Arabella

Arabella is a service, narrowly targeted to the needs of OSINT investigators, for producing iterations of existing FOSS tools.

Hackathon Note

This project was conceived in the wake of the Social Network Hackathon. All the work on this project, investigation, documentation and coding is documented in the github repository at https://github.com/jvrodley/arabella.git. The history of what work was done when can be seen there in its entirety. As per hack rule and tradition, all coding was left for the period of the hack itself. However, much of the prep-work including documentation and design was pre-loaded (see the github history). If that negatively affects our hack score, c'est la guerre, we're prepared to take the hit.

Background

In the course of the Bellingcat Social Network Hackathon several data points came together to inspire a system proposal that we're building a prototype of named Arabella.

1. In the Bellingcat survey of OSINT investigators (https://www.bellingcat.com/resources/2022/08/12/these-are-the-tools-open-source-researchers-say-they-need/) 23.4% of respondents reported being frustrated in the use of FOSS tools on GitHub. Reasons reported included things like "tool doesn't work at all", "no documentation" and "doesn't produce the output I need".
2. A quick, random survey of a half-dozen open source tools* produced the anecdata that two* of the tools didn't work at all, but could be (and were) fixed by a professional coder with a couple of lines of code in less than 2 hours of work.
3. In networking with the other participants, it became apparent that there is an untapped set of professional and semi professional coders and data scientists that:
    a. want to contribute to the work of OSINT but don't know what needs to be done that matches their skill set
    OR
    b. want to career-switch to OSINT but don't have a way in to the field
4. From the same survey, *developers who build tools for online researchers will currently only reach a quarter of our respondents if they make their tool only available via services like GitHub, rather than a web or desktop app with user interface.* Putting a user-interface on top of an existing command-line tool is not rocket-science and well within the reach of an enormous community of very junior developers but the investigators who need those user interfaces have no way to reach those developers.

Goals:

1. Solve tool problems for OSINT investigators in near real-time so that investigators can solve real-world problems more quickly
2. Give Bellingcat a way to tap the expertise of a wide range of people who want to contribute but have no way currently to do so in order to expand Bellingcat's investigative reach
3. Build a "bench" of technical talent that Bellingcat can draw on when recruiting for internal positions so that Bellingcat recruiting is more time-efficient and effective.
4. Give back to the FOSS community by PR'ing any tool fixes back to the original builder so that the system is not seen as parasitic or predatory and "working with Bellingcat" is seen as worthwhile
5. Build a set of tools that are
    a. known-to-work
    b. known-to-be-useful by Bellingcat investigators
    c. can be watched and proactively kept up to date so that investigators won't be surprised by tools that "don't work anymore".

Risks:

- R1 Supply chain attacks – abuse of the system would allow attackers to directly target Bellingcat through its tools rather than shotgun shoot-and-pray through tools Bellingcat might or might not use
- R2 OSINT investigator opsec – information about what individual investigators use is potentially exposed
- R3 System maintenance cost/effort
- R4 System operational cost/effort
- R5 Participant vetting cost/effort

Mitigations:

1. Vetting of participants
2. Automated and manual vetting of contributions

Functional Specification

Create a Need

Discord message to a single special, restricted-access channel with JSON formatted message:

- Github URL
- Need description (e.g. can someone put a UI on top of this that makes sense of the gazillion flags and options?)
- Need URL – the URL of a longer description of what's desired in this fix
- Target OS name
- Target OS version
- List of [target thing name/thing version]

The message to the need channel triggers:

- Creation of an invitation-only discord channel dedicated to this need
- Copy/commit/push of the need message and need URL contents into the repo/main

Examples:
{ url: https://github.com/jvrodley/arabella, description: "Needs to work in Windows 10/python 3.6", target_os_name: "Windows", target_os_version: "10", targets: [{ target_name: "python", target_version: "3.6"}]}
{ url: https://github.com/jvrodley/arabella, description: "Needs a user interface and installer package to run locally", target_os_name: "MacOS", target_os_version: "Monterey", targets:[]}

Create a Fix

Pressing a button on the **Fixer** Need-list page causes:

- Invitation to join the discord-channel dedicated to this **Need**
- Private fork of need-github-url into Arabella account with only main branch
- Creation of develop and fixer-specific branches
- Invitation of fixer to collaborate with commit rights only to fixer-specific branch

Create a PR

In GitHub

- Fixer creates a PR-to-develop.
- Admin reviews PR and fixer deals with any issues
- Admin approves PR-to-develop
- PR-to-develop triggers automated callback to Arabella

In Arabella, accepted PR-to-develop notification triggers:

- Series of expandable, configurable, automated security checks against isolated, locally-cloned copy of develop branch

- Automated PR-to-main request into GitHub

In GitHub

- Admin reviews PR
- Admin approves PR
- PR-to-main triggers automated callback to Arabella

In Arabella, accepted PR-to-main notification triggers

- email notification to Investigator-watchers
- Discord message to “fixed” channel
- PR back to originating project on GitHub
- Recalculation of FRep
- Update of Arabella front page

Roles

Arabella Admin

Identified by username, password and hardware key (required)
Capabilities:

- Administer users

Investigator

Identified by username, password and usual MFA (hardware key or authy)
Capabilities:

- Watch a fix
- Endorse a fix

Fixer

Identified by username, password and usual MFA (hardware key or authy)
Capabilities:

- Create/update a fix
- Create a PR

Investigator Vetting (IVet)

We expect that investigator vetting is an entirely offline process handled internally by Bellingcat/Arabella. All that Arabella needs to know is that an investigator has, or has not, been vetted. IVet is a single screen, reachable only through hardware key. Invitation to investigator status is a manual process.

Fix Vetting

Fixes are vetted through a 2-step process using traditional Git pull request process. A project being worked on is forked to a PRIVATE repository and 3 branches are created - main, develop and feature. The Fixer may only commit to feature.

A PR down to the develop branch on the forked repo is accepted or rejected with comment by the Arabella team. An accepted PR to develop triggers an automated PR down to main.

A PR down to main goes through a set of configurable and expandable security scans and a final manual approval before being merged down to main.

An accepted PR down to main automatically sets the repo visibility to public and triggers a notification to the original requester.

Fixer Reputation (FRep)

As the system is designed to pull contributors not currently known to us into the system, FRep must be part of the system and allow contributors with a reputation of zero to make meaningful contributions. FRep is a combination of feedback provided in Fix Vetting (FVet) and Fix Acceptance (Facc). Vetters either accept or reject a fix, and if they accept they provide a numeric score for the quality and difficulty of the fix. That score feeds into Fixer Reputation. When an investigator accepts a fix, they provide a single quality score. Those feedbacks are raw data while how that data is reflected in FRep is an ongoing algorithmic exercise.

PR Back

The PR Back function automatically creates a pull-request back to the originating repo to give the original tool author and his users a chance to benefit from the fix. Feedback from the PR Back process is added to the FRep score - with “accepted without comment” being the highest score.

Fixer UX

- Login
    - Home page
    - List of current projects

Investigator UX

- Login
    - Home page
    - List of personal arabella projects
    - Create a project button
- Project status page
    - Claimed/not-claimed
    - Current status in-progress, waiting on PR to develop, waiting on PR to main, no-broken-works-for-me, done
    - Request re-open if closed-not-done
- Create project page
    - Project github URL
    - Description of the problem
    - Error output if available
    - Desired operating system/language version/browser version
    - Characterization of need – hair-on-fire, need-it-soon, nice-to-have

Demo Script

Create a Need by pasting a JSON message into a discord channel

- Show new need-specific Discord channel
- Show **Needer's** invite to new need-specific Discord channel
- Show new **Need** on Arabella's home page

Create a Fix by pushing a button on the Fixer home page

- Show **Fixer's** invite to need-specific Discord channel
- Show **Needer's** notification of fix-in-progress
- Show new Github project in Arabella account with main, develop and fix branches
- Show **Fixer's** invitation to collaborate on GitHub project

Clone the new Arabella project locally, make a change, commit the change to fix-branch, create a PR to develop.

Accept the PR-to-develop in GitHub

- Show one automated security check run against an isolated clone of the develop branch
- Show automated PR-to-main

Accept the PR-to-main in GitHub

- ** Show email notification to Investigator-watchers
- Show Discord message to need-specific channel
- Show PR back to originating project on GitHub
- ** Show update of Arabella front page

Schema

Entities

- User
- Need
- Fix
- Fixer
- Investigator
- Administrator
- GitHub project
- Arabella project

Branding: Bellingcat or Arabella

Though this question is mostly out-of-scope for the hack, whether Arabella is internal-to-Bellingcat or independent is important. The following factors, in our opinion, argue strongly for starting Arabella as a standalone entity.

- Arabella could fizzle or fail thus negatively affecting Bellingcat's reputation.
- Arabella has skill needs that are not core to Bellingcat (e.g. engagement building, CI/CD ..).
- Any integration with Bellingcat increases Bellingcat's attack surface
- Arabella could succeed as a service without materially advancing the mission of Bellingcat.
- Arabella can always be brought in-house.

Go To Market

1. Bellingcat employees as investigators, invite small tranche of beta testers from hackathon as fixers (2 months) - fix bugs
2. Small tranche of external investigators, more hackathon fixers (2 months) – improve useability
3. Large tranche of external investigators and invite all hack participants as fixers (2 months) – button up for public
4. Open beta (3 months) – tune for engagement
5. GA