Cloud Native Threat Intelligence Platform

Mark Blaha

Collin College

Professor Moore

18 November 2025

Table of Contents

Executive Summary

Small businesses are targeted by approximately 70% of threat actors, yet 51% operate without any security measures. Commercial threat intelligence offerings cost $10,000 to $100,000 annually which can exceed small business cybersecurity budgets of $5,000 to $50,000 entirely. This has contributed to consequences such as the complete closure of unprotected small businesses like The Heritage Company and Wood Ranch Medical following ransomware attacks.

To address this market gap, I developed a cloud native threat intelligence platform and designed it specifically for small business use. The platform was built on AWS and it uses serverless technologies like Lambda functions and DynamoDB in conjunction with S3 buckets and a Restful API Gateway to collect raw indicators of compromise (IOCs) from open source threat intelligence feeds, normalizes raw TI data to industry standard STIX 2.1 format, enriches IOCs using the Shodan API, and presents human readable, helpful intelligence to users through a data driven dashboard. The entire platform was designed from the ground up to be operated effectively by a single individual regardless of their cybersecurity expertise.

The platform maintained continuous uptime throughout the entire 1-month testing period and successfully collected and processed over 32,000 IOCs from two live sources, incurring a total of $0.82 in AWS costs. Projected annual costs for this platform are anywhere between $12.00 to $120.00 depending heavily on data volume which is approximately 99% less than most commercial threat intelligence offerings on the market today. The platform exists as a working proof of concept with several technical limitations, including a configuration state drift that necessitated the creation of a hybrid deployment strategy and manual instead of automated data collection. The core premise has been validated, and tangible threat intelligence capabilities can be offered to small businesses at a cost that is appropriate for their needs.

Cloud Native Threat Intelligence Platform

Problem Identification

Although most modern large enterprises have developed remarkably sophisticated cybersecurity measures and IT governance practices, a clear and apparent gap exists concerning small businesses. According to Khalil (2025), “40-72% of SMBs” reported breaches from 2024 to 2025, and during that same time, “ransomware and phishing dominate.” According to IBM (2025), when a successful breach occurs, “the average breach now costs $4.88 million globally and takes 204 days to identify.” While large enterprises may be able to absorb the cost of a breach, the effects on many small businesses can be devastating.

Evidence reveals threat actors are aware of this issue, and according to Bizplanr (2025), “around 70% of cyber attackers intentionally go after small businesses, knowing they often lack the resources for strong cybersecurity.” Despite the strong public knowledge of this problem, “about 33% of small businesses are still relying on outdated cybersecurity tools and systems” (Bizplanr, 2025). Even more alarming, “47% of businesses with under 50 employees don’t budget for cybersecurity, and 51% operate without any IT security measures” at all (Bizplanr, 2025). When you consider that “just 17% of small businesses have cyber insurance,” successful breaches can completely devastate a small business and everyone involved (Rahmonbek, 2025).

Case Study 1 – The Heritage Company

In October 2019, 61-year-old nonprofit telemarketing firm The Heritage Company and its more than 300 employees were hit by a ransomware attack that ultimately led to the firm’s collapse (O’Donnell, 2020). A letter issued by CEO Sandra Franecke reveals numerous critical failures on the part of the organization, including a lack of threat detection systems; no incident response plan in place; failed recovery despite ransom payment; and a complete disruption of

payroll capabilities (O'Donnell, 2020). The financial impact of this breach proved to be insurmountable for the company, and Cluley (2020) astutely observed that "it wasn't ransomware which brought about the apparent demise of The Heritage Company, but instead a lack of secure backups and a resilient disaster recovery plan."

Case Study 2 – Wood Ranch Medical

Similarly, in August 2019, a ransomware attack on Wood Ranch Medical, a healthcare provider in California, caused extensive damage to the company's systems and ultimately led to its complete closure in December 2019 (Spambrella, 2020). Wood Ranch made the legitimate choice not to pay the demanded ransom, and as a result "there was extensive encryption of records, which prevented access to healthcare data" (Garcia, 2024). Although Wood Ranch Medical "had created backups of patient records … those backups were also encrypted and could not be used to restore patient data" (Spambrella, 2020). The result of this incident was damage "[so] extensive that it completely ruined the computer systems" containing "5,835" patients' electronic medical records.

Root Cause Analysis: 5 Whys method

Problem Statement: Around 70% of threat actors intentionally target small businesses despite their smaller attack surfaces compared to large enterprises.

Why #1: Why do threat actors target small businesses? Small businesses are perceived as easier targets that have a weaker security posture and lower likelihood of adequate incident response.

Why #2: Why do small businesses have weaker security postures compared to large enterprises? Smaller budgets force prioritization of business needs rather than security investments that may not deliver an immediate return to stakeholders.

Why #3: Why can't small businesses afford sufficient security tools? Most commercial cybersecurity platforms are priced to their audience, and commercial threat intelligence providers cater their services to organizations with substantial cybersecurity budgets.

Why #4: Why haven't affordable threat intelligence solutions emerged on the market? Market forces are more favorable to enterprise sales and the cost to maintain a threat intelligence platform will be constant regardless of the size of its customers.

Why #5 - Root Cause: The cybersecurity industry targets their services to large enterprises with substantial regulatory GRC mandates and substantial cybersecurity budgets, leaving many small businesses unmotivated to invest in cybersecurity until a breach occurs.

Case Study Analysis

After analysis, these two case studies reveal many similarities. Although the point of initial access is not clear, the impact to both businesses following the attack was the same, and using the MITRE ATT&CK framework as a reference both cases resulted in T1486: Data Encrypted for Impact which involves attackers encrypting "data on target systems … in a network to interrupt availability to system and network resources" (MITRE, 2025). This result strongly suggests that IOC monitoring if utilized in combination with other security measures could have potentially given both affected businesses an early warning before their systems were compromised.

Application of Cybersecurity Knowledge

Domain Identification and Application

The platform's design encompasses five cybersecurity domains. First, threat intelligence / information sharing forms the core functionality of the platform, and I implemented Structured Threat Information Expression (STIX) 2.1 to ensure standardized data formatting and adhere to industry best practices. According to OASIS (2021), STIX 2.1 "enables organizations to share CTI with one another in a consistent and machine-readable manner." The implementation of STIX 2.1 enables the platform to ingest and process data not only from Open Threat Exchange, but any open source threat intelligence which is vital for the modular intent of the platform.

Second, cloud security principles and best practices guided the serverless design of the project. This strategy takes advantage of the shared responsibility model, where AWS manages infrastructure security or security of the cloud while the platform itself is responsible for maintaining security of the data in the cloud. The serverless approach eliminates any vulnerabilities associated with the provisioning and management of traditional servers while at the same time handling a variable level of data volume. As a result of the serverless approach, neither patching nor server maintenance is ever required and platform costs scale with data usage only which creates significant value for small business use.

Third, data protection ensures confidentiality and threat intelligence data integrity. DynamoDB provides encryption by default, and AWS API gateway provides encryption via TLS for all data in transit. The platform implements 90-day data archiving using an S3 bucket giving potential users lower expenses over a long period of time. I configured IAM roles with least privilege access to ensure that each individual component can only access cloud resources required for its operation, leading to increased security.

Fourth, the incident response process specifically for small business use was carefully considered when creating the data processing pipeline. The platform's three lambda functions, which handle data collection, enrichment, and processing, operate in sequence to transform raw threat intelligence data pulled from the source APIs into helpful intelligence. Real time enrichment courtesy of Shodan adds significant value on top of the raw threat data that is specifically intended to be human readable, for example real time geolocation mapping and organization detection. Considering the small business intended audience of the project heavily influenced the creation of the minimalist, data driven and information rich dashboard of the final deliverable.

Fifth, risk management and general GRC principles heavily influenced the project's development, and a cost benefit analysis revealed that preventing just a single breach and alleviating the risk associated with estimated costs of $4.88 million (IBM, 2025) would justify years of platform operation at its current upper bound cost estimate of $100 per year.

Literature and Documentation Review

The platform's design adheres to industry standard frameworks to give comprehensive coverage. First, the NIST (2024) CSF provides extensive guidance to the foundational structure of the project through its core functions. The platform continuously identifies threats by automatically collecting data from multiple open source TI feeds. The platform is currently storing over 32000 IOCS and the STIX 2.1 implementation used by the project makes threat identification consistent over multiple sources. STIX 2.1 implementation additionally enables future modular upgrades to the existing project.

Protective controls are implemented in a proof-of-concept capacity. API gateway limiting prevents abuse and controls costs, while IAM roles have been fully implemented and ensure

role-based AC and least privilege access. IAM roles combined with the project's serverless architecture inherently create a more isolated environment when compared to the legacy server model which would prevent lateral movement within the system since every task is given its own IAM role.

Real time detection is the entire purpose of the project, and by continuously monitoring TI feeds the project can help potential users detect threats before they pose a risk to their organization. Specifically for small business use, even slightly lesser quality open source CTI is far more useful when it comes to detection than simply not using threat intelligence at all. In addition, the frontend was designed to be completely data driven, with basic metrics being front and center so that users can get a high level overview of the data points immediately.

Automated response was carefully developed with small business users in mind. When threats are identified and interacted with by the user, the enrichment lambda adds helpful context without any further human intervention needed such as geolocation data, port information on that host, and organization data if it's available. Thanks to the Shodan API, the enrichment lambda can transform raw TI JSON data points into human readable and helpful intelligence. The project's workflow was tailored specifically for its intended audience and designed to be effectively operated by only one individual.

The serverless design and especially terraform implementation was created to provide users with easy recovery. In theory, the entire project infrastructure can be destroyed and re-provisioned in less than 1 hour which means that potential users will be able to rapidly recover their threat intelligence capabilities following an incident.

Infrastructure development proceeded with security as a top priority. I created the security terraform module first, which created IAM roles that would be used by all the

subsequent infrastructure modules. The principle of least privilege was extensively applied throughout the project using IAM roles. Each lambda function operates with an IAM role that gives it the minimum permissions necessary to accomplish its task. For example, the collection lambda collector.py can write to DynamoDB but is not able to modify enrichment data. Furthermore, the enrichment lambda can read and update IOCs to accomplish its task of appending enrichment data to them but cannot delete them. This approach takes advantage of least privilege access to not only create a more secure system but limit any potential impact following a compromise.

Trade-offs and Cost Considerations

Following the initial project plan creation, I made several modifications to account for reasonable limitations of a student project and cost considerations. First, the originally proposed fully automated data collection pipeline was significantly modified. The original project plan suggested routine data collection at regular intervals, but to limit use of cloud resources the final project collects data on manual user request only. Advanced analytics such as machine learning based data point correlation have not been implemented in the project, and the API gateway was created with rate limiting built in solely for the purpose of limiting cloud resource expenditure.

Even with those hard coded limitations in mind, the project's performance has exceeded expectations. The platform's ability to collect and process over 32,000 data points for less than $1.00 in total infrastructure costs demonstrates the effectiveness of the serverless model. The platform's use of terraform created one of the biggest issues with the project in configuration state drift. During the testing phase, I encountered a serious bug during the Terraform compute module deployment, specifically lambda package deployment timeouts that required a workaround. To address this issue, the compute module was deployed manually using the AWS

CLI. As a result of this, the project developed a persistent configuration state drift meaning that the current state maintained in AWS did not match the expected state found in terraform configuration files. Although in theory, the advantage of using Terraform means that the entire project infrastructure could be taken down and re-deployed in minutes, the configuration state drift issue in practice means that any Terraform deployment commands would effectively break the API gateway and require extensive troubleshooting.

Problem Solving and Solution Development

Solution Alternatives and Market Research

Commercial Threat Intelligence Platforms

The commercial threat intelligence market, while mature, is dominated by enterprise focused vendors who have priced their offerings towards their target market and effectively cut out small businesses. Recorded Future is one of the market leaders in this space, however they do not self-publish pricing for their CTI services. Vendr (2025) reveals that the median Recorded Future customer spends “$70,200” per year on threat intelligence services, with a range between “$22,100” and “$125,249” (Vendr, 2025). CrowdStrike’s Falcon offering is more compelling at first glance from a pricing standpoint, with their Falcon Go Small Business service priced at “$59.99” per endpoint per month with a stipulation that a minimum unspecified number of endpoints is required (TrustRadius, 2025). When you consider that even the smallest small businesses can be assumed to have at least 10 machines at a minimum, this figure will end up north of $6,000 per year. For context, Tsanev (2025) reveals that small businesses “typically spend between $5,000 and $50,000 per year on cybersecurity,” which means that commercial services priced towards the typical large enterprise cybersecurity budget of between $10,000 and $100,000 annually are just not feasible.

Open source alternatives

Open source threat intelligence platforms offer an alternative to stiff commercial pricing structures however they introduce their own concerns. Malware Information Sharing Platform (MISP) provides crowd sourced IOC sharing capabilities, however MISP deployments require substantial infrastructure deployments and hardware requirements such as “16GB memory and 2vcpus [which] are quite common for smaller sharing hubs,” all the way up to “128GB memory

with 32 physical CPU cores on modern Xeon CPUs" for larger deployments (MISP Project, 2025). Additionally, a MariaDB deployment is required and "using SSDs is highly recommended" (MISP Project, 2025). As a result, an MISP deployment may save businesses in terms of vendor costs however infrastructure deployment and all the ongoing maintenance that goes along with that means that this approach requires a considerable manpower investment in infrastructure expertise.

Serverless solution

The serverless approach selected for the project effectively addresses the limitations of both platforms. By utilizing AWS managed web services like lambda functions and DynamoDB, the platform eliminates the infrastructure management burden and achieves costs appropriate for small business use. The serverless approach eliminates server patching and maintenance requirements and the front end was intentionally created to be data driven and readily usable by a single individual regardless of their technical knowledge. The final product does not attempt to replicate the advanced analytics or ML features that commercial offerings like CrowdStrike or Recorded Future are able to offer, its sole focus is on the core value proposition of providing valuable threat intelligence to businesses that otherwise would have none.

Solution Development Process

The development process followed a structured four phase approach with a priority placed on infrastructure development before proceeding to data processing, initial deployment and testing, and the final user interface.

Advanced artificial intelligence tools, specifically Claude Code was used exhaustively throughout the project's development. Claude Code is an agentic terminal based artificial intelligence tool that was used for planning, project design, user interface design, code

generation, troubleshooting, and project testing. By taking full advantage of artificial intelligence tools like Claude Code during the project's development, higher quality code was produced significantly faster while at the same time improving the quality of the final product.

Phase 1: Infrastructure Development

I began development by creating the AWS infrastructure using Terraform. Using terraform configuration files, I created foundational components: the security module that created and established IAM roles, DynamoDB tables created for structured data storage, S3 buckets for raw data storage and frontend hosting, CloudWatch for basic monitoring, and a compute module that created the foundation for the lambda functions.

Initial decisions made during this phase include the selection of DynamoDB over alternatives like RDS due to Dynamo's serverless billing model that is completely in line with the rest of the project's design, built in encryption at rest, and the ability to handle raw STIX 2.1 data points. Additionally, static hosting using an S3 bucket was an ideal choice for the platform primarily due to its extremely low cost making it appropriate for a student project. I made a conscious choice to develop the security module first and implement least privilege access using IAM roles.

Phase 2: Backend Development

The second phase involved the creation of the lambda function-based data processing pipeline. I developed the collection function first, and integrated Open Threat Exchange (OTX) to pull threat intelligence data. I created basic STIX 2.1 logic to normalize all incoming raw data into STIX 2.1 format regardless of the source. The enrichment lambda calls the Shodan API to add human readable, helpful intelligence to the raw threat intelligence data. This implementation transforms raw TI data points into helpful intelligence appropriate for small business use. The

processing lambda handles data processing including de-duplication using basic hash-based comparison as well as DB write operations.

Phase 3: Initial deployment and testing

Following phase two, testing began. This was by far the most challenging aspect of the entire project for numerous reasons. I began the testing phase with initial infrastructure deployment, which led to the discovery of a deployment bug after running the terraform apply command. Following the successful deployment of the security, database, and storage modules using Terraform, lambda package deployment timeouts occurred when deploying the compute and networking module. To get around this, I deployed the lambdas directly to AWS using the AWS CLI which led to a persistent terraform configuration state drift issue. As a result of this state drift, getting the API gateway up and running posed a significant challenge. The API gateway and its three endpoints were initially created using the AWS CLI rather than from the Terraform networking module which led to further terraform state drift. Project testing proceeded and I discovered an eventual work around utilizing a hybrid deployment strategy, where the compute module containing the lambda functions as well as the networking module containing the API gateway were both deployed manually using the AWS CLI. This solution was successful and all three API gateway endpoints were working as expected.

Following successful deployment of the infrastructure and API gateway, I began testing the endpoints. I initiated the first collection using the /collect endpoint and verified that live TI data was being correctly returned by the /search endpoints. I then tested the /enrich endpoint using well known references (1.1.1.1 and tesla.com) and the expected enrichment data was returned by Shodan. Finally, to complete the testing phase I integrated a second TI source (Abuse.ch) into the lambdas and re-deployed them using the AWS CLI.

The backend development phase of the project was extremely challenging for two main reasons. First, due to the novel nature of backend development a fair amount of time was spent learning about new concepts for the first time before putting them into practice. Second, the terraform state drift issue proved to be exceptionally challenging. Getting the API Gateway up and running was a hurdle by itself and then developing the hybrid deployment strategy so that work could continue took substantial thought and trial and error.

Phase 4: Frontend Development

Following the completion of the testing phase, frontend development could proceed. The general frontend plan involved the creation of a data driven, easy to read user experience that would be appropriate for small business use. I created the dashboard using vanilla TypeScript and a modular component-based design, TailwindCSS, Leaflet, and Chart.js. I integrated real time data into the frontend from the very beginning of development, and this approach was advantageous compared to for example using dummy data due to the data first nature of the project. The dashboard displays real time metrics front and center including total IOCs, high risk IOCs, recent threat activity, and active data sources. A basic geolocation map created using Leaflet visualizes the geographic location of top IOCs giving users an at-a-glance understanding of the current state of the threat landscape without requiring specialized cybersecurity knowledge. Finally, I implemented basic data analytics in a proof-of-concept capacity to give users a visual representation of their data. Analytics features implemented at this time include IOC type distribution, Confidence distribution using confidence data provided by TI sources, source effectiveness, and basic geographic risk analysis using Shodan enrichment data.

I designed the entire frontend around the small business intent of the project, specifically for those outlier small businesses discussed before that have no threat intelligence capabilities

whatsoever. The entire frontend was intended to be used by a single individual with very minimal cybersecurity technical knowledge so that they can get valuable intelligence and hopefully augment their existing security strategy with that intelligence.

Project Architecture

Component Details

API Gateway

AWS API Gateway provides three primary endpoints:

/collect: Triggers initial data collection, pulls fresh TI from connected sources

/enrich: Initiates enrichment for stored data points using Shodan API

/search: Provides search query functionality to frontend dashboard or manual testing

All AWS API Gateway endpoints implement rate limiting to prevent abuse, and endpoints are encrypted using HTTPS.

Lambda Functions

I created three python-based lambdas:

collector.py: Connects to external TI API keys, currently Open Threat Exchange (OTX) and Abuse.ch, Implements basic data normalization to STIX 2.1 to standardize data points coming from different sources. Handles API authentication using AWS Secrets Manager

enrichment.py: Calls the Shodan API to add context to raw TI data points, adds IP based geolocation incl. country, city, coordinates to appropriate data points, detects open ports and services, performs organization identification and ISP identification.

processor.py: Performs basic hash based de-duplication to prevent storage of identical data points, writes processed data points to DynamoDB, handles all search query logic.

Storage

DynamoDB: Primary structured data storage, three tables created which are the primary TI table, de-duplication table for hash value storage, and OSINT enrichment table. Global secondary index (GSIs) was used to enable various search query features by indicator type, source,

timestamp. DynamoDB is encrypted at rest by default as well as serverless, scaling automatically with data volume rather than requiring instance provisioning which saves significantly on costs. **S3:** Raw data archive that preserves API responses. Lifecycle policies were implemented to transition data to lower cost tiers after 90 days to significantly reduce storage costs.

Frontend

The project's frontend provides users with a data driven interface designed specifically for small business users. Created using vanilla TS and a component-based design, the dashboard provides users with clarity into their data and easily usable components rather than artificial complexity. The main dashboard view displays four key metrics prominently: total IOC count, high risk IOC count, recent threat count, and data source count. This summary enables individuals regardless of their cybersecurity acumen the ability to get a quick glance into the threat landscape based on recently pulled data points. Basic search functionality with partial matching is built right into the dashboard, and users can perform a query into a specific data point to ascertain more information about it. Enrichment is performed manually upon user request, and the Shodan API is used to enrich all searched indicators after they are interacted with by the user. A basic heatmap built using Leaflet provides visual representations of where the top IOCs are in the world which may help users identify geographic patterns in the threat data without requiring low level knowledge of this area. Finally, basic analytics created using Chart.js provide a visual view of TI data that is stored by the platform. Users can quickly see IOC type distribution, confidence distribution, source effectiveness, and geographic risk as well as collection history all in one place.

Infrastructure

Terraform was used for the project's infrastructure design, and discovered bugs notwithstanding this design choice would give users the ability to completely reproduce the project infrastructure which is crucial for small business use. In addition, infrastructure changes are tracked by terraform and theoretically, the use of terraform would give users the ability to rapidly recover from any potential incidents and rapidly re-provision all resources in minutes. Additionally, Terraform is highly cost transparent since every time you run a terraform plan you get an explicit list of all resources that are about to be provisioned.

Risk Assessment and Mitigation Strategies

Risk Assessment Methodology

I developed the project's risk assessment strategy using guidance obtained from the NIST Risk Management Framework (RMF) and adapted it for project context. I identified and analyzed risks using a qualitative approach and displayed them in a Risk Assessment Matrix, found below. Risk likelihood was rated as Low, Medium, or High based on probability, while impact was rated similarly based on consequences posed to the platform should that risk be realized. I determined that risks with Medium or lower residual value can be reasonably accepted given the project's context and status as a pre-alpha testing proof of concept system.

Risk Assessment Matrix

RISK	LIKELIHOOD	IMPACT	MITIGATION	RESIDUAL
Data Quality Variance – Open source feeds may not have the highest quality data points or may contain outdated IOCs	High	Medium	Preserve confidence rating from sources; Clearly display confidence levels; Enrichment adds valuable context	Medium - Accepted
External API limitations – External sources may limit requests that could interrupt data collection	High	Medium	Manual trigger implemented to prevent excessive API calls; second source implemented for redundancy	Medium - Accepted
Cost overruns – Unexpected cloud resource usage could exceed projections	Medium	High	DynamoDB selected due to its serverless nature; S3 lifecycle policy implemented; Manual data collection trigger	Low

			implemented over automation	
Terraform State Drift – Configuration inconsistencies caused by hybrid terraform/AWS CLI infrastructure deployment	High	Medium	Documented hybrid deployment strategy in the project repo; deployed compute and networking modules manually	Medium – Accepted
Exposed API keys – exposed keys could enable abuse of external APIs and incur costs	Low	Medium	API keys migrated to AWS Secrets Manager, hardcoded keys from early development should be refreshed	Low
AWS Outage – Any AWS outages could interrupt the platform's availability	Low	High	External factor, out of project's control	Medium

Risk acceptance rationale

Several risks retain medium residuals; all of these were accepted primarily based on the proof-of-concept state of the project. Data quality variance from lower quality open source feeds is necessary to enable the lower cost model and achieve the main threat intelligence accessibility goals of the platform. The terraform state drift issue represents a major issue however one that does not necessarily harm the platform's functionality in its current state. Exposed API keys, specifically in the form of hardcoded keys found in the repo during early development were accepted as a risk, API keys were migrated to AWS Secrets Manager and API Gateway rate limiting serves as a control against excessive API calls. AWS service outages represent a high

impact, low likelihood event that are completely beyond the project's control. In the early stage of project development, an AWS outage briefly interrupted work on the project infrastructure but it was resolved within a short time.

Solution Implementation

Development Environment

Infrastructure:

TECHNOLOGY	PURPOSE
TERRAFORM	IAC for cloud resource creation
AWS CLI	AWS interaction, hybrid deployment, testing
GIT	Version Control

Backend:

TECHNOLOGY	PURPOSE
PYTHON	Lambda function runtime
BOTO3	AWS library for python
REQUESTS	HTTP library
PYTHON-WHOIS	Domain registration lookup

Frontend:

TECHNOLOGY	PURPOSE
VITE	Build and development server
TYPESCRIPT	Frontend development
TAILWINDCSS	CSS framework (styling)
CHART.JS	Basic analytics and visualization
LEAFLET	Geolocation heatmap

Terraform Module Structure

```
infrastructure/terraform/
├── main.tf              # Root module, provider configuration
├── variables.tf         # Input variable definitions
├── outputs.tf           # Output value definitions
├── modules/
│   ├── compute/         # Lambda function resources
│   ├── database/        # DB tables and GSI definitions
│   ├── monitoring/      # CloudWatch
│   ├── networking/      # API gateway and endpoints
│   ├── storage/         # S3 buckets
│   └── security/        # IAM roles, Secrets Manager
```

IAM Role Configuration

A single IAM execution role serves all three lambdas. Trust policy only permits the AWS Lambda service to assume this role

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "sts:AssumeRole",
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      }
    }
  ]
}
```

Attached Policies

Four policies are attached to this execution role, granting minimal permissions for specific services:

AWS managed policy 1: AWSLambdaBasicExecutionRole

Provides basic Lambda permissions including CloudWatch write access

Custom policy 1: DynamoDB access

Grants specific DB permissions on project specific tables only

Permitted actions: PutItem, GetItem, Query, Scan, UpdateItem, DeleteItem, BatchGetItem, BatchWriteItem

Permitted resources: Tables matching “threat-intel-platform-*” naming convention only

Custom policy 2: S3 bucket access

Grants S3 permissions on specific buckets only

Permitted actions: GetObject, PutObject, DeleteObject, ListBucket

Permitted resources: Buckets matching “threat-intel-platform-*” naming convention only

Custom policy 3: Secrets Manager access

Permitted actions: GetSecretValue, DescribeSecret

Permitted resources: “threat-intel-platform/api-keys/dev”

Secrets Manager Implementation

AWS Secrets Manager stores API keys in JSON:

```
{
  "OTX_API_KEY": "[REDACTED]",
  "SHODAN_API_KEY": "[REDACTED]",
  "ABUSE_CH_API_KEY": "[REDACTED]"
}
```

DynamoDB Table Design

Main TI Table

ATTRIBUTE	TYPE	PURPOSE
OBJECT_ID (PK)	String	STIX Object ID
OBJECT_TYPE (SK)	String	STIX object type
CREATED_DATE	String	Timestamp
SOURCE_NAME	String	Source identifier
CONFIDENCE	Number	Confidence score 0-100
PATTERN_HASH	String	Sha-256 hash for de-dup
THREAT_TYPE	String	Threat classification
RISK_SCORE	Number	Risk score 0-100
GEOGRAPHIC_REGION	String	Geographic region
IOC_TYPE	String	IOC type
LAST_MODIFIED	String	timestamp

De-duplication Table

ATTRIBUTE	TYPE	PURPOSE
CONTENT_HASH (PK)	String	Sha-256 hash for de-dup
EXPIRES_AT	Number	TTL - timestamp

OSINT enrichment Table

ATTRIBUTE	TYPE	PURPOSE
OBSERVABLE_VALUE (PK)	String	Ip, domain, url, hash
ENRICHMENT_TYPE (SK)	String	Enrichment source
EXPIRES_AT	Number	TTL - timestamp

GSIs

GSI NAME	PK	SK	PURPOSE
TIME-INDEX	OBJECT_TYPE	CREATED_DATE	Recent indicators by type
SOURCE-INDEX	SOURCE_NAME	CONFIDENCE	High confidence IOCs
PATTERN-HASH-INDEX	PATTERN_HASH	N/A	De-dup check
RISK-ANALYTICS-INDEX	THREAT_TYPE	RISK_SCORE	High risk threats by category
GEOGRAPHIC-INDEX	GEOGRAPHIC_REGION	CONFIDENCE	Geographic threat identification
IOC-PATTERN-INDEX	IOC_TYPE	PATTERN_HASH	Search by ioc type

S3 Bucket Configuration

Raw data bucket: Archives original API responses from TI sources, lifecycle policies transition bucket objects to S3 standard-IA (infrequent access) after 90 days and support user configurable retention periods to minimize storage costs

Processed data bucket: Stores STIX 2.1 formatted data for long term retention

Frontend bucket: Hosts the static web dashboard, public read access configured for S3 static website hosting

Lambda function implementation

Three python lambdas collect and process threat intelligence data:

Collection (collector.py)

CONFIGURATION	VALUE
MEMORY	256MB
TIMEOUT	5 minutes
RUNTIME	Python 3.11

What it does:

1. Retrieve API keys from Secrets Manager

2. Connect to Open Threat Exchange (OTX) and Abuse.ch to pull TI feeds
3. Parse responses, normalize using STIX 2.1
4. Archive raw responses to S3 raw bucket
5. Write processed indicators to DynamoDB TI table
6. Perform basic hash based de-dup using de-dup table

Enrichment (enrichment.py)

CONFIGURATION	VALUE
MEMORY	1024MB
TIMEOUT	5 minutes
RUNTIME	Python 3.11

What it does:

1. Receives input data points that need enrichment
2. Queries Shodan API
3. Performs Whois lookup
4. Store enrichment results in DB
5. Returns enriched data to endpoint

Processing (processor.py)

CONFIGURATION	VALUE
MEMORY	512MB
TIMEOUT	5 minutes
RUNTIME	Python 3.11

What it does:

1. Handles search queries from frontend / manual API calls
2. Performs DB queries using GSIs based on search term
3. Return JSON to endpoint

CloudWatch

CloudWatch logs created with 7-day retention policies have been attached to each lambda. All Logs are encrypted using AWS KMS (See Fig. 5)

Documentation and Reporting

GitHub Repository Structure

The project's source code is public and can be located at https://github.com/realmbs/senior-project. A standard README.md file provides a high level overview of the project complete with detailed repository structure and basic instructions. The repository structure separates infrastructure and backend components (located in infrastructure/terraform/), and frontend components (located in frontend/). Git was used for version control throughout the entire project's development.

Infrastructure Documentation

Terraform configuration files extensively document all cloud resources. Each terraform module includes semantic comments explaining resource purpose. Each module features a main.tf configuration file that configures and provisions cloud resources, a variables.tf file that documents variables, and an outputs.tf that defines export values for use between modules.

Testing and Validation

Endpoint testing results

/collect endpoint: Successfully retrieved and processed raw threat intelligence data from OTX and Abuse.ch APIs (See Fig. 5)

/enrich endpoint: Validated using known references (1.1.1.1 and tesla.com) – Shodan API correctly returned expected data (See Fig. 1)

/search endpoint: Confirmed search queries return correct results (See Fig. 2)

DB testing

See Fig. 3 – AWS DynamoDB ItemCount, TableSizeBytes, TableStatus

32,000+ IOCs successfully collected and stored in DynamoDB, STIX 2.1 format compliance verified using manual inspection (See Fig. 4)

Monitoring (CloudWatch)

See Fig. 5 - CloudWatch logs capture all Lambda details w/ 7-day retention

Pricing Validation

See Fig. 6 - Platform brought online and kept online continuously for one month, total cost as billed by AWS $0.82

Technical Challenges and What Worked Well

The most substantive technical challenges encountered during the project related to initial Terraform module deployment. As discussed previously, lambda package deployment timeout issues encountered during the terraform apply process necessitated the creation of a hybrid Terraform/AWS CLI deployment strategy. This approach, although successful, led to significant Terraform configuration state drift that would have to be addressed before production use of the platform. AWS API Gateway configuration was the most time intensive component of the project due to a complete lack of prior backend development experience. Troubleshooting the API Gateway deployment was the most challenging aspect of the entire project.

Several aspects of the project came together exceptionally well. The security first development strategy incorporating least privilege principles prevented permission issues while at the same time giving the platform increased security. DynamoDB's serverless billing approach perfectly aligned with the project's overall intent and goals and contributed substantially to the lower than expected 1-month bill. The use of Claude Code significantly accelerated and improved the entire development process, not only improving code quality but reducing development time substantially compared to manual work alone. Claude Code was additionally

used to create semantic commit messages throughout project development and generally get the most out of the Git version control system.

Finally, the final one-month cost of $0.82 was substantially more affordable than even the lower end estimates taken at the beginning of the project. The general idea behind the entire project is that some threat intelligence is far more beneficial to small businesses who previously had zero access than no intelligence at all, and with that 1-month billing result the project's feasibility in a business setting is strong.

Future Improvements

Although the project has successfully achieved a proof-of-concept working state, there is obviously room for significant improvement. The highest priority being the Terraform configuration state drift issue that would require most likely a complete rebuild of the API Gateway, once this issue is resolved the project would be ready for an alpha testing phase. Automated testing of the lambdas using pytest would be a valuable addition to the project, as would CI/CD integration for the frontend. Finally, advanced analytics and threat correlation using a reference like the MITRE ATTACK framework would be incredibly valuable for an alpha test of the platform.

Communication and Presentation

Presentation Strategy

I structured the presentation around a live demo rather than planning a long winded presentation. The project's value can be best communicated by demonstrating it in operation rather than speaking on technical details. The live demonstration will consist of navigating to the live website; explaining the data forward nature of the dashboard; showing which sources are currently connected; demonstrating the search functionality to display an IOC; testing the /enrich endpoint using two well known sources (1.1.1.1 and tesla.com); showing the visualization heatmap; and showing the analytics features.

Supporting materials include a slide deck that effectively communicates the problem, the developed solution, a detailed cost breakdown comparing the platform to current commercial offerings, the status of the project including project metrics, and an overview of the project's technology stack.

Innovation and Critical Thinking

The platform was designed from the ground up for small business use, and the platform can be effectively operated by a single individual and give them valuable information about the threat landscape without requiring cybersecurity expertise. Small businesses who previously had no threat intelligence capabilities should be able to use this platform to meaningfully improve their security posture. This approach represents a significant departure from the large enterprise that operates a 24/7 SOC and employs a team of professionals to staff it to obtain the best level of protection.

The project's core premise is that threat intelligence should be accessible to all organizations, not merely those with impressive budgets that can afford the best commercial threat intelligence solutions. The current threat intelligence market is creating a two-tiered system where large enterprises are able to purchase increasingly advanced and comprehensive threat visibility, while the small businesses that make up the majority of organizations in the United States are left in many cases to operate completely blind. This platform effectively demonstrates that these artificial barriers to data access can be overcome by taking advantage of cloud computing and open source threat intelligence feeds.

Overall Impact and Effectiveness

Final Project Metrics

METRIC	RESULT
TOTAL IOCS COLLECTED	32,000+
LIVE DATA SOURCES	2 (OTX, ABUSE.CH)
TOTAL AWS SPEND (30 DAYS)	$0.82
PLATFORM UPTIME	30+ DAYS CONTINUOUS
PROJECTED ANNUAL COST	$12 - $120

The project's primary goal was to create a functional and usable threat intelligence platform for small business use and use a serverless cloud native design to achieve significant cost savings over commercial TI offerings. The result exceeded expectations, and a proof-of-concept deployment resulted in a functional threat intelligence platform that can collect and store over 32,000 IOCs from multiple sources for a 1-month cost of $0.82.

Additionally, the Shodan API was successfully integrated as well as visualization features used to transform raw STIX 2.1 threat intelligence data into helpful and human readable intelligence. The user interface was designed to effectively be operated by one individual regardless of their cybersecurity expertise and give potential users the capability to harness threat intelligence data in their day-to-day operations.

The project's cloud native design means that potential expansion beyond a basic proof of concept into an alpha testing phase would be feasible with moderate effort. Additional feeds could be integrated and advanced automation could be added to the project to automatically schedule TI data collections on a regular basis.

The Terraform configuration state drift issue is currently the biggest issue that needs to be addressed before the project could transition into a future state. Although the current hybrid Terraform/AWS CLI deployment strategy is functional, it is far from the optimal approach and this deployment inconsistency introduces a major risk to potential users.

References

Bizplanr. (2025). *Top cybersecurity statistics for businesses in 2025*. https://bizplanr.ai/blog/cyber-security-statistics

Blaha, M. (2025). *Senior Project* [GitHub repository]. https://github.com/realmbs/senior-project

Blaha, M. (2025). *Threat Intelligence v1.0.* [Project website] http://threat-intel-platform-frontend-dev-53cc9e74.s3-website-us-east-1.amazonaws.com

Bolen, S. (2024, November 26). *The future of cyber threat intelligence (CTI) for SMBs: Success stories and trends to watch in 2025*. Medium. https://medium.com/@scottbolen/the-future-of-cyber-threat-intelligence-cti-for-smbs-success-stories-and-trends-to-watch-in-2025-e09377f00de8

Cluley, G. (2020, January 6). *Company held hostage by ransomware shuts down, tells 300 employees to find new jobs*. Graham Cluley. https://grahamcluley.com/ransomware-shuts-company/

Garcia, C. (2024, January 25). *Wood Ranch Medical totally shut down operations due to ransomware attack*. CalHIPAA. https://www.calhipaa.com/wood-ranch-medical-totally-shut-down-operations-due-to-ransomware-attack/

IBM. (2025). *Cost of a data breach 2025 report*. IBM Security. https://www.ibm.com/security/data-breach

Khalil, M. (2025). *Cyber Attacks on small businesses 2025*. DeepStrike. https://deepstrike.io/blog/cyber-attacks-on-small-businesses

Market.us. (2025, June). *Cyber threat intelligence market size, share analysis report*. https://market.us/report/cyber-threat-intelligence-market/

MITRE. (2025). *Data Encrypted for Impact | The Mitre ATT&CK Framework v18.* MITRE Corporation. https://attack.mitre.org/techniques/T1486/

MISP Project. (2025). *Sizing your MISP Instance*. https://www.misp-project.org/sizing-your-misp-instance/

NIST. (2024). *The NIST Cybersecurity Framework (CSF)2.0.* https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.29.pdf

OASIS. (2021). *STIX™ version 2.1 OASIS standard*. Organization for the Advancement of Structured Information Standards. https://docs.oasis-open.org/cti/stix/v2.1/os/stix-v2.1-os.html

O'Donnell, L. (2020, January 3). *Ransomware attack topples telemarketing firm, leaving hundreds jobless*. Threatpost. https://threatpost.com/ransomware-attack-topples-telemarketing-firm/151530/

Rahmonbek, K. (2025). *35 alarming small business cybersecurity statistics for 2025*. StrongDM. https://www.strongdm.com/blog/small-business-cyber-security-statistics

Spambrella. (2020, February 27). *Michigan practice Brookside ENT closes doors following ransomware attack*. https://www.spambrella.com/michigan-practice-brookside-ent-closes-doors-following-ransomware-attack/

Strom, D. (2025). *Threat intelligence platform buyer's guide: Top vendors, selection advice*. https://www.csoonline.com/article/3984720/threat-intelligence-platform-buyers-guide-how-to-pick-the-best-platform.html

TrustRadius. (2025). *Pricing Overview for CrowdStrike Falcon.* https://www.trustradius.com/products/crowdstrike-falcon/pricing

Tsanev, V. (2024). *Cost of cybersecurity for small businesses in 2025*. https://www.execweb.com/post/cost-of-cybersecurity-for-small-businesses

Vendr. (2025). *Recorded Future Software Pricing & Plans 2025: See Your Cost.* https://www.vendr.com/marketplace/recorded-future

Appendix A: Figures

Figure 1

AWS CLI output - API Gateway /search endpoint test

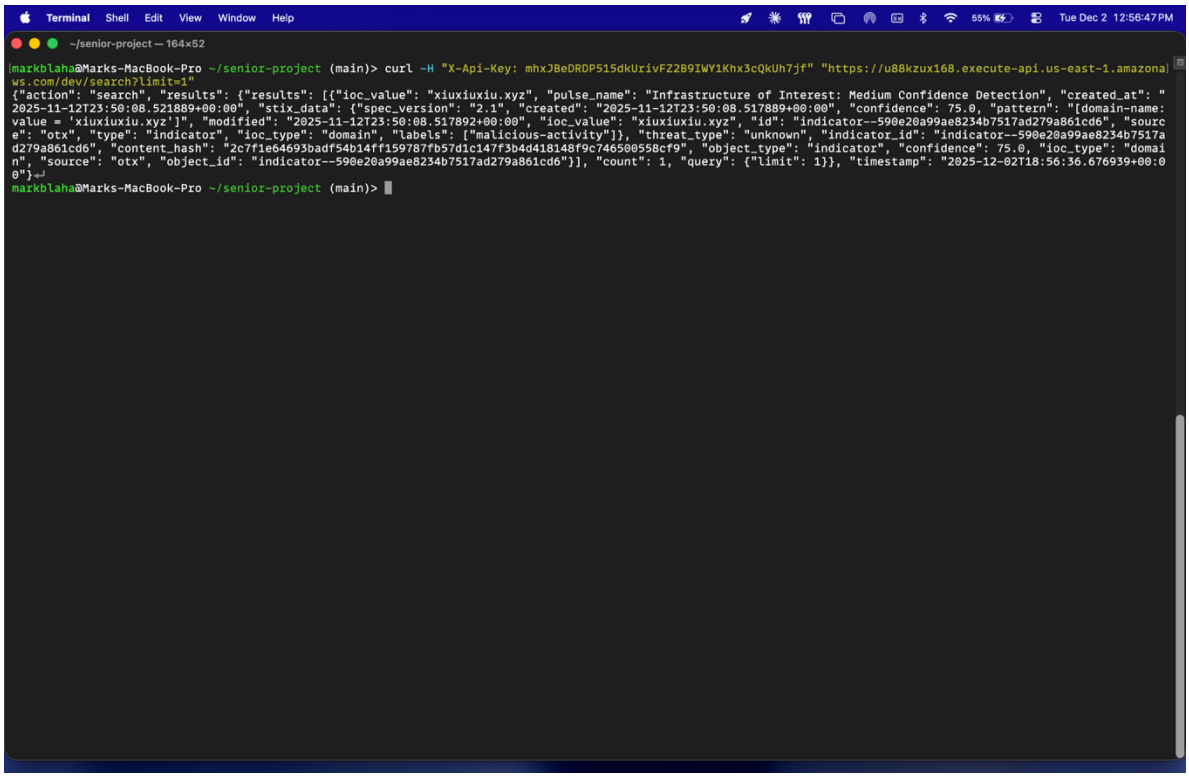

Figure 2

AWS CLI output - API Gateway /enrich endpoint test

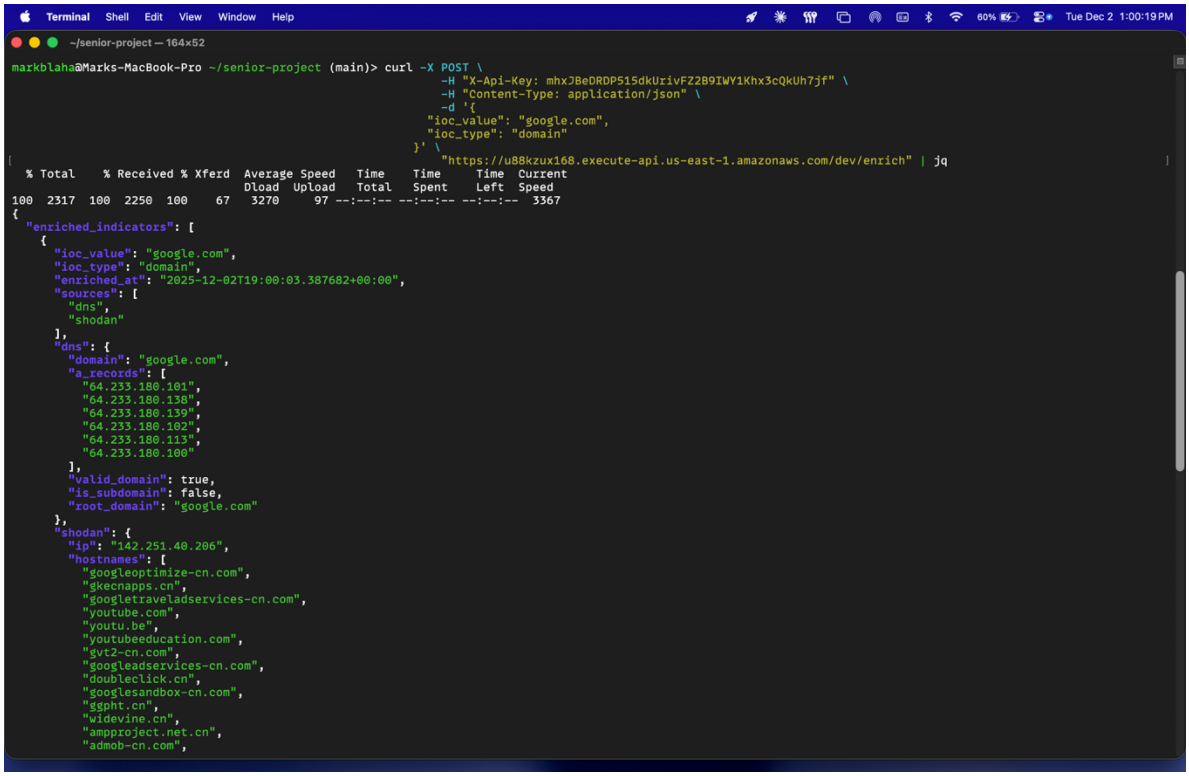

Figure 3

AWS CLI output - DynamoDB manual ItemCount, TableSizeBytes, TableStatus test

Figure 4

Live website displaying raw STIX 2.1 data

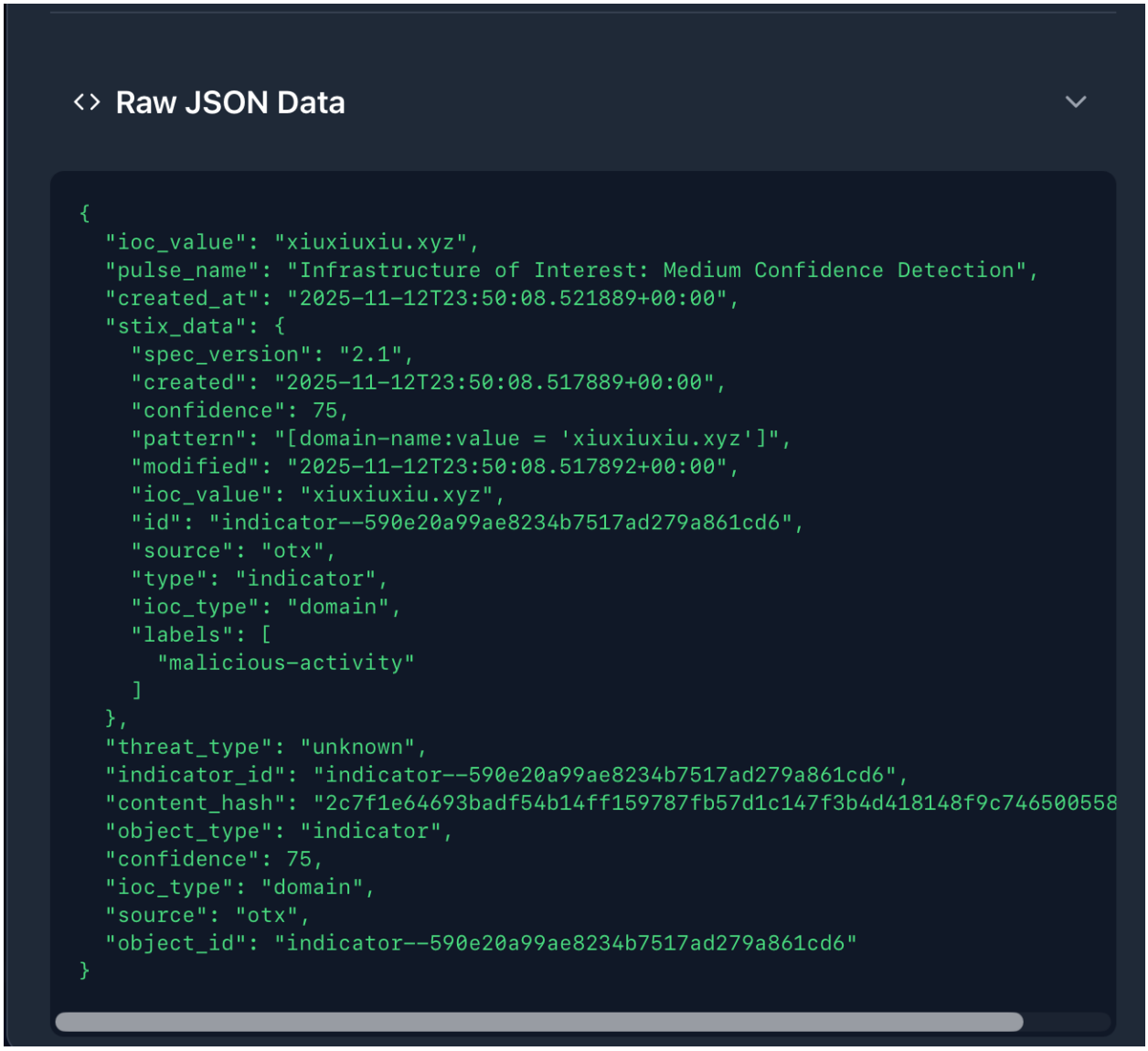

Figure 5

AWS CloudWatch log stream displaying successful /collect API call

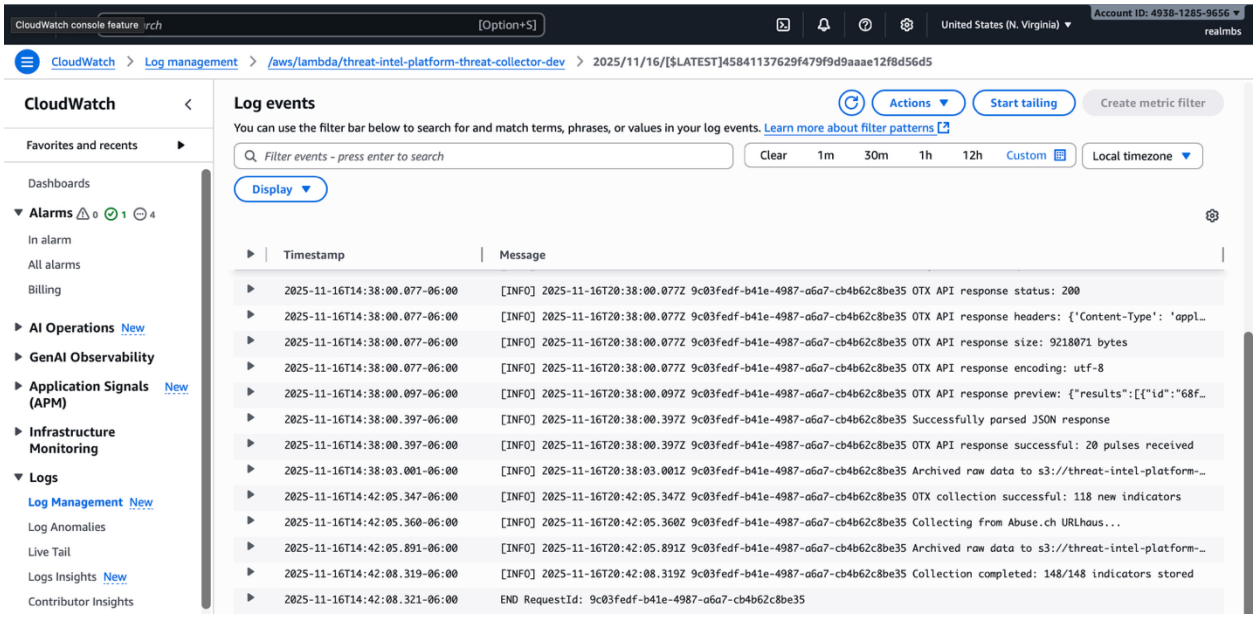

Figure 6

AWS Billing displaying pricing model validation after 1 month of continuous operation

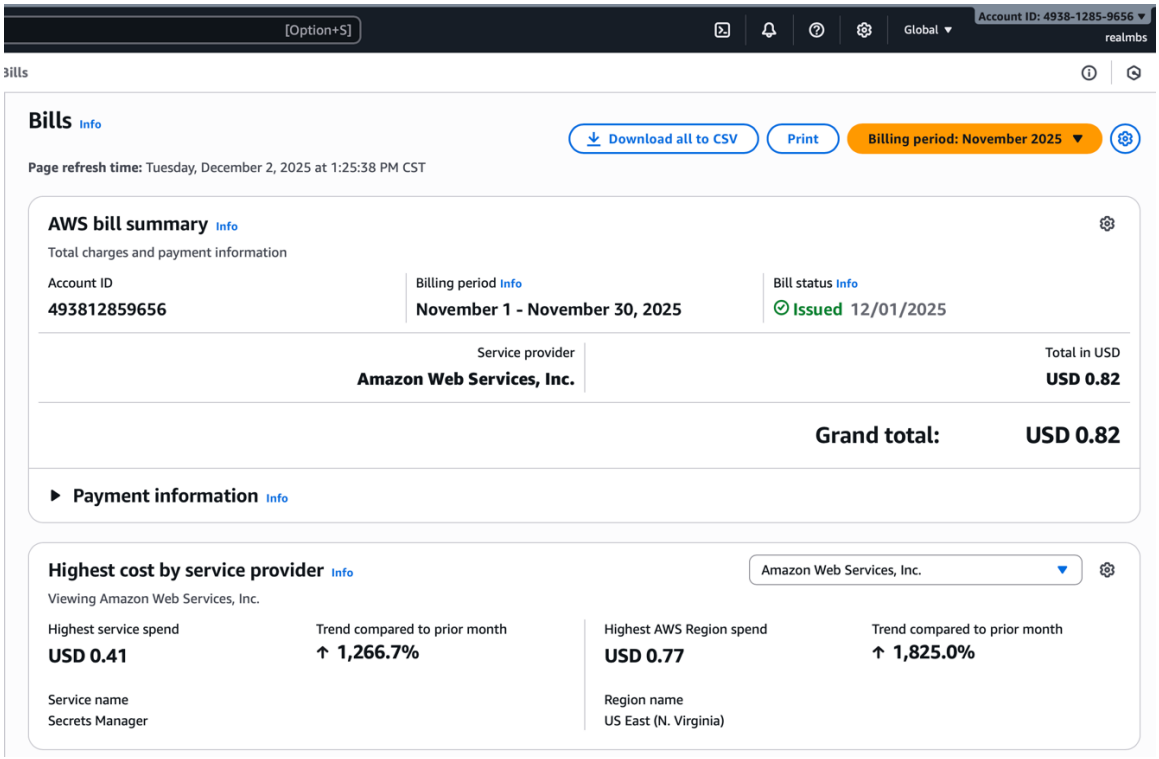

Figure 7

Live website displaying project dashboard

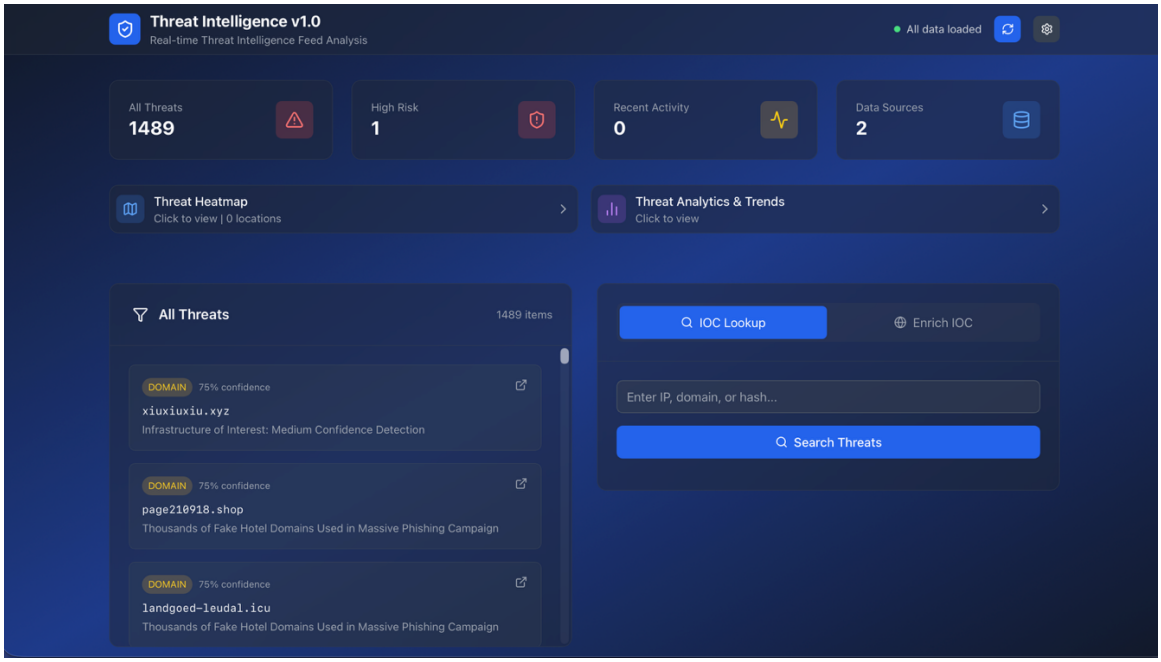

Figure 8

Live website displaying IOC information

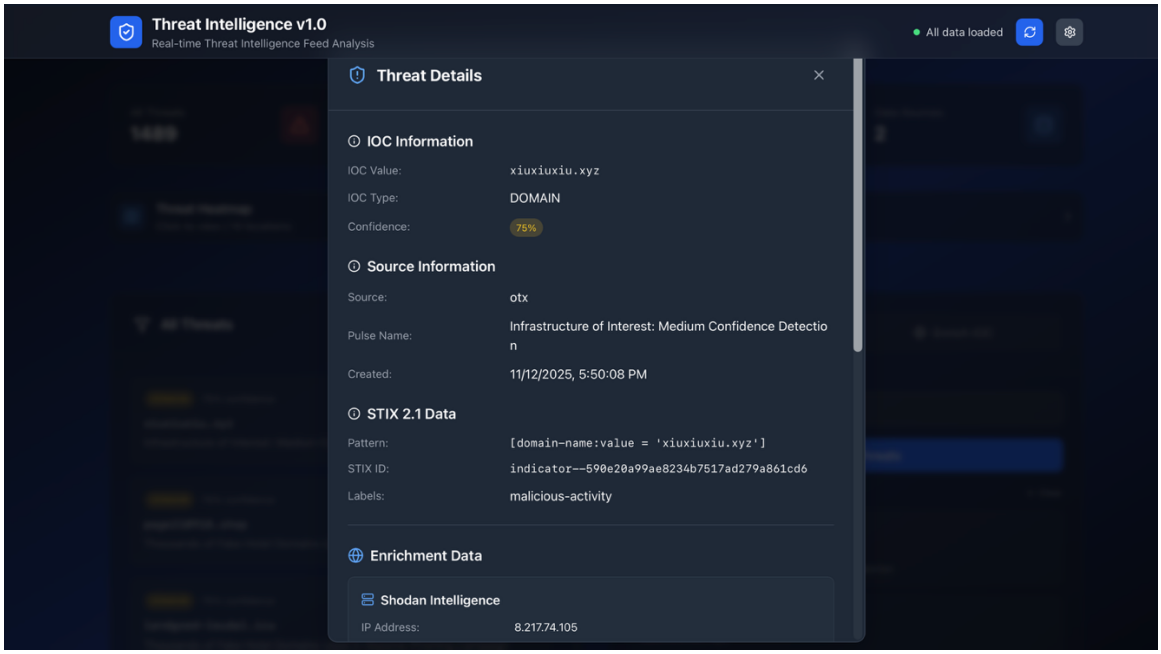

Figure 9

Live website displaying threat heatmap

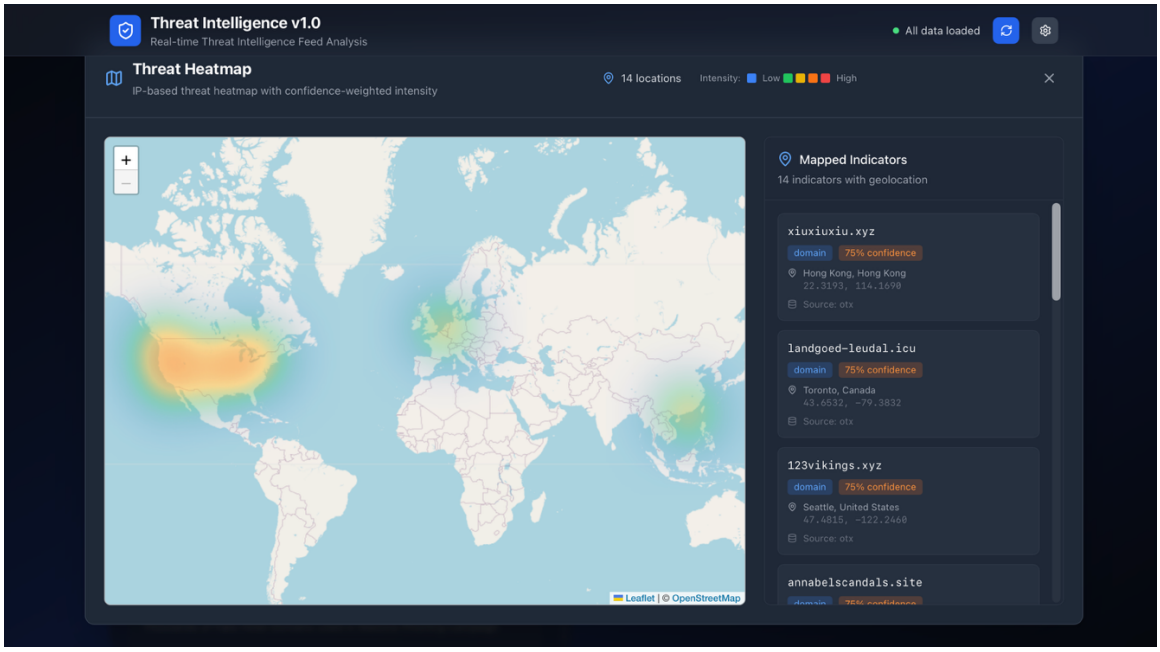

Figure 10

Live website displaying IOC distribution

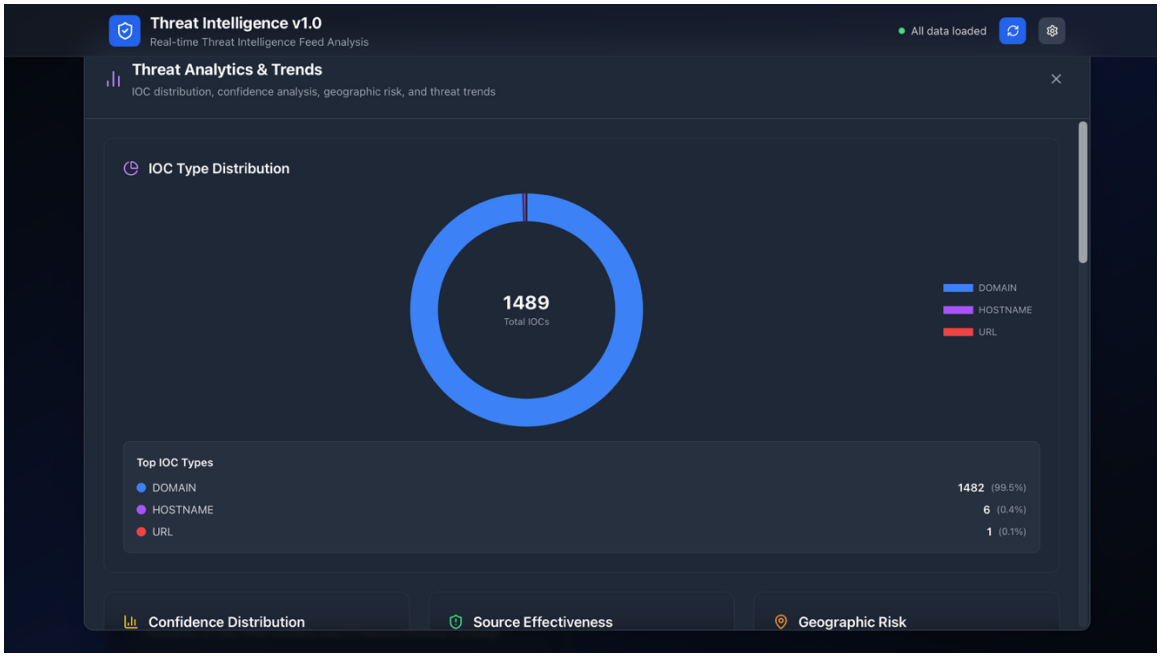

Figure 11

Live website displaying confidence distribution, source effectiveness, geographic risk