Identifeye

ONLINE ACCOUNTS THAT BELONG TO THE SAME PERSON

About Identifeye

When running some online service, you have the **problem** of users attempting to hide their identity in order to avoid IP bans or log in to an account they shouldn't.

Our **solution** is to build a heuristics engine that can determine the real identities of users and whether they're doing anything malicious. It takes in data from the players such as their IP address and username and returns whether the user is doing anything suspicious.

That data can then be used by the server to **send alerts** to staff about the **suspicious behavior**.

Data Collection

Our **data was collected** in partnership with Penguin's Minigame Server, a game server that deals with malicious users at scale.

We collected data on **user accounts** which included users' account names, character names, IP addresses, countries, UUIDs, number of minutes spent on the server, and whether they were banned. However, since this is **sensitive data**, we wrote a script that hashes all the values using the bcrypt algorithm, and salting the data.

Challenges

The hardest part of this project was figuring out how to architect the **data analytics system** and how to make it efficient.

Also, basing our project off a **third party** made it more difficult to obtain our **data** and added a single point of failure if the third party decided to change their mind on giving away the data.

These challenges along with the time constraints and a **full schedule of classes** limited the project.

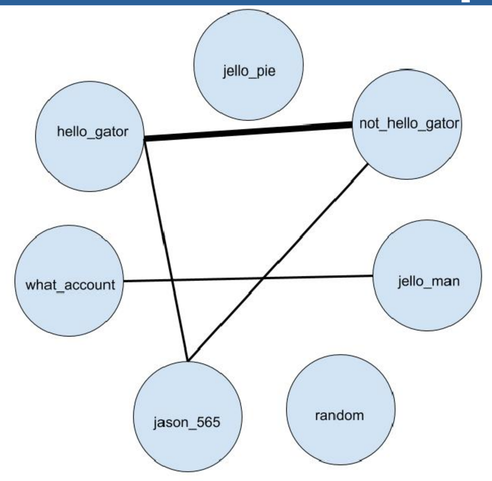
Technologies




Example Data

ID	Account Name	Character Name	IP	UUID	IP Geolocation	Is Banned	Activity Time
0	hello_gator	Bob	127.1.0.1	56897	US	1	105
1	what_account	Jason	265.2.0.2	63541	Canada	0	212
2	jello_pie	Jason	315.6.4.7	88541	Mexico	0	105
3	jason_565	Rambo	127.1.0.1	11115	US	0	2
4	not_hello_gator	Bob	127.1.0.1	56897	US	0	5
5	random	Ryan	666.3.2.1	78912	Mexico	0	88
6	jello_man	Simba	111.5.6.0	63541	Canada	0	12

Connections Graph



Connections Confidence

```
Account Name hello_gator
 -> Connected To: = not_hello_gator with a confidence of 59.2%
 -> Connected To: = jason_565 with a confidence of 28.6%


Account Name what_account
 -> Connected To: = jello_man with a confidence of 28.6%


Account Name jason_565
 -> Connected To: = not_hello_gator with a confidence of 28.6%
 -> Connected To: = hello_gator with a confidence of 28.6%


Account Name not_hello_gator
 -> Connected To: = jason_565 with a confidence of 28.6%
 -> Connected To: = hello_gator with a confidence of 59.2%


Account Name jello_man
 -> Connected To: = what_account with a confidence of 28.6%
```

Data Analysis

The program **collects a series of data points** from any player interaction such as when they log in. When comparing one property to another, we can draw an association between data points. The more **properties** that are **shared**, the more likely it is that two points are the **same person**.

To detect ban evaders, our database keeps track of a ban probability. When a new data point is added, all other data points are **cross-referenced**. For every **shared attribute**, the probability that the two accounts are the same person is increased.

Components

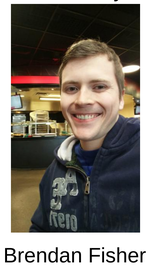
Anonymizer - script to take source data and remove any sensitive information by hashing it

Interfacing library - C# library to communicate with the analysis server

Analysis server - main Python application to run data analysis

Team Members

Zak Fahey



Brendan Fisher

Mitchell Haas

Dr. Raj Bhatnagar