם המשלוחים הרגילים אחרי שיתוקן התקר — הם אינם דחופים.
- **נהג B,** שלושה קילומטרים משם ועם יכולת קירור, נשלח לקחת את הפריטים המקוררים. הבעיה: הוא סיים משמרת. האג'נט מציע לו בונוס של 45 דולר עבור שעות נוספות, והוא צריך להסכים.
- **נהג C,** רחוק יותר וללא קירור, לוקח את המשלוח הקריטי לישיבת ההנהלה.
- **תקשורת יזומה:** הודעה לנהג A על מי מגיע ומה נלקח, הודעה ללקוח הארגוני שהמשלוח מתוכנן להגיע עם expected delivery מעודכן ל-12:12, והודעה ללקוחות המשלוחים המקוררים. בסוף — לוג של הכול למערכת [21:25].

הפרומפט שהוצג פותח ב-you are a fleet response agent, ומזין לאג'נט את האלרטים, את ההזמנות מהדאטהבייס, את הנהגים הקרובים שהמערכת איתרה אוטומטית, ואת ה-policies של שעות נוספות — ואז מורה לו: להחליט אילו משלוחים להעביר, לאשר overtime אם מוצדק, לבצע שינויי ניתוב, להודיע ללקוחות ולתעד [22:56].

החיבור למערכת

השינוי בארכיטקטורה מינימלי: בין זרם האלרטים לסדרן נכנסת Lambda שמנתבת הודעות לאג'נטים, ואלרטים שהוגדרו כחייבי אדם ממשיכים ישירות לסדרן. "שימו לב שלא הרבה הודעות מגיעות לאג'נטים, אז זה אומר שהעלויות יהיו מאוד מאוד נמוכות, אנחנו נקרא להם אולי מאות פעמים ביום" [22:56].

מתחת למכסה המנוע יש שני אג'נטים: אג'נט אנליזה שמקבל היסטוריה של בעיות דומות, חיבורי MCP למשיכת נתונים מהדאטהבייס ולבקשת ניתוב מחדש מהמערכת, ו-knowledge base עם מדיניות החברה; ואג'נט תקשורת שמתמחה בכתיבת ההודעות לנהגים וללקוחות. השניים מתקשרים דרך Amazon SNS, והכול מנוטר ב-AgentCore Observability [24:32].



עמודת המצב הקיים בהשוואה שהוצגה, לפני שנחשפה עמודת ה-After

לפני האג'נט: זמן תגובה של 20 עד 45 דקות, פריטי קירור בסיכון, ארוחת הצהריים הארגונית מוחמצת עם זיכוי של 700 דולר, תקשורת ריאקטיבית — התנצלות אחרי המעשה — והחלטת שעות נוספות שמעכבת את התהליך. אחרי:

הבעיה נפתרת תוך דקות, הכול מגיע בזמן, התקשורת פרואקטיבית, והעלות היחידה היא תשלום השעות הנוספות [26:04].

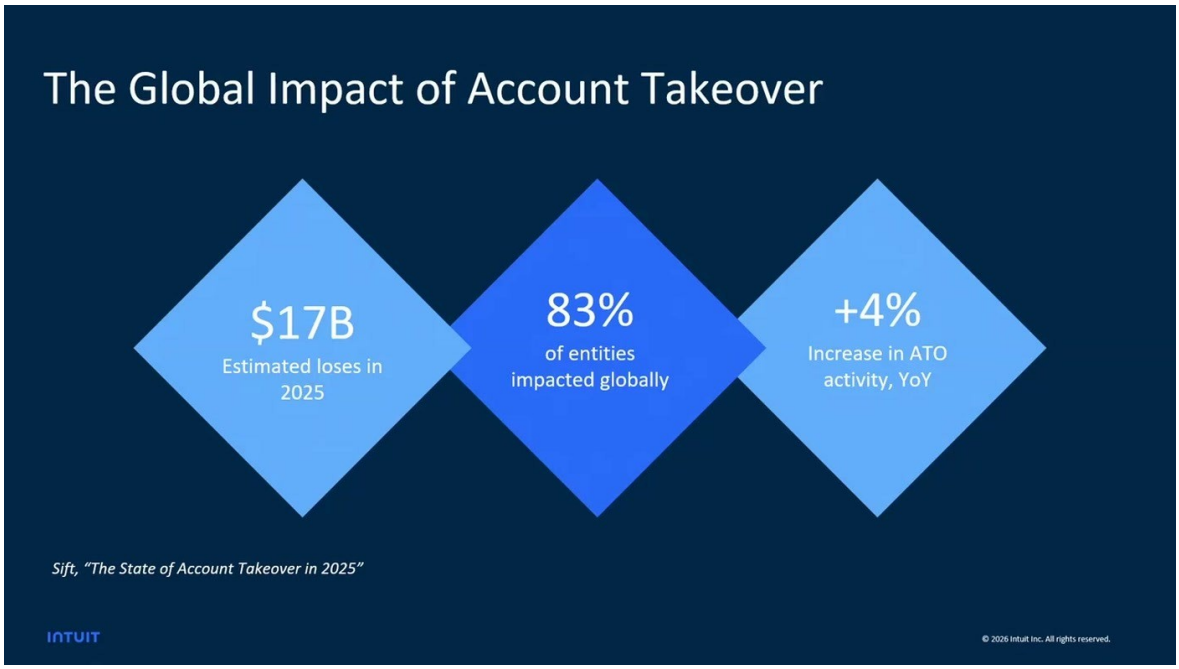
המסקנה שאופר סיכם בה את חלקו: הפייפליין המקורי כבר החזיק את כל המידע, ופשוט היה קשה לטפל בבעיות המורכבות. "היום אני יכול בקלות מאוד להוסיף אייג'נטים, ולפתור את כל הבעיות המורכבות שבעבר היה לי קשה לפתור" [26:04].

7. Intuit: עולם ה-Account Takeover

החלק השלישי הועבר על ידי Senior Staff AI Scientist מ-Intuit, שעוסק בשנים האחרונות במניעת הונאות ובפרט בהשתלטות על חשבונות. הוא פתח בסיפור אמיתי כמסגרת לכל מה שבא אחריו.

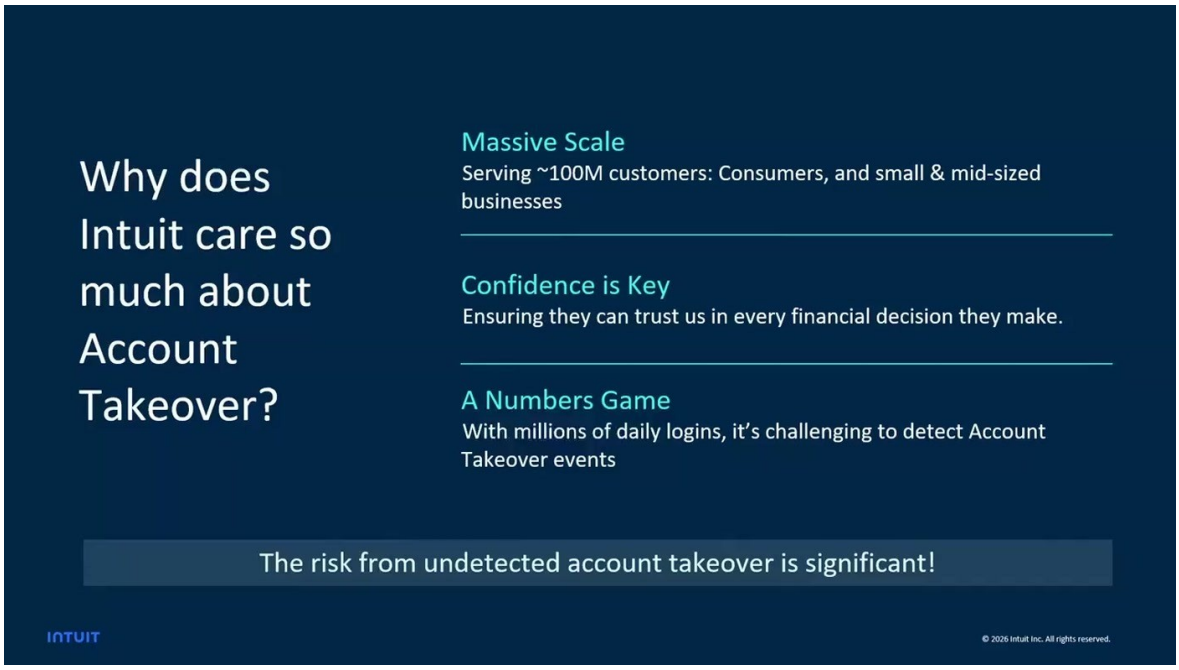
מתקפת SIM Swap מ-2018 כפי שהוצגה: יזם קריפטו, נער בן 15, ושוחד לפקיד

מייקל טרפין, יזם קריפטו מוכר, היה יעד נוח דווקא בגלל הפרסום שלו. נער בן 15 שיחד פקיד בחברת סלולר כדי שינפיק כרטיס SIM חדש עם מספרו של טרפין. מרגע זה כל הודעות השחזור הגיעו לטלפון של התוקף, הוא איפס את הסיסמאות ומשך 24 מיליון דולר מארנק הקריפטו [27:38]. "הסיפור של SIM swap או לשחד וכו', זה לא סיפור חדש" [27:38].



שלושת המספרים שעל הסלייד, מתוך דוח Sift על מצב ה-Account Takeover ב-2025

על פי הסלייד: 17 מיליארד דולר הפסדים מוערכים ב-2025, 83% מהישויות נפגעות ברחבי העולם, ועלייה של 4% בפעילות ATO משנה לשנה. הנפגעים הם גם אנשים פרטיים וגם עסקים — ולדבריו, לא מעט עסקים פושטים רגל בגלל זה [29:11]. הסיבה לגידול פשוטה: הטכנולוגיה מתקדמת בשני הצדדים, "רק שלהם אין שום גבולות" [29:11].



שלוש הסיבות שהוצגו לרגישות המיוחדת של החברה לנושא

Intuit מחזיקה פורטפוליו פינטק — TurboTax לדוחות מס, Credit Karma לניהול פיננסי אישי, QuickBooks לעסקים קטנים ובינוניים ו-Mailchimp — ומשרתת כ-100 מיליון לקוחות [30:44]. הסלייד מסכם: סקייל עצום, אמון כנכס מרכזי, ומשחק מספרים — מתוך מיליוני התחברויות יומיות, קשה למצוא את האחוזים הקטנים של ההונאה, אבל הם משמעותיים כי הונאה אחת יכולה לשאת עשרות מיליוני דולרים [30:44].

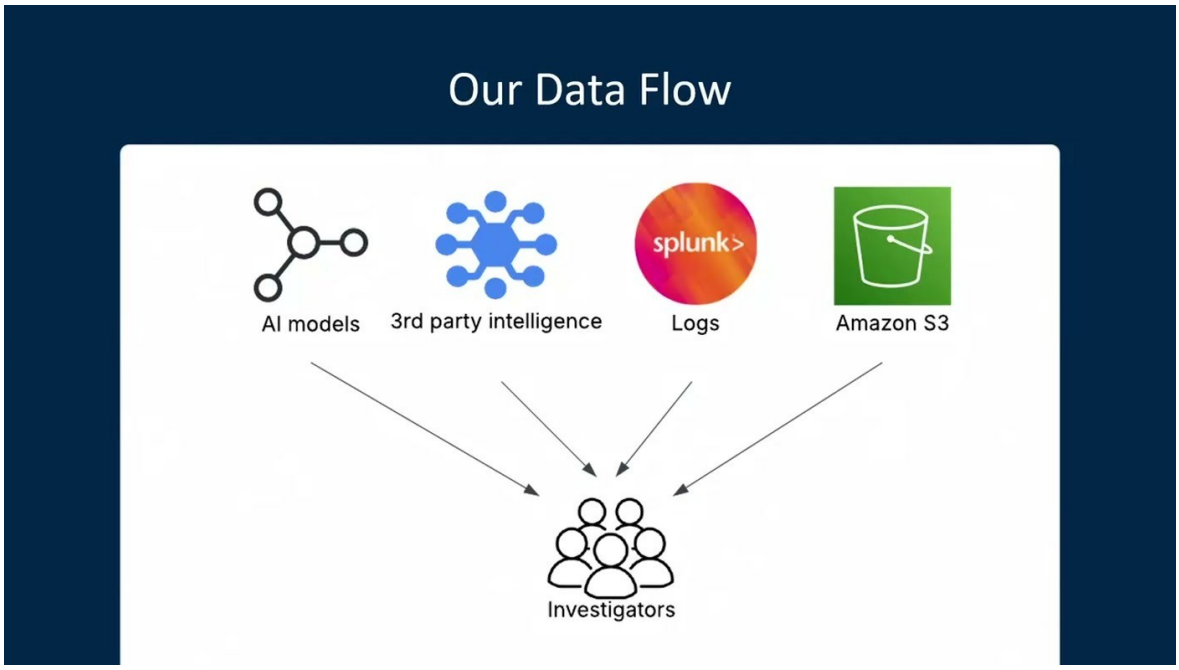
מה עושים החוקרים היום, ולמה זה לא מספיק

זיהוי השתלטות על חשבון נשען על צירוף סיגנלים — מאות ואלפים — כדי לאמוד את ההסתברות שמדובר במשתמש האמיתי. במקרה של טרפין, כתובת ה-IP הייתה בניו יורק בעוד הוא מתגורר בלאס וגאס [30:44]. המערכות בנויות שכבות, ובקצה יושבים חוקרים אנושיים. טרפין עצמו הבין מהר שמשהו קרה, אך עד שהגיע לנציג אנושי וטופל — בין שעה לשעתיים — הכסף כבר נעלם [32:17].

- **סקייל:** "אי אפשר להגדיל את כמות החוקרים כשיש מתקפה או ככל שהעסק שלך גדל" [32:17] — גיוס וצמצום של כוח אדם אנושי אינם ניתנים למתיחה אינסופית.
- **זמן:** שעה עד שעתיים הן נצח בתרחיש שבו הכסף יוצא תוך דקות.
- **עקביות:** בין חוקרים שונים, ואפילו אצל אותו חוקר ביום ראשון מול יום שני, או בתחילת הקריירה מול המשכה [32:17].
- **עומס אנושי:** "זאת באמת עבודה מאוד מאוד קשה ומלחיצה" בסקייל הזה [32:17].

8. הארכיטקטורה בפועל

הנקודה שהודגשה: מערכת ה-ML הקיימת ממשיכה לעבוד כרגיל, והאג'נט מתחבר אליה. הזרימה הקיימת [33:48]: אירועי לוגין — כל אחד עם כתובת IP, זמן ומזהה — זורמים ל-Kafka מנוהל על AWS; Flink מעבד אותם בזמן אמת ומייצר פיצ'רים, למשל סטטיסטיקות על כתובות IP; הפיצ'רים נשמרים ב-DynamoDB לגישה מהירה. כשמשתמש מתחבר, המערכת שולפת את המאפיינים מ-DynamoDB, קוראת למודל ב-Amazon SageMaker לקבלת הסתברות לפריצה, וה-risk service מחליט: לאשר, לחסום, או להעביר לבדיקה ידנית [35:20].



ארבעת מקורות המידע שמזינים את ה-AI Assistant, ובקצה — החוקרים

ה-AI Assistant יושב במרכז ומוזן מארבעה מקורות: מודלי AI וציוני model scores שכבר חושבו קודם, מודיעין מספקי צד שלישי — שיודעים לומר שכתובת IP מזוהה עם פעילות הונאתית או שכתובת מייל שייכת למשתמש לגיטימי — לוגים של המערכות, וכמויות דאטה גדולות ב-Amazon S3 [33:48]. הדובר הבהיר שזו הצגה מפושטת של מערכת מורכבת בהרבה [35:20].

לאסיסטנט יש שני מסלולי הפעלה. הראשון, החקירה הידנית: החוקר שואל אותו בשפה חופשית, כמו שיחה עם ChatGPT. השני, אוטומציות — לדברים שכבר יודעים לזהות בוודאות גבוהה, ואז אין צורך בחקירה ידנית ואפשר לפעול ב-near real time. "במקום שזה ייקח שעה שעתיים זה היה לוקח עשר דקות או חמש דקות", ובאוטומציה אפילו "כמה עשרות שניות" [35:20].

דוגמה לפיצ'רים

הדוגמה שניתנה: ספירת מספר ההתחברויות לכל כתובת IP — מדד זול וקל ב-Flink. אבל כדי להבחין באמת בתוקף, רוצים גם לדעת כמה משתמשים ייחודיים ניגשו מאותה כתובת: הרבה distinct users מאותו IP מצביע על מישהו שסורק חשבונות. ספירה ייחודית מדויקת יקרה, והפתרון שהוצע לשיעורי בית הוא HyperLogLog, שהופך את המימוש לפשוט וזול לניהול [36:52]. הפיצ'רים זורמים ל-DynamoDB, וגם ה-AI agent יכול למשוך משם מידע.

שלושת האתגרים שנמנו, וההחלטה הטכנית שנבחרה לכל אחד

- **Freshness —** במערכות כבדות יש ETL-ים שלוקח להם שעות עד יום. "נתאר סיטואציה שמייקל טרפין מתקשר, ואומרים לו, אנחנו לא רואים בכלל בטבלאות את המידע שלך, תחזור עוד שעתיים — עוד שעתיים כבר מאוחר מדי" [38:26]. הפתרון: דאטה מוזרם ב-Kafka ו-Flink.
- **Latency —** ה-LLM איטי יחסית לתהליכים מבוססי קוד, ולכן מוסיפים שכבת caching ואופטימיזציות נוספות [38:26].
- **Efficiency —** "אם יש לכם משהו שאתם מזהים שזאת משימה שחוזרת על עצמה, ולמעשה על ה-AI agent אנחנו סתם מבזבזים עליו טוקנים — במקרה הזה אנחנו נמיר את זה לקוד דטרמיניסטי", וניתן לאג'נט לקרוא לחתיכה הזאת במכה אחת [38:26].

9. התוצר והתוצאות

המוצר המוגמר, כפי שתואר: החוקר שואל האם המשתמש הזה עבר ATO, והאסיסטנט מחזיר את כל המידע הרלוונטי — כתובת IP חדשה עבור הלקוח, ISP חדש, ציון גבוה יחסית ממודל ה-ATO, ופעילות חשודה כמו שינוי שם המשתמש כדי להשתלט על החשבון, בדיוק כפי שנעשה במקרה שפתח את ההרצאה. בסוף ניתן conclusion: מה קרה ומה כדאי לעשות [40:00].

התוצאות שדווחו: שיפור של פי 64 במהירות הטיפול בחלק מהתהליכים, חוקרים רגועים ובטוחים יותר בהחלטותיהם, ויכולת חסימה בזמן אמת [40:00]. שלוש ההמלצות שסיכם: לצמצם את כמות הקריאות ל-LLM ולכלים, לצמצם את גודל הפלט שלהם, ולהמיר לאוטומציה את מה שחוזר על עצמו [40:00].

> **לתשומת לב** מה שלא נמדד בפומבי: לא הוצג שיעור false positives לפני ואחרי, ולא עלות הריצה. "פי 64" מתייחס לחלק מהתהליכים ולא לכלל החקירות. גם הטענה על הפחתת רעש ב-95% [15:19] נאמרה כתצפית כללית על לקוחות, לא כמדידה במקרה ספציפי.

10. מה לוקחים מכאן

- **המידע כבר אצלכם.** המסר הסופי של הסשן: "תזכרו שהמידע ברוב המערכות שלכם כבר נמצא שם, רק צריך להוסיף אינטליג'נס ולפתור בעיות שהיה לנו קשה לפתור פעם" [41:37].
- **התחילו מאלרט אחד עם ערך עסקי גבוה.** ההמלצה המפורשת: לנתח אילו אלרטים מורכבים נושאים ערך עסקי שמצדיק השקעה, ולבנות סביבם [41:37].
- **האג'נט הוא שכבה, לא החלפה.** בשלושת המקרים — Flink נשאר, Kafka נשאר, מודל ה-ML נשאר. האג'נט נכנס בנקודת ההסלמה בלבד.
- **פצלו לפי אחריות, לא לפי חוכמה.** הנימוקים לפיצול היו סקיילינג, דיבאגינג ו-prompt ממוקד — לא איכות התשובה [10:42].
- **מדדו טוקנים כמו שמודדים compute.** שני דוברים נפרדים הגיעו לאותה מסקנה: מה שחוזר על עצמו צריך לעבור לקוד דטרמיניסטי או ל-cache [38:26], [12:18].
- **התקשורת עוברת מריאקטיבית לפרואקטיבית.** בדוגמת המשלוחים, השינוי המשמעותי ביותר ללקוח לא היה מהירות הפתרון אלא העובדה שהודיעו לו מראש במקום להתנצל בדיעבד [26:04].

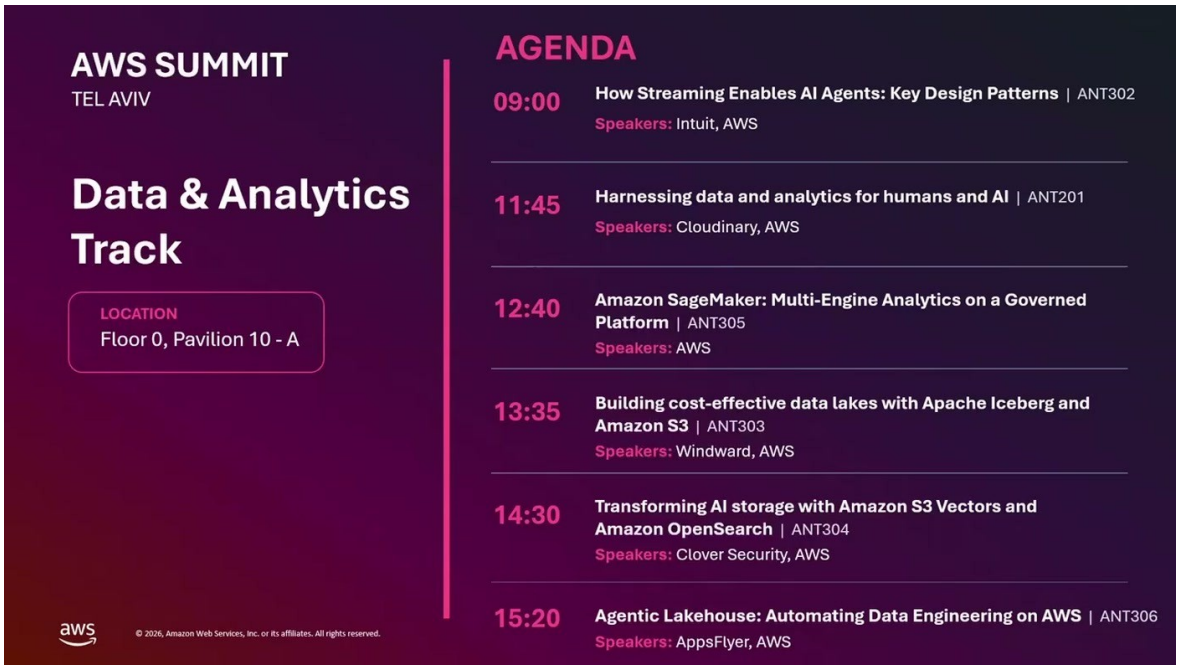
נקודות להמשך

נושא	למה זה חשוב	חותמת זמן
Identity ו-policies ב-AgentCore	הוזכרו כרכיבי הפלטפורמה אך לא הודגמו; בארגון מפוקח הם מכריעי היתכנות	6:05
מי מאשר פעולה אוטונומית	בדוגמת המשלוחים האג'נט מאשר שעות נוספות ומודיע ללקוחות — ללא human in the loop מוצג	21:25
עלות מול ערך	הטיעון כולו נשען על מיעוט קריאות; שווה למדוד את יחס האלרטים שמגיעים לאג'נט	22:56
HyperLogLog לספירות ייחודיות	פתרון זול לפיצ'ר distinct users בזרם	36:52
לולאת המשוב לחוקים	החזרת דפוסים דטרמיניסטיים מהאג'נט אל שכבת החוקים — שם נמצא החיסכון	12:18

11. מונחון

מונח	משמעות
Account Takeover (ATO)	השתלטות על חשבון קיים: תוקף מזדהה בשם בעל החשבון ומבצע בו פעולות
SIM Swap	העברת מספר טלפון לכרטיס SIM של התוקף, כדי לחטוף את מסלול שחזור הסיסמה

מונח	משמעות
Stream Storage	שכבת קליטת האירועים בקצב גבוה עם יכולת replay — Amazon MSK, Kinesis Data Streams
Amazon Managed Service for Apache Flink	עיבוד זרם מנוהל: חלונות זמן, state, פיצ'רים ו-deduplication
Bedrock AgentCore	פלטפורמת AWS להרצת אג'נטים בפרודקשן: runtime, memory, tools, identity, observability
Multi-agent workflow	פיצול אג'נט אחד לכמה אג'נטים בעלי סקופ צר, המחוברים בתורים
Event Driven Architecture	ארכיטקטורה שבה שירותים צורכים אירועים מ-topic או מתור, במקום קריאות ישירות
MCP	פרוטוקול אחיד לחיבור מודל לכלים, למקורות מידע ולשירותים חיצוניים
Geofence alert	התראה על רכב שיצא מהאזור המוגדר לו או שאינו במיקום הצפוי
HyperLogLog	מבנה נתונים הסתברותי לספירת ערכים ייחודיים בזיכרון קטן
Cold chain	משלוחים הדורשים קירור רציף, עם חלון זמן קשיח



הסשן נפתח את מסלול Data & Analytics; שאר ההרצאות במסלול באותו חדר