MADDVIPR Prototype

Introduction

The core element of the MADDVIPR project is the MADDVIPR framework. Through the framework, we aim to consolidate and synthesize the information collected in the previous stages of our project, regarding single points of failure, vulnerabilities, and (D)DoS attacks against the domain name system (DNS) to produce actionable intelligence for DNS operators to detect, analyze and prevent DDoS attacks, and to improve infrastructure resilience.

Architecture

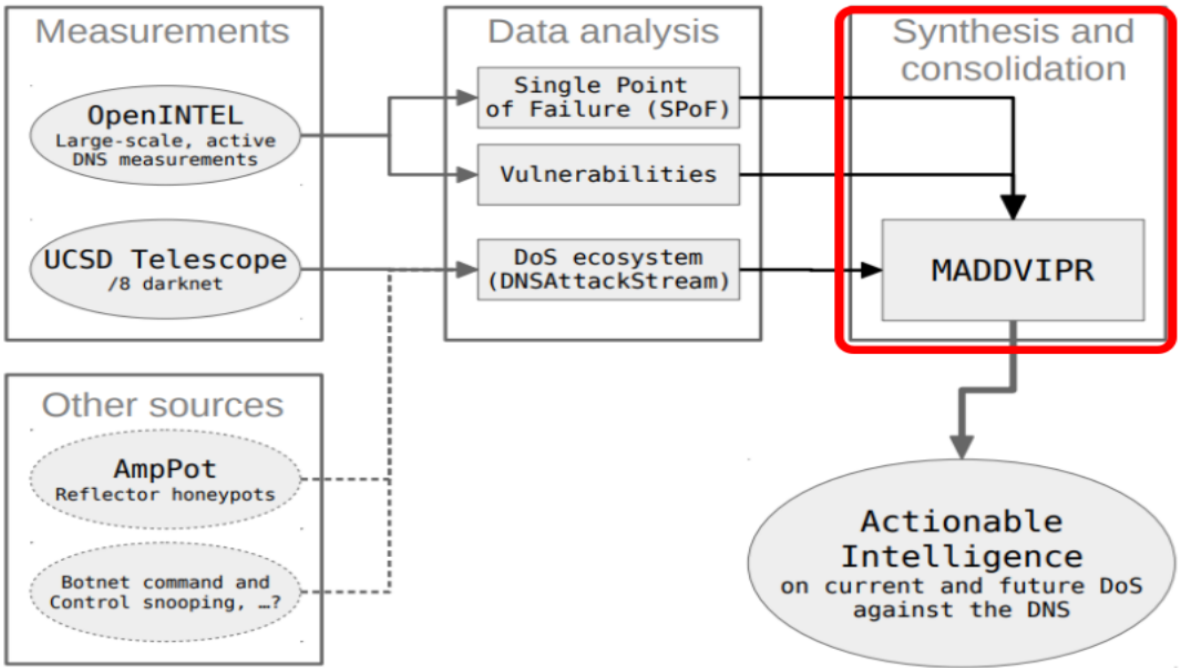

Figure 1. MADDVIPR Architecture

To achieve the main goal of the MADDVIPR Framework we will design two main components:

1. A highly configurable, reactive DNS measurement platform, designed to be scalable through a cloud-based, multi-tenant infrastructure, geographically distributed over the global Internet.
2. A variety of dashboards, fed with real-time data and intelligence, to provide operators insights into the current state of the DNS ecosystem and to help them to identify misconfigurations, vulnerabilities, and attacks.

In this design document, we put our focus on the first of the two main components.
We next outline potential data sources for our live DNS measurement:

1. DNSAttackStream -- fuses UCSD network telescope data with OpenINTEL DNS measurement data to provide a near real-time feed of attacks on DNS infrastructure. Responding to attack events with additional measurements provides opportunities to better understand the practical consequences and impact of attacks and helps identify working strategies towards improving resilience.
2. Extensible Provisioning Protocol (EPP) or zone file live updates -- Gathering near real-time changes to zone files or updates in EPP logs allows to proactively identify domain name related

misconfiguration, such as lame delegations and delegation inconsistency. Learning this as soon as possible can help operators guard resilience properties and assist in operating optimally working infrastructure.

3. Certificate Transparency (CT) logs -- public append-only logs that provide a real-time feed into issued TLS certificates and the associated domain names. Reactively measuring such domain names can help us identify malicious activity, such as potential hijacking, domain name squatting, and registrations towards malware distribution.
4. Amplification HoneyPot (AmpPot) -- a type of honeypot that catches reflection and amplification DDoS attacks. This provides a complementary view on attack activity for DNSAttackStream.
5. CommonCrawl -- a large-scale Web archive that recursively crawls hundreds of millions of domain names. The learned names can, among others, provide additional coverage on country-code specific top-level domains and domain names beyond the registered domain name (i.e., in lower levels of the DNS hierarchy).

To satisfy the requirement that the system is able to handle thousands of reactive measurements per second on distributed vantage points (VP) around the world, we decided to design and implement a message-broker-based solution, based on Apache Kafka.

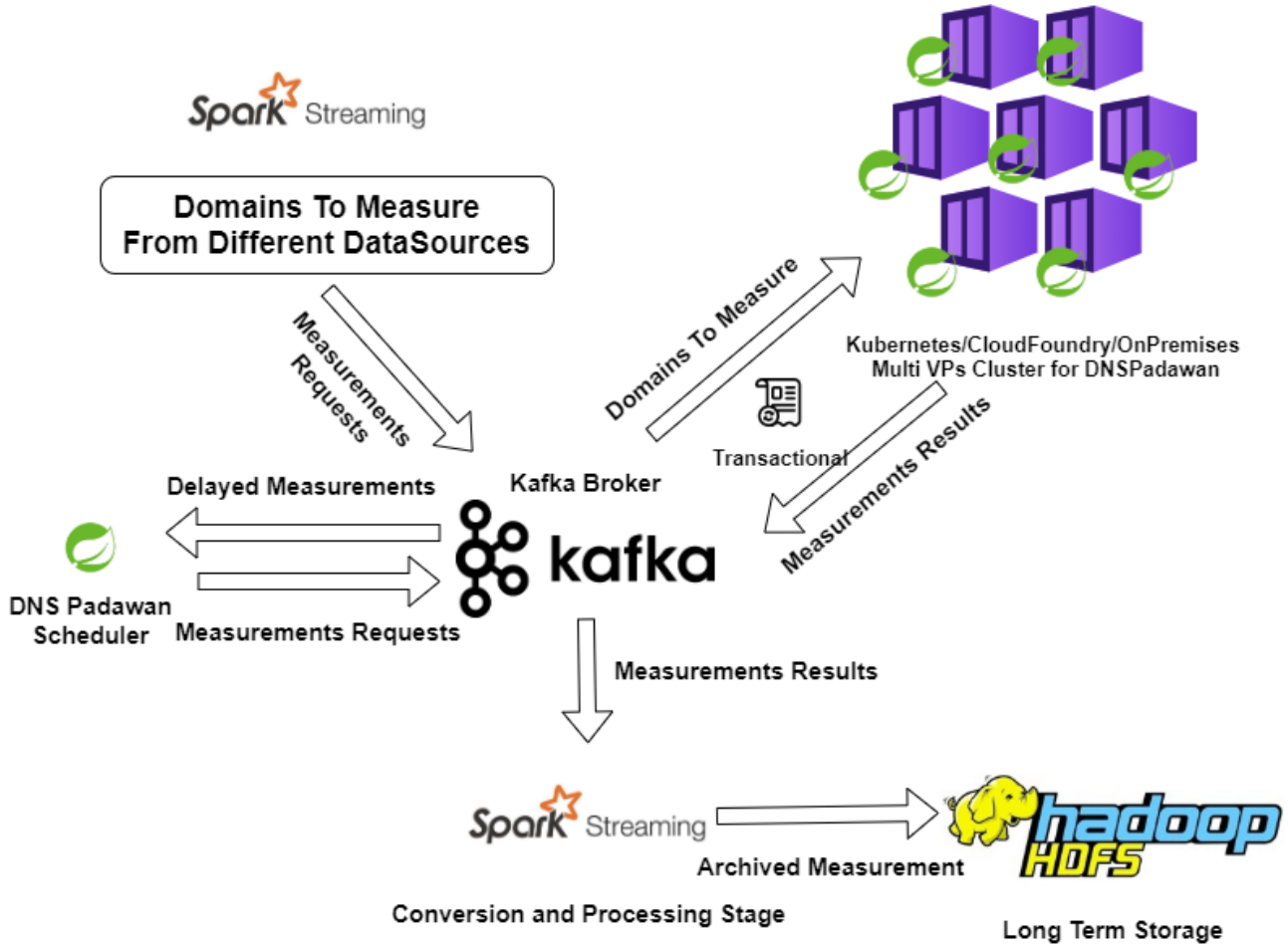

Figure 2. MADDVIPR measurement framework overview

In Fig 2, we illustrate the overall architecture of the measurement framework.

The domain names to measure, which are inferred from different data sources, are published live on a Kafka Topic by a streaming application (e.g., in Spark Structured Streaming). Then, we differentiate the measurements into two main types: *Immediate* and *Scheduled*.

Immediate measurements are performed in a best-effort approach as soon as possible by the measurement software (the working name of which is *DNSPadawan*) deployed on several platforms (Kubernetes, CloudFoundry, and on-premise's deployments).

Scheduled measurements are collected by the DNSPadawan scheduler component, which is responsible for scheduling an measurement at the requested time and for the requested number of iterations/repetition/times.

By using the proven Kafka Streams technology we can gain out-of-the-box embedded load balancing mechanisms across different VPs.

The DNSPadawan measurement and scheduler components are both implemented using the Java Spring framework. This approach allows us to obtain a solid system, natively interoperable with Kafka and several clouds orchestration technologies.

GitHub Repo: https://github.com/raffysommy/dns-padawan