AFRL-RI-RS-TR-2023-186

ASED-EDGE

DATA MACHINES CORP.

OCTOBER 2023

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nations. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2023-186 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /
MICHAEL J. MANNO
Work Unit Manager

/ S /
SCOTT D. PATRICK, Signed for
MATTHEW KOCHAN
Technical Advisor
Intelligence Systems Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
		START DATE	END DATE
OCTOBER 2023	FINAL TECHNICAL REPORT	FEBRUARY 2021	MAY 2023

4. TITLE AND SUBTITLE
ASED-EDGE

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
FA8750-21-C-0180	N/A	DoD, DARPA

5d. PROJECT NUMBER	5e. TASK NUMBER	5f. WORK UNIT NUMBER
		R348

6. AUTHOR(S)
Eric Whyne

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Data Machines Corp.
Suite 110, 44933 George Washington Blvd
Ashburn VA 20147

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)
Air Force Research Laboratory/RIED 525 Brooks Road Rome NY 13441-4505	AFRL/ RI	AFRL-RI-RS-TR-2023-186

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The goal of the Active Social Engineering Defense (ASED) program was to automatically identify, disrupt, and investigate spear-phishing and social engineering attacks. The core technology developed in this program was the capability to automatically elicit information from a malicious adversary to identify, disrupt, and investigate social engineering attacks. ASED attempted to do this by mediating communications between users and potential attackers with bots that actively detect attacks and coordinate investigations to discover the identity of the attacker.

15. SUBJECT TERMS
Machine Learning, Social Engineering, Data Pipelines, Workflow Management

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT	b. ABSTRACT	C. THIS PAGE		
U	U	U	SAR	24

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
MICHAEL MANNO	N/A

Table of Contents

List of Figures

List of Tables

1.0 Summary

The program defined three technical areas (TA). TA1 provided automated detection of social engineering attacks, led by USC-ISI. TA2 provided automated investigation of social engineers, led by SRI. TA3 provided the scalable evaluation team, led by Thomson-Reuters Special Services. Data Machines Corp (Data Machines, or DMC) played a critical role in ensuring program success by providing a meta-framework for running machine learning (ML) research, a bespoke evaluation environment suite, and technical support to performers and potential transition partners.

Initially, Data Machines provided the compute infrastructure, version control, ticketing, mail servers, and basic admin services needed to run evaluations and operate the meta-framework. As program needs grew, DMC successfully scaled the number, size, and composition of the evaluation environments to ensure timely, successful completion. This allowed performers to grow the capacity and complexity of their solutions while still maintaining firm project timelines.

Data Machines also provided advice to the TA3 team on a range of technical issues. DMC introduced a separate orchestration service to handle the increasing number and complexity of performer deployments. When TA2 and TA3 performers needed additional GPU resources, Data Machines deployed dedicated GPU resources for them to use. And, when existing solutions did not exist, DMC provided software development support in the form of bridge services, messaging adapters, message exchange libraries, and client interfaces.

2.0 Introduction

The goal of the Active Social Engineering Defense (ASED) program was to automatically identify, disrupt, and investigate spearphishing and social engineering attacks. The core technology to be developed in this program was the capability to automatically elicit information from a malicious adversary to identify, disrupt, and investigate social engineering attacks. ASED attempted to do this by mediating communications between users and potential attackers with bots that actively detect attacks and coordinate investigations to discover the identity of the attacker.

3.0 Methods and Assumptions, and Procedures

Meta-Framework

Data Machines worked with the TA groups to define and implement a meta-framework toolkit for Machine Learning (ML) research. To achieve the highest degree of flexibility for performers integrating with the system, we chose a loosely coupled, asynchronous messaging approach built upon the Apache Kafka streaming data platform.[1] Individual components were packaged as Docker containers by Gitlab[2] and deployed through a continuous integration (CI) pipeline, and deployed to a hardened, RBAC-enabled Kubernetes[3] environment by Concourse[4] using a standardized continuous delivery (CD) pipeline.

We defined a standard interface in the form of a Python library through which all interactions with the messaging system occurred. The library contained all interaction primitives, message handling logic, authentication, and domain model abstractions required for performers to interact with the system. This abstraction allowed performers to combine and compose their technologies in the manner that best suited their needs without enforcing excessive schema, packaging, or deployment restrictions upon them.

As the system evolved, additional third-party services were introduced as dependencies to the evaluation environment. To streamline interaction with these services, we introduced RESTful APIs for SMS, email, and managed data transfers. These APIs shared a common identity, authentication, and authorization management mechanism (IAAM) with the Python library.

The project Gitlab Docker Repository served as the defacto library of independent and uniquely functional technology components. Performers committed their source code to Gitlab where it underwent CI testing and was packaged into a Docker container image. All images built by the

[1] Apache Kafka. https://kafka.apache.org/
[2] Gitlab CI. https://about.gitlab.com/solutions/continuous-integration/
[3] Kubernetes. https://kubernetes.io/
[4] Concourse CI. https://concourse-ci.org

system were tagged using a branch and commit hash strategy that allowed every container running in the system to be traced back to the exact version of the source code from which it was produced. Promotion gates in the CICD workflow allowed formal release tags to be created for each component before the start of an evaluation. This allowed us to properly version each component as well as each performer team's technology stack as a whole for each evaluation.

Evaluation Environment Suite

Data Machines utilized the meta-framework to develop an evaluation environment suite based on a self-service research and development platform with full implementations of access control, security, user identity management, audition, and encryption. We used a centralized FreeIPA[5] server to supply identity management across the project. A private Gitlab instance hosted project source code control. Gitlab Runners supplied continuous integration for each repository. Formal software documentation was written in ASCIIDoc, then compiled into HTML and served as static HTML by Antora[6]. Continuous deployment of performer code to Kubernetes was performed by Concourse. MinIo[7] provided secure data transfer services across performer environments. Kafka provided data ingestion to enable data analysis of communication data including emails and SMS texts.

Data Machines secured several third-party services to provide critical functionality to the project. Google Workspace and Office 365 accounts provided predefined email accounts used for evaluations. Twilio[8] and Plivo[9] provided SMS functionality to performers and a pool of reserved phone numbers to use during evaluations. Amazon Web Services (AWS) provided domain name resolution services, serverless functions, message queues, and notification services to filter, transform, and forward incoming email to multiple destinations.

Through the synergistic use of these technologies, a wide pool of developers and statistical/analytic researchers had the tools to safely iterate on sensitive data sets to derive novel methods, code, and technologies. Researchers were able to quickly iterate on the development of their approach and deploy their code securely into their own secure, namespaced tenant within the evaluation environment through automated CI/CD pipelines. This allowed for significantly easier sharing of newly developed technologies via inherently secure, authenticated interfaces.

Data Machines used the multi-tenant design of the evaluation environment suite to facilitate transitional activities starting in year 2 of the project. Using a second, replica environment, the existing research codes and services could be deployed into an isolated environment where research transition partners could access the prototype solutions with full reporting, metrics, auditing, access control, and management interfaces available.

---

[5] FreeIPA. https://www.freeipa.org/page/Main_Page
[6] Antora. https://antora.org/
[7] MinIO. https://min.io/
[8] Twilio. https://www.twilio.com/
[9] Plivo. https://www.plivo.com/sms/

4.0 Results and Discussion

Validation of Research and Transition Engagements

To maximize the potential utility of the various ML technologies produced by the performers, we developed a novel, message-driven execution environment based on Kubernetes and Kafka that allowed the TA3 team to assess and validate distributed execution of the performer's ML technologies on a range of hardware to ensure functionality, accuracy, and readiness. Specifically, this allowed the TA3 team to run evaluations across three different evaluation environments, seamlessly scaling as needed to meet the bursting requirements of each evaluation.

Integrative Services

The choice to use a message-driven architecture allowed us to satisfy the conflicting technical requirements of the performer teams while allowing us to scale the capacity of the datasets and infrastructure for demonstrations and evaluations. To bridge the gap between our asynchronous message-driven architecture and the paradigms most familiar to each performer team, we developed several integrative services described in the following sections.

Gmail Retriever

We developed the Gmail Retriever system to provide the researchers with a constant source of emails to be analyzed. The system utilized a RESTful API gateway and AWS Lambda functions to automatically transform and feed emails from one of the reserved Google domains to a Kafka queue. The routing choice is based on an email registration created manually by performers using a custom web application developed to manage the relationship. From Kafka, researchers were able to retrieve the emails for their activities.

Android SMS Application

Data Machines developed an Android application that served as a bridge between Android text messages and the ASED evaluation environment suite, primarily aiming to upload both incoming and outgoing text messages to the evaluation Kafka server. The application allowed researchers to schedule batch uploads when devices retained connectivity and reply to incoming messages with ASED-generated responses. Providing this tool allowed researchers to interactively exercise their ML algorithms using the full end-to-end evaluation suite and quickly demonstrate their solutions.

SMS API and Web UI

To examine SMS attack surfaces across platforms like Plivo and Twilio, we integrated synthetic phone numbers via a RESTful API that researchers could query directly from their applications. The API leveraged the internal authentication and authorization protocol of the evaluation platform and ensured data isolation between performer teams.

Messaging Interface

We developed a Python library to abstract interactions with the message queues and ease data handling. For the data to be persisted and analyzed later on by the evaluation team, we created a custom Kafka client adapter to persist data to a PostgreSQL database. This adapter and resulting data were critical in supplying data for the custom Superset dashboards we created to visualize evaluation results.

Collaborative Services

We deployed the following collaboration and development tools which support the agile iterative development of technologies.

Gitlab

A private Gitlab server (gitlab.ased.io), Gitlab Docker Registry (registry.ased.io), and dedicated Gitlab Runners were provided to project participants to allow them to collaborate on code, track issues, manage project deliverables, and implement software best practices around continuous integration and code quality review. Gitlab accounts were tied to ASED user accounts through the project's FreeIPA server.

Nexus

A private Nexus[10] server (nexus.ased.io) was deployed to host secure Python and NPM images, provide a secure PyPi mirror, and host raw asset files needed for image builds during the CI process. Nexus accounts were tied to ASED user accounts through the project's FreeIPA server.

Slack

A private Slack team (ased-hq.slack.com) was created for the ASED project and made available to users by invitation only.

ScoreStream

To streamline notifications for performers when a new score becomes available, we developed an automated Slack plugin. This plugin alerted users in a designated Slack channel whenever a new score appeared on the Kafka stream.

Kafka UI

There was a program-wide need for comprehensive visibility into the granular message and report data stored on the Kafka system. To address this, we developed and

[10] Sonatype Nexus. https://www.sonatype.com/products/sonatype-nexus-repository

deployed Kafka-ui[11] and KOWL[12], a Kafka workload management UI, granting both performers and the DARPA government team the necessary insights.

Data Transfer Service

Data Machines deployed MinIo, a high-efficiency object storage system, to facilitate seamless data transfer for the participants. This platform was employed to transfer data between both internal and external cloud infrastructures. Operated from a single server, it enabled the ASED sub-program CORA to transmit data between servers utilizing the MinIo CLI tool. This mechanism proved particularly advantageous for managing large sets of data within the CORA program. Data Machines provided documentation and hands-on help to enable this system for CORA performers.

5.0 Conclusions

Performer Support

Data Machines diligently deployed their resources to assist ASED performers in multiple facets of the platform. This included guidance on platform utilization, streamlined user onboarding procedures, VPN support, as well as meticulous access control protocols. Special attention was directed towards performers with a limited background in platform technologies, for whom one-on-one help was provided to ensure proficient system usage. Furthermore, the team at Data Machines promptly addressed system anomalies, from hard drive failures to data transfers, ensuring optimal hardware performance at all times. Additionally, Data Machines facilitated the extraction of data for users as and when required.

Hardware Procurement and Deployment

Data Machines was responsible for the procurement and maintenance of the data center infrastructure for ASED. Data Machines repurposed hardware from previously concluded DARPA programs, notably the D3M program, and transitioned this hardware for use in the ASED initiative. Additionally, Data Machines sourced hardware from external vendors, ensuring its seamless deployment and usability for the program. A proactive approach was adopted for hardware failures, with swift repairs and replacements as necessary. In the event of critical issues, DMC ensured a smooth recovery process. On multiple instances, we had redundant backup clusters prepared to mitigate the risk of system downtimes. Backup systems were maintained for GitLab, Kafka, Wiki documents, and more. Those containing real user data, and Gitlab codes were encrypted and stored in a manner to maintain the necessary security classification requirements.

[11] Kafka-UI. https://github.com/provectus/kafka-ui
[12] KOWL. https://github.com/theurichde/kowl

Transition Support

Data Machines chose to build the ASED evaluation environment upon open source and open standards to enable an easier transition to wider DOD cloud infrastructure, potentially at higher classification levels. All messaging, processing, routing, and infrastructure services are available as hosted cloud services in federal regions of every major cloud provider. The third-party hosted services such as email, SMS, and webhook processing leverage standards and protocols readily available by most major service providers as well as numerous open source projects. Performer codes were built and packaged such that they can run with any recent Kubernetes runtime, and the identity management system can be lifted and shifted directly to the cloud, or swapped out with any LDAP-compliant alternative.

While the mandate was not given to transition the project to a specific transition partner, we made every effort to produce a product that will ease the transition at a future time while ensuring both the functionality of the research system and its security without creating vulnerabilities in existing military operations.

Additionally, performers from the CORA sub-program within ASED received comprehensive support for utilizing the ASED system. Several GPU servers were transitioned to the UMD-ARLIS lab from ASED and CORA programs. Our team facilitated the data transfer process and ensured a smooth transfer of GPU server hardware to align with the program's objectives.

Documentation

Every effort was made to ensure the security of platform components and provide complete lifecycle National Institute of Standards and Technology Risk Management Framework (NIST RMF) support with system registration, documentation development, Information Assurance (IA) control baselining, and validating and completing mandated IA requirements.

The system and its components were documented at several levels. Each component repository contained a README document describing its purpose, usage, development, and packaging. Higher-level system documentation was generated and served from an Antora web server as static HTML (docs.ased.io). Overall project information, including design decisions and system interaction information, was documented in a private Confluence server.

Appendix: Evaluation Environment Infrastructure

Kubernetes Clusters

DMC deployed Kubernetes clusters to host the containerized ASED performer services. We chose Kubernetes because of its ability to provide a secure, multi-tenant environment in which to host the necessary infrastructure services for multiple performer groups while ensuring judicious resource utilization, transparent monitoring, integrated storage and identity management, and detailed accounting.

We used the Ansible-based Kubespray[13] software with a customized configuration to deploy the ASED Kubernetes cluster. The customizations consisted of enabling Role Based Access Controls (RBAC) and OpenID Connect (OIDC) authentication using the project LDAP server, establishing preconfigured storage classes backed by Ceph, and default namespaces and service accounts created for performers for each evaluation. This configuration allowed the project to reduce its physical footprint by consolidating the compute and storage infrastructure onto fewer physical nodes while still ensuring fault tolerance and reliability.

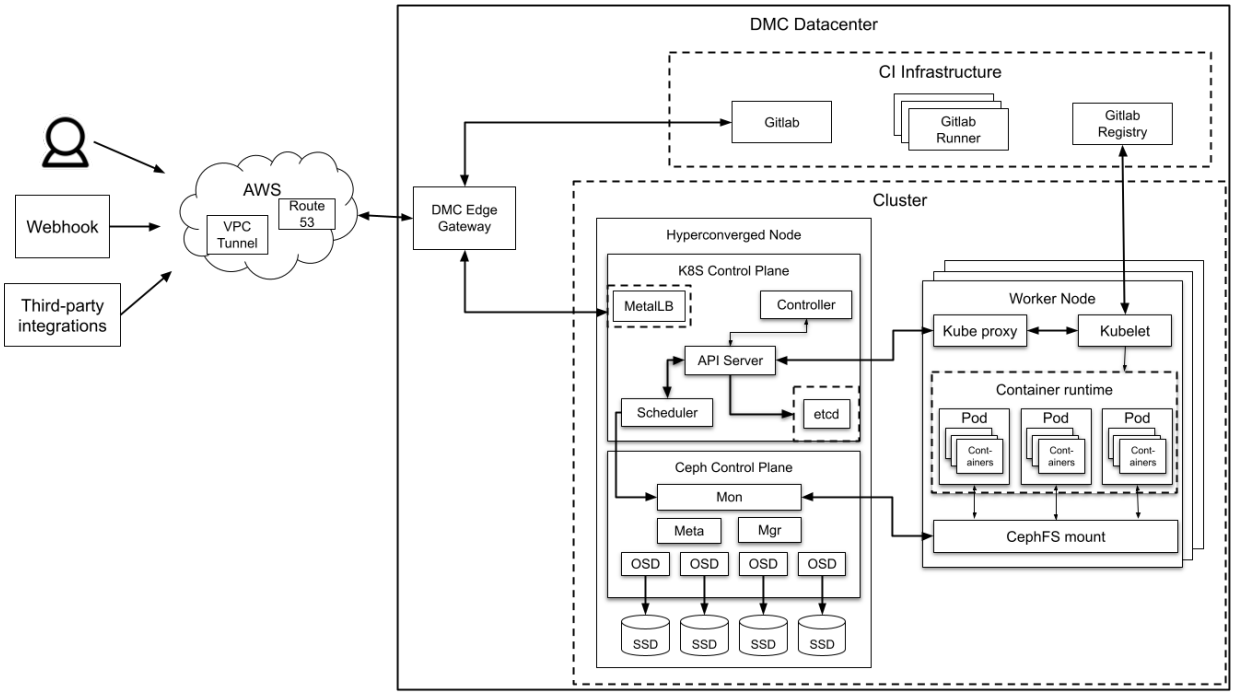

Figure A-1 The high-level architectural diagram for the ASED Kubernetes clusters. Each cluster had distinct hardware with a common architecture and configuration.

Kubernetes also provided the necessary infrastructure upon which we monitored and supported system software development through log aggregation, metric collection, and system

[13] Kubespray. https://github.com/kubernetes-sigs/kubespray

monitoring. Our ground-up approach to system monitoring allowed us to proactively assist performers through pain points developing and deploying their code into production. It also allowed us to provide better insight into the progress of each evaluation, both on the expected time to completion of each performer and on the resources required for them to complete. This allowed us to proactively adjust resource availability for the next evaluation based on past performance.

Data Machines deployed a single Kubernetes cluster at the start of the project. Throughput concerns before the Fall 2021 evaluation led DMC to procure additional hardware and deploy a second cluster. DMC then deployed a third cluster before the Summer 2022 evaluation to support potential transition activities and hedge the program's risk of bottlenecks appearing during the evaluation. Each cluster had dedicated physical hardware with identical software stacks. A breakdown of the three systems is provided in Table 1.

Table A-1 Description of the 3 Kubernetes clusters deployed over the life of the project

	Alpha	**Bravo**	**Charlie**
CPU Nodes	3	5	3
CPU Cores / Node	72	48	48
Memory / Node	508 GB	512 GB	256/512
GPU Nodes	3	1	2
GPU Model	NVIDIA GeForce GTX 1080	NVIDIA A100	NVIDIA A5000
GPU / Node	4	4	4
GPU Memory / Node	12 GB	12 GB	24 GB
K8S Version	1.20.7	1.20.7	1.20.7
Ceph Version	1.14.3 (Nautilus)	1.14.3 (Nautilus)	1.14.3 (Nautilus)
Kafka broker	baremetal	baremetal	baremetal
Services	Dex, Grafana, Loki, Prometheus, MinIo, MetalLB, KubeDB	Dex, Grafana, Loki, Prometheus, MinIo, MetalLB, KubeDB	Dex, Grafana, Loki, Prometheus, MinIo, MetalLB, KubeDB

Kafka Data Collection and Exchange System

The Kafka message queue system was critical for the ASED evaluation process, acting as the backbone that enabled the exchange, collection, and analysis of extensive message data. To ensure seamless integration and accommodate the volume and complexity of data, multiple clusters of the distributed Kafka system were established by Data Machines. A joint effort between Data Machines, performers, and the DARPA government team led to the development of a system specification. This initiative resulted in standardization throughout the system, ensuring uniformity and compatibility of the message format.

The Kafka system was instrumental for the ASED framework, serving as a foundational element that facilitated performers in storing, retrieving, tracking, and responding to multifaceted message data. The Kafka system encompassed not only maintaining data communication but also reporting usage, flagging dialogues, and queuing data, thereby magnifying the scalability of the performer's systems for program-wide data ingestion.

To facilitate this integration, Data Machines created several Python packages, making it easier for performers to retrieve, publish, and monitor text messages within the Kafka infrastructure.

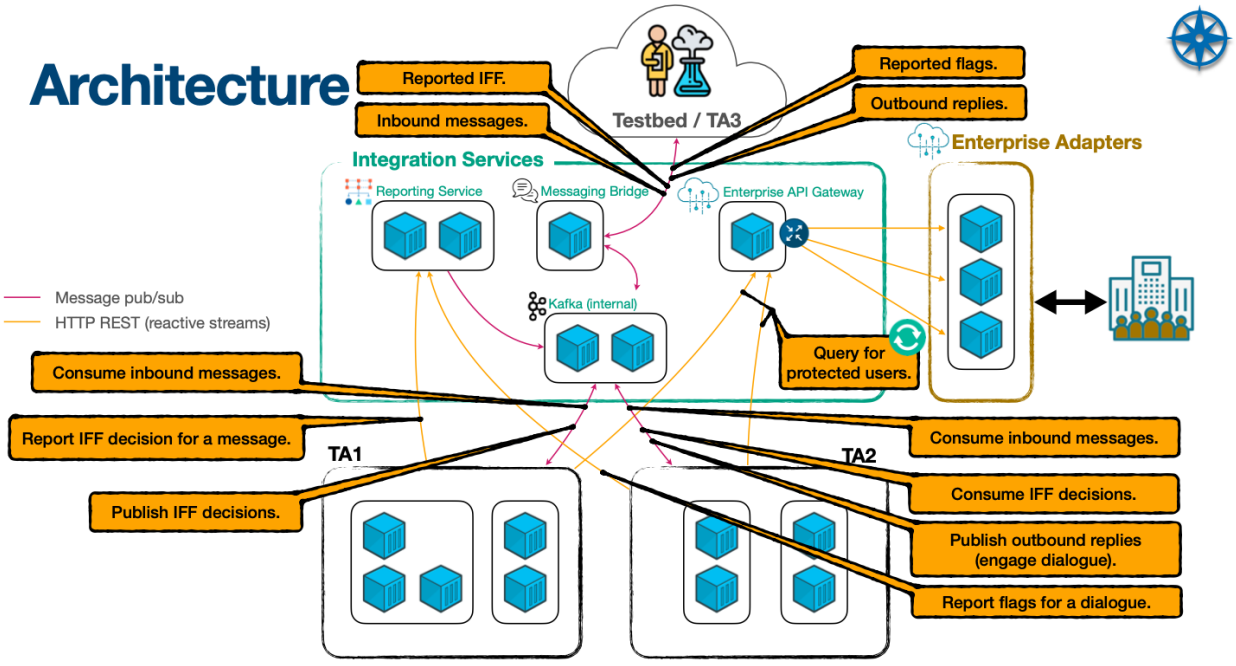

Figure A-2 A high-level diagram showing the interaction between different components such as Gmail retriever, Android SMS mobile app, Web UI, Kafka, Minio, and k8s clusters.

Technical Challenges

A technical challenge we encountered during the initial cluster deployment centered around running Kafka on Kubernetes. At the start of the project, the Kubernetes community

recommended not to run stateful service on Kubernetes. Through a combination of experimentation and hardware tuning, we achieved both a reliable and high-performance deployment of Kafka on our Kubernetes cluster. However, due to the performers' desire to leverage LUKS volumes (a type of encrypted volume), we migrated all Kafka instances to OpenStack virtual machines.

OpenStack Private Cloud

All ASED virtual machines were provided by DMC's private OpenStack[14] cloud. The following services ran within the cloud environment:

- Reverse proxy
- PowerDNS[15]
- Gitlab
- Gitlab Runners
- Confluence
- Kafka
- Nexus

[14] OpenStack. https://www.openstack.org/
[15] PowerDNS. https://github.com/PowerDNS/pdns

Appendix: Integrative Service Development

Gmail Retriever

To provide the ASED researchers with a constant source of emails to be analyzed, the Data Machines team created a system to automatically feed emails to the evaluation environment suite. The system consisted of two primary subsystems. The first was a dedicated Google email domain that allowed researchers and other program members to create email accounts that would receive the emails. The second subsystem consisted of AWS services that periodically retrieved all emails from these accounts and forwarded them to Kafka for ingestion.

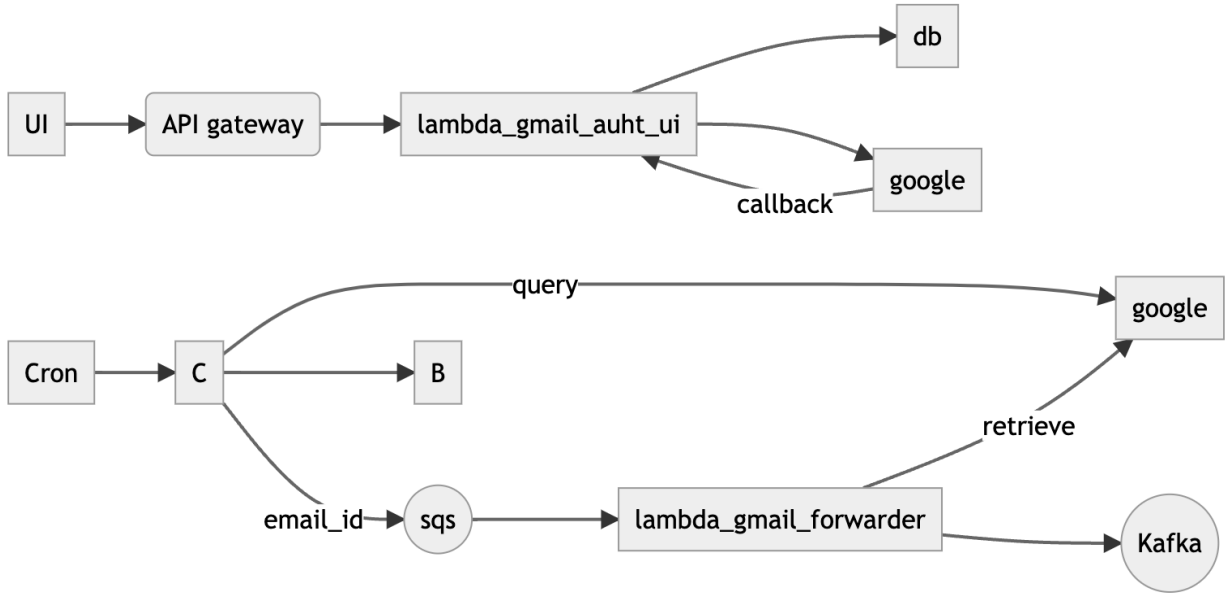

Figure A-3 Data flow diagram for the Gmail Retriever application developed by Data Machines.

To support researchers during development and various exercises, multiple Google domains were created. Researchers and participants were able to request email accounts in specific Google domains for their differing purposes. Once accounts were created, the first challenge was to allow the AWS subsystem access to all the email accounts so that it could pull emails to be forwarded. To address this issue, an ASED registration web application was developed and registered with the appropriate Google domain. With the ASED registration web application, an ASED participant was able to register with the ASED system and grant access to their email account. The registration web application also allowed participants to unregister with the ASED system.

Once participants were registered, the AWS subsystem was then able to access their email accounts. Using AWS CloudWatch events, an AWS lambda function was executed on a periodic basis, typically hourly. On execution, the AWS lambda function queried each email account registered and retrieved all the mails since the last execution. Because each account

could have many emails to be processed, each email was then placed on an AWS SQS queue for individual processing. This allowed for parallel processing of individual emails as multiple instances of another AWS lambda function were able to retrieve multiple individual emails and process them in parallel. The SQS consumer AWS lambda function retrieved the individual emails from SQS, converted the emails from the Google API format to the defined ASED format, and forwarded them to a preconfigured Kafka queue. From Kafka, the ASED researchers were able to retrieve the emails for their activities.

Android SMS Application

Data Machines developed an Android application that serves as a bridge between Android text messages and the ASED infrastructure system, primarily aiming to upload both incoming and outgoing text messages to the evaluation Kafka server. The app was developed with an intuitive interface, highlighting key functionalities such as collecting messages offline and scheduled uploads when data connectivity is present. Additionally, a distinctive feature allows for the generation of text replies using ASED-generated text messages. The coding structure comprises significant components that ensure efficient data collection, storage, and transmission, focusing on parameters like form, thread, data, date, and more. Complementing this description are screenshots and UI diagrams that provide a visual representation of the app's design and user flow, showcasing its seamless integration and ease of use.

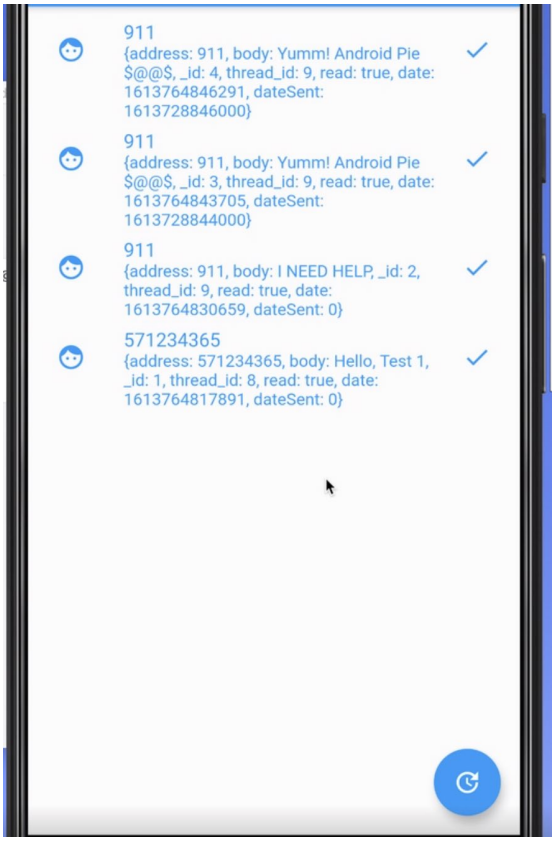

Figure A-4 Screenshot of the Android SMS application.

Web UI for Sending SMS

Data Machines developed a RESTful API to handle data flow from both the ASED Mobile App and synthetic SMS numbers from Twilio and Plivo. The API facilitated sending, receiving, and viewing message history through a single common interface. To demonstrate the API's functionality, Data Machines developed a minimalist web UI using Bootstrap.js and Socket.io.

The web UI became a convenient tool to demonstrate SMS data flow to stakeholders across the project.

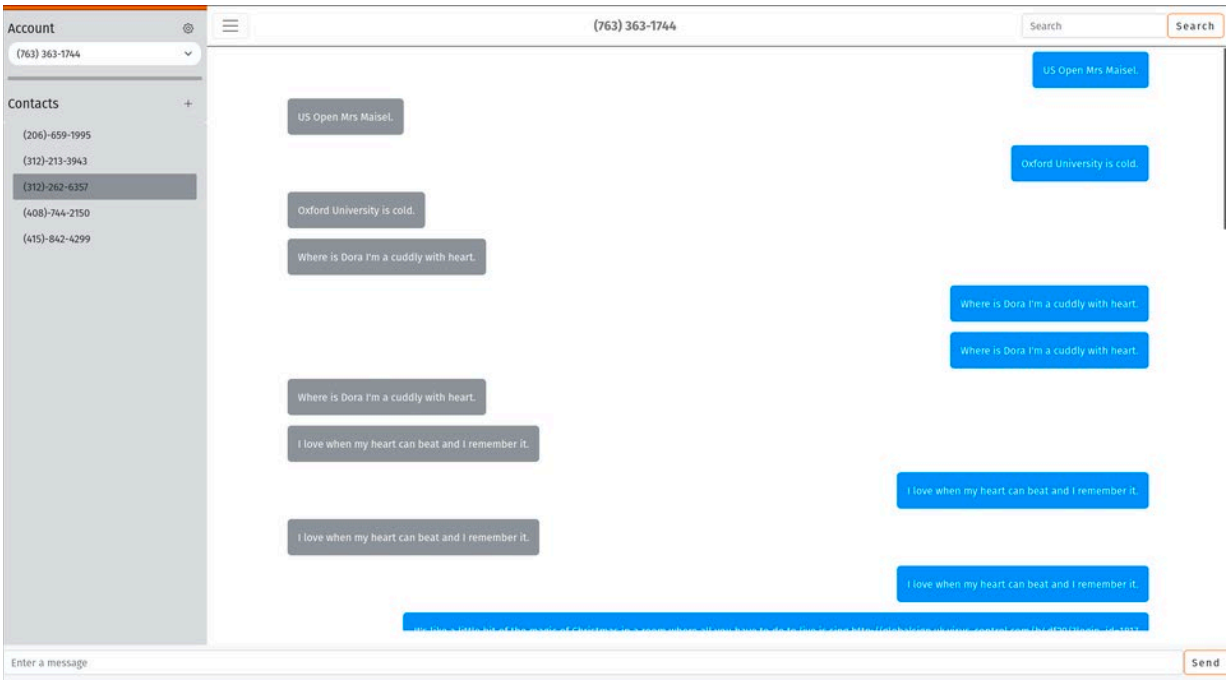

Figure A-5 Screenshot of the user interface used to exercise the RESTful SMS API.

Superset Data Analysis + Superset Dashboard

Data Machines utilized Apache Superset to generate a range of dashboards for ASED evaluation, powered by a PostgreSQL database. These dashboards included visual representations like charts, tables, and histograms, spotlighting critical data such as message counts per cohort email, data latency, precision, recall, daily attack messages, and other vital insights. Additionally, we integrated LDAP into the Superset dashboards to implement Role-Based Access Control (RBAC) security measures, safeguarding sensitive government-team level information from inadvertent access. Dashboards were utilized by the TA3 team to track and analyze the success of TA2 teams during evaluations and convey results during meetings and demonstrations.

Source Code Repositories

All our source code can be found in the `ASEDdmcGitlabSourceCode.tar.gz` file. The archive contains Gitlab exports of the codes produced by Data Machines. The original README files are included within each folder and offer detailed information about the respective project. The archived path and project description of each code within the repository is provided in the next table.

Table A-2 The locations and descriptions of Data Machines developed codes within the source code archive delivered with this report.

dmc/{project_directory}	**Description**
ased-canary	Set of CLI tools and Docker images for running processes that repeatedly pulse various ASED attack surfaces.
ased-evaluation-mirrormaker	Kafka Mirror Maker for Replication, Disaster Recovery
ased-evaluation-mirrormaker2	Kafka Mirror Maker for Replication, Disaster Recovery during phase 2
ased-faux-iff	Simulates certain data input conditions for testing purposes.
ased-kafka-middleware	Middleware utility for managing Kafka operations specific to ASED.
ased-mail-relayer	Manages and forwards email data, to spam identification during phase 2
ased-mail-reporter	Watches and sends reports based on mail data analytics.
ased-metrics	Collects and manages performance and data metrics for TA3
ased-mongodb	MongoDB Operator for Kubernetes and Example Deployment using Kustomize
ased-schema-converter	Transforms and adapts data structures to comply with ASED's specific schemas.
ased-sms-api	API for handling and ingesting SMS data within ASED.
ased-sms-relayer	Tool for processing SMS data
ased_metrics	Utility for gathering and analyzing metrics specific to ASED.

ased_slack	Tool to collect slack message data and publish to ASED system
asedsms	CLI Utility for posting synthetic SMS
cora_gpu_metrics	Monitoring GPU Performance for CORA Project with Docker-Compose and Prometheus Exporters
cp-ansible	Configuration and automation scripts using Ansible for standing up Kafka platform
evaluation-store-ingest	Utility for ingesting evaluation data into the evaluation store database
gmail-retriever	Extracts and processes Gmail data
kafka-tools	Set of tools for managing and optimizing Kafka operations.
kafka-ui	User interface for monitoring and managing Kafka streams and topics. Basic KOWL workload UI.
minio	Infrastrucure as code for standing up Object storage server-minio
promstack	Stack integrating Prometheus for monitoring and alerting of system health.
schemas	Collection of data structure definitions and templates for ASED.
scorestream	Utility for posting score data to Slack
sms-workflow	Workflow utility for processing, filtering, and analyzing SMS spam data.
superset	Code for Apache superset dashboard

Additonally, software developed by other perfomers are provided on the `/ops` and `/infrastrucure` directory of the deliverable source code archive file.

6.0 List of Symbols, Abbreviations, and Acronyms

ASED	Active Social Engineering Defense
USC-ISI	University of Southern California Information Sciences Institute
ML	Machine Learning
GPU	Graphics Processing Unit
API	Application Programming Interface
SMS	Short Messaging Service
IAAM	Identity, Authentication, and Authorization Management
CICD	Continuous Integration Continuous Delivery
HTML	Hyper Text Markup Language
NPM	Node Package Manager
UI	User Interface
VPN	Virtual Private Network
RMF	Risk Management Framework
IA	Information Assurance
LDAP	Lightweight Directory Access Protocol