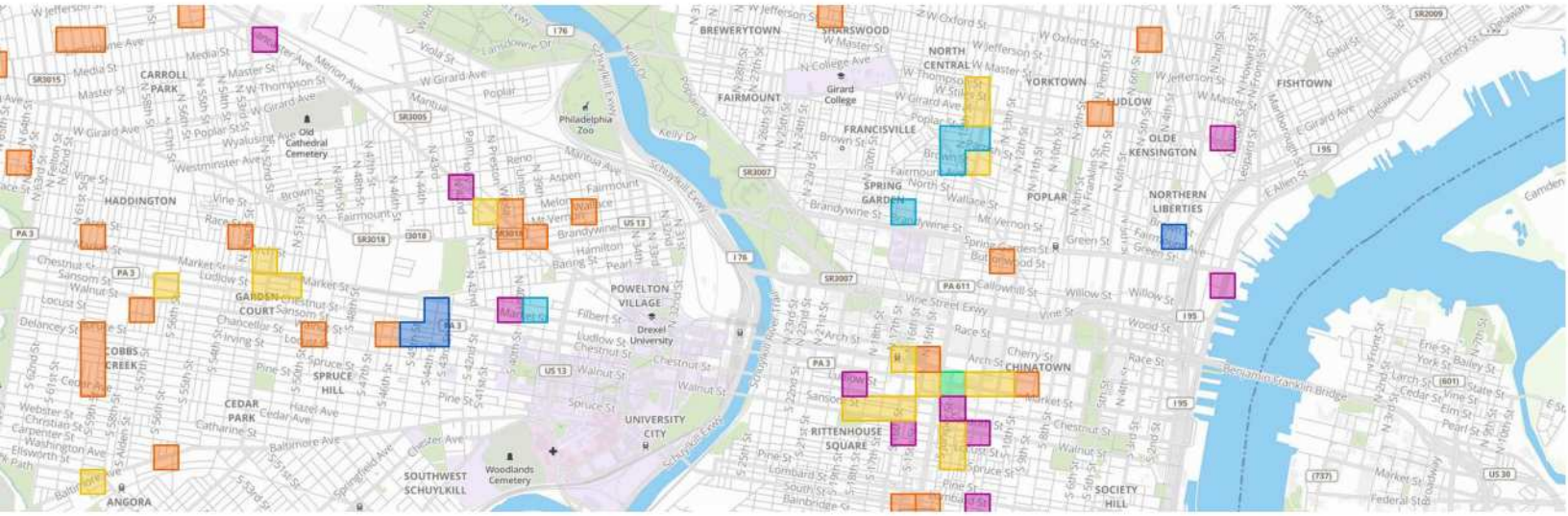
Answers to Questionnaire



HunchLab

Prepared by

azavea
340 N 12th St, Suite 402
Philadelphia, PA 19107
(215) 925-2600
http://www.azavea.com

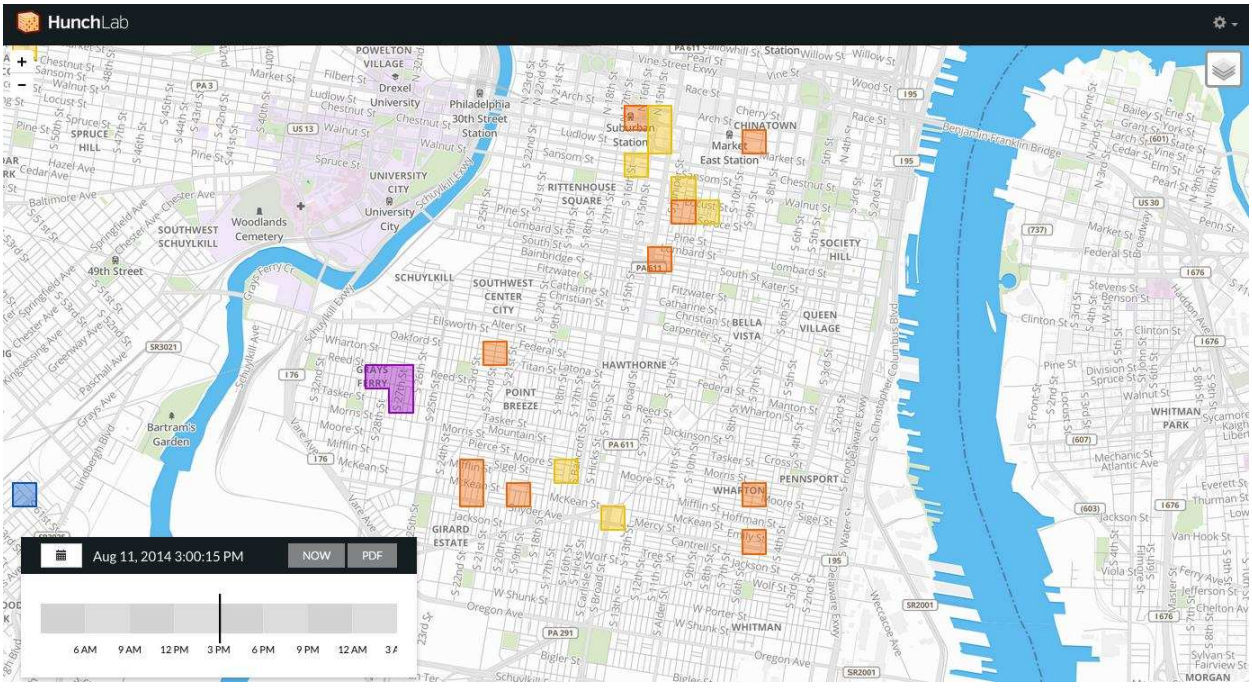
Hours of operation: 9:00 a.m. to 5:00 p.m. weekdays
Contact: Robert Cheetham, President and CEO
E-Mail: cheetham@azavea.com

Questions

1) What is the company's staffing model?

Azavea's core competencies lie with its staff and their skills. Our staff members include dedicated teams of software developers, user interface and user experience (UI/UX) designers, GIS analysts, data scientists, project managers, and business developers. These talented individuals provide services to a growing multinational client base that extends across North America, Western Europe, and Asia. Most of our 55 staff members work out of our offices in Philadelphia, PA. A few full-time employees work remotely from locations across the United States. The software development for HunchLab has occurred entirely by full-time employees of Azavea. Our vision is a world in which geospatial analysis is a broad foundation upon which government, private, and non-profit organizations operate.

2) Does the tool/system provide integration points into existing systems, e.g. REST endpoint. That is, can the product be integrated with existing NYPD systems?

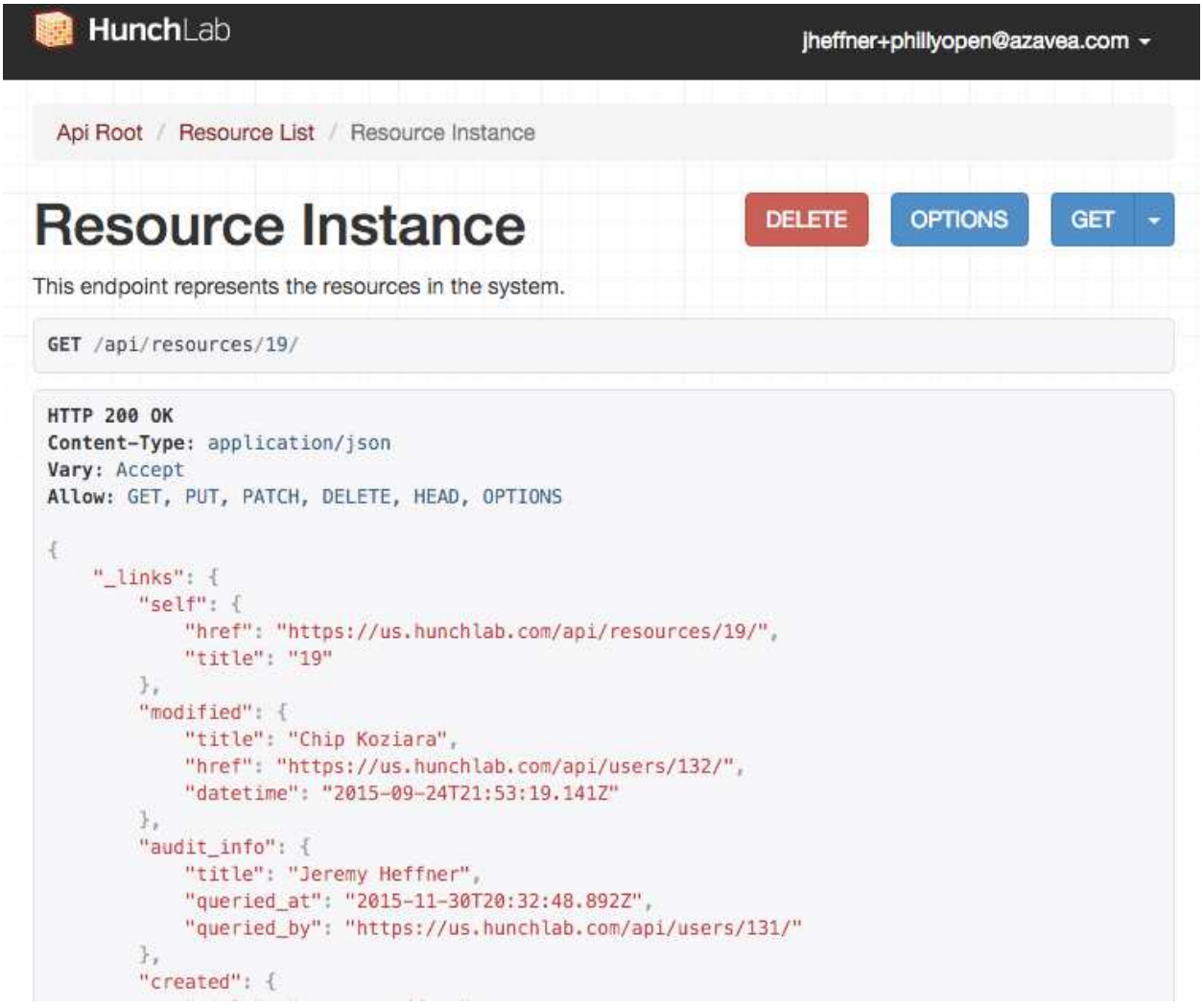


The HunchLab user interface is built as a static HTML5 application (without the use of any browser plugins such as Flash or Silverlight) that communicates with a RESTful application programming interface (API). This approach insures that by design any functionality provided in the HunchLab application is also accessible to 3rd parties through our API. Third party applications authenticate with our API by providing a simple API key in the same way that users of our interface authenticate (once logged in).

The same permissions and audit trails are enforced for all users of the application whether humans or other systems since they all use the same API to access HunchLab data. We have also implemented a browseable API to aid developers that are interacting with our application.

Our REST endpoints mostly consume and produce JSON formatted data unless another format was more appropriate. For instance, our REST endpoint that accepts new crime event data accepts CSV formatted data because CSVs are more commonly used to represent simple XY geographic data. Uploading new crime data to HunchLab is a simple POST of a CSV with a few key columns to our REST endpoint. Our predictive missions are displayed within our user interface as a GeoJSON layer, which is supported by many mapping products and can be transformed into other standard GIS formats.

We have assembled some documentation and example scripts for interacting with our API on our GitHub repository available at https://github.com/azavea/azavea-hunchlab-examples



We have provided support to integrate user authentication with existing systems via the SAML standard. For instance, Active Directory Federation Services (ADFS) can provide a SAML compliant authentication page. Users would simply authenticate to the ADFS page and then select to be redirected to HunchLab. User permissions to HunchLab can be managed through groups within Active Directory. This approach enables a police department to centrally roll out advanced authentication requirements such as those specified within the CJIS guidelines and have HunchLab seamlessly benefit. Because the SAML standard requires no direct communication between the HunchLab servers and the client's servers, no firewall modifications need to be made. Cryptographically signed authentication assertions are simply passed by the user's browser from one application to the other. This design means that your SAML authentication provider can even be entirely inaccessible outside of the NYPD network. In this case 2 factor authentication may not be necessary to meet the CJIS guidelines because the user's authentication occurs on the trusted NYPD network.

3) Does the product provide a measure of confidence/risk? An estimated volume?

The predictions made for each configured crime model are an expected count (volume) of crime for each square raster cell for a specific time period of a specific day and is based upon a Poisson distribution of event counts. This prediction is a point estimate designed not only to accurately reflect the risk in that specific area but also aggregate to an accurate assessment of risk across multiple locations/times. The predictions are not, however, a confidence interval or range of possible values.

Our predictions tend to be fractions of an event. For example, the expected count may be 0.05 burglaries. In almost all cases the probability that no event will occur exceeds the probability that an event will occur. We imagine that confidence intervals are most useful when asking the question "how sure is the system that something will happen there today?" The answer to that question is that it less likely than it is that something will happen. This is due to the very nature of modeling a stochastic process such as crime. What is more interesting is to say which of two locations is more likely to experience a crime and our point estimates do that well.

4) Can the size of prediction areas be customized?

HunchLab's analysis is conducted on a raster grid of configurable size. We typically use a raster resolution of approximately 150 meters, but have run analyses at resolutions of 50 meters. Higher density urban areas may benefit by smaller resolution predictions and our testing so far suggests that the predictive power of the model is maintained even with smaller resolutions. Smaller resolutions will amplify data quality issues, however. For instance, if your crime data is geocoded to street centerlines, a smaller analytic resolution will begin to carve out the area contained within blocks because no crimes

are geocoded to the actual locations of the events. Also, the operational implications of small units should be considered. Does telling an officer to go to a 25-foot area make sense?

After building predictions on the grid, predictions areas are turned into a vector representation and simplified such that touching areas with the same focus are combined into one mission area but currently maintain a grid inspired shape. We have considered running an analysis at a fine resolution (such as 25 or 50 meters) and then translating our predictions into mission areas based upon the outlines of the nearby parcels or street segments as a way of turning our analytic output into outputs more aligned with real-world features.

5) Can predictions be provided at custom time intervals (e.g., by tour, weekly etc.)?

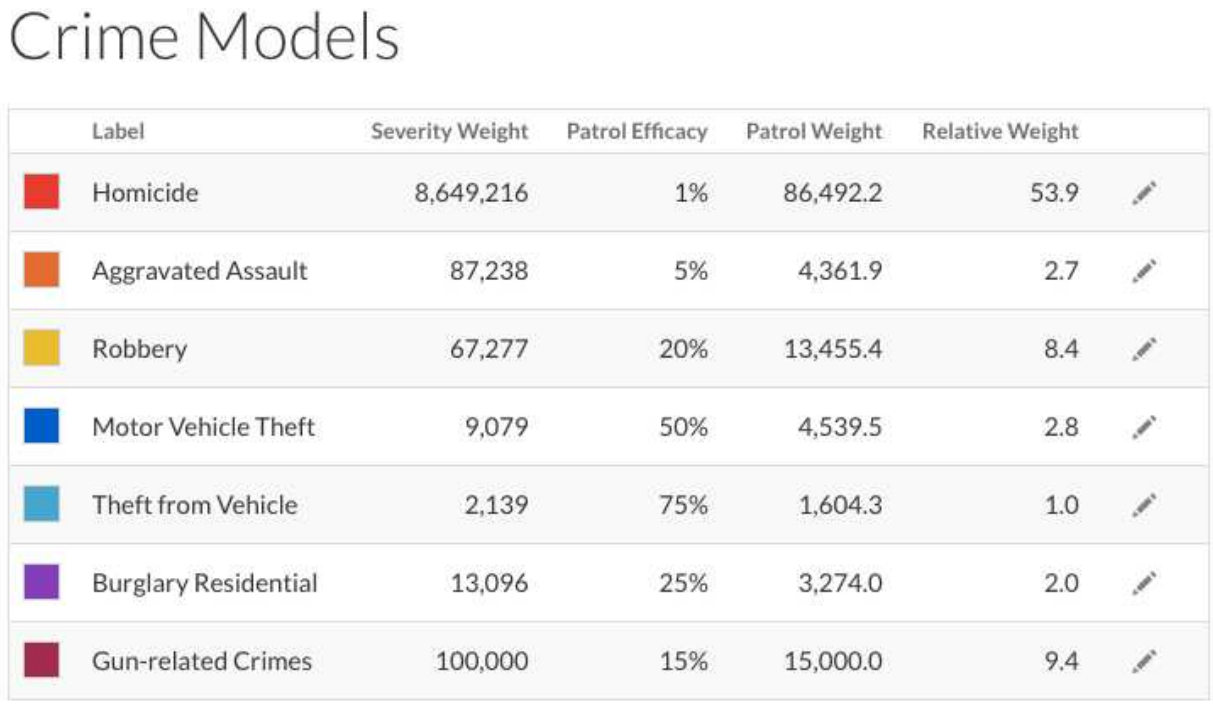
HunchLab predictions are maintained for the next 24 hours within the system to serve future use cases such as use in external scheduling systems. The system can make predictions down to an hour-based resolution (for example, 24 predictions across the 24 hours of the day). These predictions can then be combined into prediction periods for various tours which we call shifts within the system. Some of our clients map these prediction periods to entire tours whereas other clients want predictions to change more often throughout the shift. This is fully customizable with the only requirement being that our predictions start and end at the top of the hour.

Our current predictive model is focused on near-term predictions meaning that our model is optimized to predict the crime today and tomorrow not several weeks into the future. This design aligns with the use of our predictions for patrol purposes.

6) Can predictions for selected crimes be weighted more heavily based on Departmental and/or Commanding Officer priorities?

Yes, HunchLab configuration includes crime weighting information that informs the system how important different crime types are to prevent. For example, the department specifies how much more value there is in preventing a robbery versus preventing a burglary. There are a few common approaches we have seen for these weightings. One approach is to use published information about the cost-of-crime such as from the RAND Corporation. In this case, the weightings are expressed as a monetary value. The downside to this approach is that such numbers are only available for major crime types. An alternative approach is to use sentencing guidelines as a proxy for harm. This approach is put forward by Jerry Ratcliffe from Temple University. While sentencing guidelines are not perfect, they are a measure of the import that society places on various offenses. One benefit to this approach is that sentencing guidelines are available for all types of crimes. A second approach is that because the focus of predictive policing is to prevent crimes (and their associated arrests and incarcerations), by aligning

proactive work with the areas where the most potential incarceration would occur due to crime, the system is optimizing to reduce the incarceration rate.



Crime Models

Label	Severity Weight	Patrol Efficacy	Patrol Weight	Relative Weight
Homicide	8,649,216	1%	86,492.2	53.9
Aggravated Assault	87,238	5%	4,361.9	2.7
Robbery	67,277	20%	13,455.4	8.4
Motor Vehicle Theft	9,079	50%	4,539.5	2.8
Theft from Vehicle	2,139	75%	1,604.3	1.0
Burglary Residential	13,096	25%	3,274.0	2.0
Gun-related Crimes	100,000	15%	15,000.0	9.4

7) Can the predictions take into account verticality (or three-dimensional data)?

HunchLab does not directly use the third spatial dimension, but rather, it can be used to separate activity that occurs above, below, and at street level. We have previously used this approach to separate events that occur indoors (such as in elevators) from events that occur in public places (and hence are more appropriate for patrols to address). Essentially, when data is transformed for import into HunchLab, we create a compound crime classification label that incorporates the type of event and the nature of the event. For example, a burglary occurring at street level may be labeled as "burglary:street" whereas a burglary that occurs in a high rise may be labeled as "burglary:abovestreet". These separate crime classes can then be individually selected to form meaningful crime models. For example, we may create a model for street-level burglaries and include it in a basket of models driving standard patrol mission areas. Separately, we may create a basket of models for events that occur above street level and build a separate set of mission areas for a specialized unit that is conducting such work.

The same approach can also be used to create a compound classification system that separates domestic incidents from non-domestic incidents.

8) Does the product provide accuracy (hit-rate) reports after data updates are delivered?

Once historic data is loaded into the system and the desired crime models are configured, the system automatically builds statistical models. The model building process has two main phases. In the first phase, the last 90 days of crime data is held back to provide accuracy assessments of the models. Models are built without access to this recent data and then predictions are made across the 90-day period. After this assessment is complete, the best model configuration is re-built including the recent data to arrive at an up-to-date model.

The accuracy of individual models is assessed on the held-out 90 days of data. These accuracy metrics measure the several aspects of the predictions of each independent model. For example, if the system is configured to create three separate crime models for burglaries, robberies, and assaults, then predictions for each of the three crime models are made across the 90-day period and the accuracy of each model is measured separately. This approach lets us see how well the system is performing for different types of crimes and can help to identify crime types where additional covariate data may be most useful to assemble.

The predictions generated by the system are expected counts for each crime model in each raster cell location for a specifically configured 'shift'. For example, the system may predict 0.001 assaults, 0.02 robberies, and 0.03 burglaries. Keep in mind that we are dealing with a small area for a small period of time and so the expected levels of crime would also be small. These predictions are then compared with the actual counts across the 90 day period in several ways. One approach is to look at the numeric precision of the predictions. For this purpose, we calculate metrics such as the root mean squared error and the mean percent error (to detect predictions that are overall too high or too low). More importantly, however, we measure the prioritization power of the model to see whether the predictions can separate locations and times when crimes are more likely to occur from locations and time when they are less likely. For this purpose, we calculate metrics such as the percent of crimes captured in the top 1% of locations/times, the average ranking of the crimes that occurred, the normalized Gini coefficient, and the area under the receiver operating characteristics curve.

At the same time that the system creates predictions using its preferred machine learning algorithm, it also generates predictions using six baseline models designed to mimic output that an analyst could produce manually. These baseline models include hotspot style predictions using temporal windows ranging from the last week to the last year of activity. These serve as important points of comparison so that we can demonstrate that the system is providing value.

The hit rate (accuracy) of the predictive areas themselves is a function of the accuracy of the individual crime models as well as the prevalence of each crime type and the weightings associated with each crime model. We can run the system over time, generate mission areas, and then measure the accuracy of the missions directly, but to do so the missions cannot be used operationally. This limitation is

important because interventions in the missions are aimed at preventing crimes, which will distort the outcome.

9) What, if any, security protocols will be incorporated, (e.g., two-phase authentication)?

The HunchLab architecture and hosting environment provided by Amazon Web Services has many protocols focused on security. Attached to this document is an appendix that outlines the system architecture and security features of the system, such as encryption of data at rest and transit, firewalls, and system updates.

Given that authentication was directly mentioned, we will outline the exposed surface area of the HunchLab application to provide guidance on the selection of proper authentication.

There are three access points to the HunchLab system and infrastructure: (1) the AWS management console, (2) a remote access SSH bastion server, and (3) the application's API.

The AWS management console is protected by usernames and passwords given to individual Azavea staff members that are working on the system. These passwords conform to CJIS guidelines. The accounts are also protected by 2 factor authentication (TOTP). Interactions with the AWS console form an auditable log of activity. The remote access SSH bastion server provides Azavea staff with access to the servers running HunchLab via SSH and exposes a single port IP-limited to the Azavea office. Authentication to this bastion server requires cryptographic keys. All SSH access to the back-end servers powering HunchLab must flow through this single bastion server. To further reduce the surface area, the bastion server powers itself off when it is not being used. In order to power the bastion server on, an Azavea staff member would need to first access the AWS console.

The application's API is exposed to the public Internet on a single port and powered by an AWS managed Elastic Load Balancer. The API is written using the Django framework. Most interactions with the API are authenticated by a 40 character randomly generated API token. Users can obtain this token by logging into the application providing a username and password. Alternatively, the API token may be provided after a user authenticates with a configured SAML authentication provider (such as mentioned in our answer to Question 2, above). A SAML authentication provider could incorporate numerous forms of additional security such as being IP limited to a private network or requiring two-factor authentication.

We have not yet built direct integration of two-factor authentication into our authentication system because we have not had a client want to use it yet. In cases so far, clients either wanted to maintain simple usernames and passwords or wanted us to delegate authentication to their existing system. Overall, rolling out two factor authentication will be simplest if each user has only one two factor authentication device and so by delegating to such a system via SAML, we attain the security of two

factor authentication while not imposing additional overhead on users and IT administrators. If the NYPD desired for us to provide two-factor authentication directly, we are open to implementing it.

10) Is the application delivered in a secure cloud-based solution?

Yes, the application is delivered as a secure cloud-based solution. Users access the application through modern web browsers. We support the last two versions of Internet Explorer (10/11), Chrome, and Firefox. All communication with the application is over a secure TLS channel using TLS 1.1 or higher due to flaws in earlier versions of TLS. This requirement prevents access from some older browsers that do not support these versions of TLS.

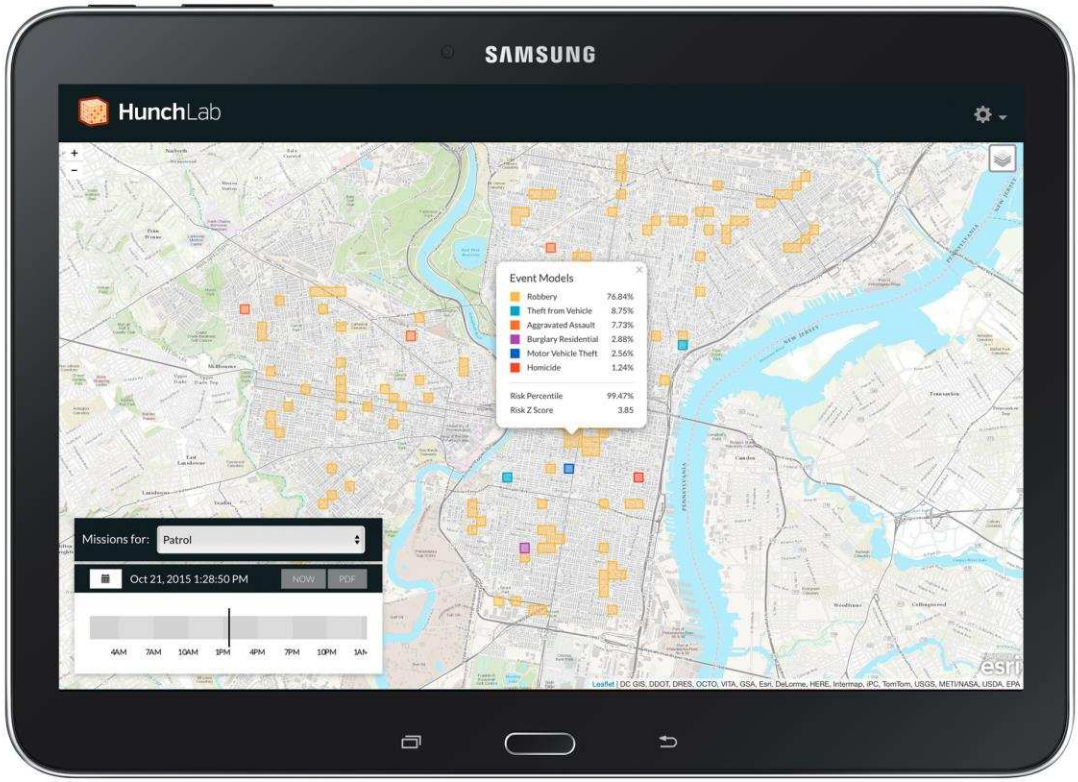
A test page for browsers and mobile devices is available at http://test.hunchlab.com

Azavea manages all IT concerns of the solution such as the deployment of new features, system updates, backups, and recovery from failures. The solution is hosted within Amazon Web Services infrastructure located within the United States to maintain data residency within US jurisdiction.

11) Are the predictions/system mobile compliant?

The web application is built using HTML5 without Silverlight or Flash. This approach enables mobile devices to access our user interface using mobile web browsers such as Internet Explorer, Chrome, and Safari. The mobile interface automatically detects the screen size of the device and adapts the display accordingly. We have also been piloting the use of the GPS feed from mobile devices (smartphones, tablets, and MDTs) to power a location-based display of information that we call Sidekick.

We have also run into departments where mobile connectivity is lacking or where the devices available in the field are antiquated. To support this use case, we can also generate simple PDF reports that display the mission locations. These PDF reports are then either printed out or disseminated by electronic means.




CONFIDENTIAL

Mission Area 3

Aggravated Assault | Robbery

Robbery RECENT EVENTS

- 1500 BLOCK W LINDLEY AV -- 8/1/14 1:51 AM

Mission Area 4

Aggravate... | Robbery

Robbery RECENT EVENTS

- N BROAD ST / W CHEW AV -- 8/1/14 3:42 PM

HunchLab District 35 Page 13 of 16

12) Is the output from the system "map-able" or map based?

The output of the system is expressly map-based. Within our application, the locations are displayed as a GeoJSON layer, which is a common GIS format for web-based maps and can be imported into other systems directly or transformed into other GIS formats.

13) What data is required as input into the system (historical crime data, weather, special events, proximity to other geographic features, demographics, etc.)?

The only required data is the historic crime data itself (along with updates at least once a day). All other data sets are optional but we encourage clients to leverage at least some basic geographic layers to provide richer context to the crime models. More information about the data sets that we can use as well as data sets that we can provide are available in the data guide appendix.

Azavea's belief is that the use of non-crime data sets as variables within an operational crime prediction system is important because variables based solely upon crime data become skewed as predictions are used operationally. As crimes are prevented in mission areas due to police response, the only variables identifying areas as high risk are skewed in other systems. By including other data sets, our system is more robust against this issue.

14) Does the system utilize a broad set of crime theories (e.g. risk terrain modeling and near-repeat modeling) in the generation of predictions?

From its earliest iteration, HunchLab's crime risk forecasting – or "predictive policing" – techniques have been based upon published academic research that has looked at an individual aspect of crime patterns. For example, Azavea software developer and data scientists worked with Professor Jerry Ratcliffe at Temple University to create a daily risk forecast in HunchLab for burglaries, shootings, and other crime based on his near-repeat pattern research. Police officers have understood for many decades that, for some crimes, the risk of being a repeat victim is quite high. In other words, if someone is a victim of a burglary, there is actually a significant chance that they will be a repeat victim in the weeks after the initial crime. But Ratcliffe and his colleagues discovered something even more interesting. Not only is there an elevated risk that someone will be a repeat victim, but that the risk of his or her neighbors becoming a victim is also higher for a few weeks after the initial crime.

In addition to the near-repeat pattern phenomenon, HunchLab includes concepts such as the Risk Terrain Modeling research being published by Rutgers University. Risk Terrain Modeling describes geographic location through correlated geographic features such as bars, schools, and transit stops. This

permits the forecasting of crime locations not because crimes occurred there yesterday, but because the social and environmental conditions are ripe for crimes to occur there in the near future.

HunchLab includes each crime theory by deriving individual sets of variables that represent the underlying concepts. For example, Risk Terrain Modeling may be represented by measuring the distance to the nearest bar and the density of bars in each raster cell. Near-repeat patterns may be represented by measuring the amount of time since the most recent crime occurred in each raster cell. These sets of variables are then passed into the modeling process, which determines the useful theories for a given crime type. The system also determines how the theories interact. For example, if the near-repeat pattern effect is stronger in areas with lots of historic crimes, the system can use that information to enhance the forecast. If assaults frequently happen on Friday evenings near bars, the system can similarly model that effect. HunchLab incorporates machine learning concepts to help the software "think" like a crime analyst by imitating years of experience drawn from a police department's own data.

The forecasting engine uses ensemble machine learning approaches that can incorporate the following crime patterns into a single prediction of criminal risk:

- Baseline crime levels
  - Similar to traditional hotspot maps
- Near repeat patterns
  - Event recency (contagion)
- Risk Terrain Modeling
  - Proximity and density of geographic features (points, lines, and polygons)
- Routine activity theory
  - Offender: proximity and concentration of known offenders
  - Guardianship: police presence (historic AVL / GPS data)
  - Targets: measures of exposure such as population, parcels, or automobiles
- Collective Efficacy
  - Socioeconomic indicators, neighborhood heterogeneity, etc.
- Temporal cycles
  - Seasonality, time of month, day of week, time of day, etc.
- Recurring temporal events
  - Holidays, sporting events, etc.
- Weather
  - Temperature, precipitation, etc.

15) Does the system use machine-learning, or some similar technology to learn which theories are the most applicable and then calibrate the model accordingly?

While available GIS tools have enabled law enforcement agencies to advance their understanding of crime through more effective geographic visualization for many years, these tools have traditionally required trained analysts, are used by a tiny subset of agency personnel, and are largely reactive in nature. There is a compelling need for new crime analysis tools that perform automated discovery of crime patterns and make that information available in a format that can be easily understood and acted upon at a variety of agency levels, including officers in the field. HunchLab provides these capabilities. It is a groundbreaking system that "learns" which crime theories matter for a given crime types and automatically calibrates the influence of these theories – both individually and as arbitrarily complex interactions – to identify and address a wide range of crime risks and other public safety issues.

In some ways, the model building process in HunchLab mimics the thought process of an experienced analyst. For instance, consider asking an analyst to decide where to place patrol resources for a given upcoming time period. She may start by looking at where crimes have occurred in concentration previously and delineate hotspots of activity. Based on her past experience, she may know that during this particular time period, schools dismiss their students, which increases petty crimes around the schools in the neighborhood. She builds up many such layers of knowledge and balances these various concerns to form a plan. After the time period concludes, she may go back and look at where activities occurred to see if she can determine additional insights into the crime patterns to include in future plans. HunchLab incorporates machine learning concepts to help the software "think" like a crime analyst by imitating years of experience drawn from a police department's own data. The concept of machine learning is to teach a computer to accomplish a particular task. In this case, we want to teach the computer to determine how likely a particular crime type is to occur at various locations for a given time period. We start this process in HunchLab by forming a set of training examples using the past several years of crime data. Each training example contains the theoretically derived variables we explained separately, as well as the outcome (how many crime events occurred). For an entire municipality this training set will often include many millions of example observations. We can then start building the model.

The primary model HunchLab currently uses is a stochastic gradient boosting machine (GBM) comprised of decision trees trained using the AdaBoost loss function. This model is built to forecast whether a crime event will occur or not in a given space-time raster cell (a binary outcome). The general way this model works is as follows: Begin by selecting a random portion of the training examples. Build a decision tree that separates examples of where crimes occurred from ones that didn't based upon the variables. For instance, the first decision within the tree might be interpreted as: "if no event happened in the last year in this location, it is very unlikely for a crime to occur today". The decision tree then splits the examples into two sets: (1) where a crime occurred during the past year and (2) where no crimes occurred during the past year. Within each set, the process repeats. For example, the next decision for the set of locations with crimes in the past year might be interpreted as: "if an event happened in the

last week, it is more likely for one to occur today". This set of examples would again be split based upon this decision rule. This process continues to build out a decision tree that describes 'why' crimes occur where they do. The decision tree is then used to make predictions of how likely crimes are for each observation in the entire training data set. This completes one training iteration within the boosting machine.

The modeling process then begins again. We start by selecting another random portion of the training examples. This random sampling process is why the model is stochastic. In this next iteration, we build another decision tree (in the same manner as above). This time, however, we build the tree to predict the errors from applying the first decision tree model to this new sample of observations. In this way we are attempting to correct our mistakes. This concept is called boosting. We then use these two trees to make predictions across the entire data set. As we conduct this process, we can keep track of how many training iterations within the machine have made incorrect predictions for each training example. We increase the importance (via weights) of observations that we continue to get wrong and decrease the importance of observations that we continue to get correct. This process is called adaptive boosting (AdaBoost). When we build the next decision tree, we tell it to focus on the observations that we continue to get wrong via these weights. Training iterations continue several hundred times. The resulting model represents tens of thousands of decision rules of 'why' crimes occur where they do. We conduct this entire process several times, each time holding back a portion of the example data. We can then use each of these models to make predictions for this held-out set of data to see how accurate the model is when we apply different quantities of training iterations from the model. For instance, if the models have 100 training iterations, we may find that the most accurate predictions come from only using the first 53 iterations. This process is called cross-validation and prevents our models from over-fitting the training data. Finally, we begin the entire process again using the whole data set to build a model with the correct number of training iterations. In this example, we would use 53 iterations.

As you can see, this modeling process mimics some activities that an analyst would go through in making decisions of where to focus resources. The predictions from this model are whether one or more crimes will occur or not. We then need to translate these probabilities into expected counts. We do this by calibrating our predictions using a generalized additive model that assumes a Poisson distribution. This regression model both translates the outputs of our model to expectations and calibrates the predictions. For example, the above model might slightly over-predict crimes on Tuesdays. This calibration step would lower the predictions for Tuesdays to center them on the training data. The process of using one model's outputs as another model's inputs is called model stacking. These models are then saved and used to generate predictions.

The predictions are calibrated count expectations for each raster cell for a given period of time. You may picture predicted counts to be numbers such as 0, 1, 2, or 3. In actuality, the predictions are real numbers that are often fractions such as 0.000001, 0.02142, or 0.12482. This represents the fact that the nature of crime is such that no software solution can say that a crime is going to happen at this specific corner at this precise time. For a small raster cell and time period, it is almost always more likely that no crime will occur. What is important is that we can use these predictions to measure the relative

risk of events between locations, time periods, and different crime types, so that we focus on the most likely types of events at the most likely locations and times.

16) Does the system provide the ability to modify parameters based on local/expert knowledge or circumstances?

Several parts of the system can be modified based upon local knowledge or circumstances. The department determines the specific types of crimes that are to be modeled. Each crime model is comprised of one or more crime classifications, which is customizable at a granular level. This feature enables a department to use local knowledge to exclude crime events that may not apply to patrol activities such as domestic incidents. Local knowledge would also inform the types of data that may make sense to feed into the system. For example, if you have anecdotal evidence that robberies increase in the days following a release of new Apple products, it may warrant assembling temporal information about the release schedules over the last few years to use in modeling.

The importance of preventing various crime types and the ability to prevent each type of crime via proactive patrol activities are informed by local knowledge in conjunction with established research by academia. It is important to realize that this is different from the question of what comprises the bulk of your crime events. The system will automatically know that you have more burglaries than homicides. What the system needs a department to inform it is how much more important is it to prevent one or the other, which is a rather moral question.

17) Provide professional references, three (3) preferred, who can describe your experience in predictive policing. It is preferred, but not required, from police departments or other law enforcement agencies.

Greensboro Police Department, Greensboro, North Carolina

The Greensboro Police Department is performed a simple research experiment with HunchLab's predictive missions which we believe you may be interested in hearing more about. It included an analysis of not only the change in crime activity but also an analysis of how officer behaviors changed due to the mission areas.

Contact Information:
Eleazer "Lee" Hunt, Information Systems Manager
Greensboro Police Department
(336) 373-2145
eleazer.hunt@greensboro-nc.gov

Philadelphia Police Department, Philadelphia, Pennsylvania

Temple University is working with the Philadelphia Police Department to test the impact of different police strategies on violence and property crime. The project is using HunchLab to boost the data available on the success of predictive policing. By randomly assigning 20 police districts to one of four experimental conditions, the study will attempt to answer whether different police responses to crime predictions estimated by a predictive policing software program can effectively reduce crime.

Contact Information:
Kevin Thomas, Director of Research and Analysis
Philadelphia Police Department
(267) 251-0974
kevin.thomas@phila.gov

Temple University NIJ Project, Philadelphia, Pennsylvania

Temple University is working for the National Institute of Justice to develop and release a free and open source crime forecasting tool that combines long term predictions of crime based upon Census information with the short-term effects of near repeat patterns. Azavea is providing software development services and statistical feedback to Temple for this project. While in some regards this is directly competitive with our commercial interests, we believe that advancing the field of crime analysis overall is more important than losing some revenue to free and open tools.

Contact Information:
Jerry Ratcliffe, Chair of the Department of Criminal Justice
Temple University
Philadelphia Police Department
(215) 204-7702
jhr@temple.edu

18) Complete and sign the Federal Bureau of Investigation Criminal Justice Information Services ("CJIS") Security Addendum, attached hereto as CJIS Security Addendum. An authorized representative shall sign the Certification to the Addendum on behalf of the Contractor. Contractor shall instruct its employees that individual employees may also be required to sign the Certification, at the discretion of the NYPD.

Attached.

19) The Contractor as an organization and personnel that have direct knowledge of this procurement will be required to submit confidentiality agreements as requested by NYPD. The selected Contractor shall provide the following information regarding all employees and subcontractor employees that will have direct responsibility regarding software design, software programming, implementation, project management and/or warranty and maintenance service responsibilities:

1. Name of Individual
2. Sex
3. Country of Birth
4. Business Title
5. Place of Business (address)
6. Telephone #
7. Email address
8. Date of Birth
9. Social Security # (or equivalent) – last four digits.
10. Home Address including Country.

See attached. Please note that due to the personal nature of the information requested, we are sending this document in hardcopy only and not including it in our e-mail response.

Appendix A: Predictive Missions

HunchLab provides customized, automated mission creation and prediction maps based on resources and crime types. Once the system is configured, no manual steps are necessary for Missions to be created. Missions are selected by the combined, weighted risk of all configured crime models. Color represents the dominant risk for a mission, as shown in Figure 1, below. In Figure 2, each mission area displays a risk profile of the components that went into selecting this location.

A police department's priorities are reflected in the crime models configured within the HunchLab administrative user interface. Severity weights enable the department to tell HunchLab how important it is to prevent each type of crime. In Figure 3, the cost of crime numbers from the RAND Corporation are utilized to align policing priorities to the societal impact of crime. Patrol efficacy values enable the department to specify how much impact they believe patrols will make on each type of crime. The result is that missions show up where the most important, preventable crimes are likely to occur.

The unit of analysis within HunchLab is configurable but typically recommends an area of approximately 250 meters square and a duration of a few hours within a shift. Maps of these targeted missions can be printed or distributed as PDF reports at the beginning of each shift and taken on patrol. If connectivity is available in the field, the reports can also be viewed on mobile and tablet devices through a web-based interface. An example is provided in the Sample Reports section of this proposal. Mission areas will reflect the latest crime information available to HunchLab at the time of their creation.

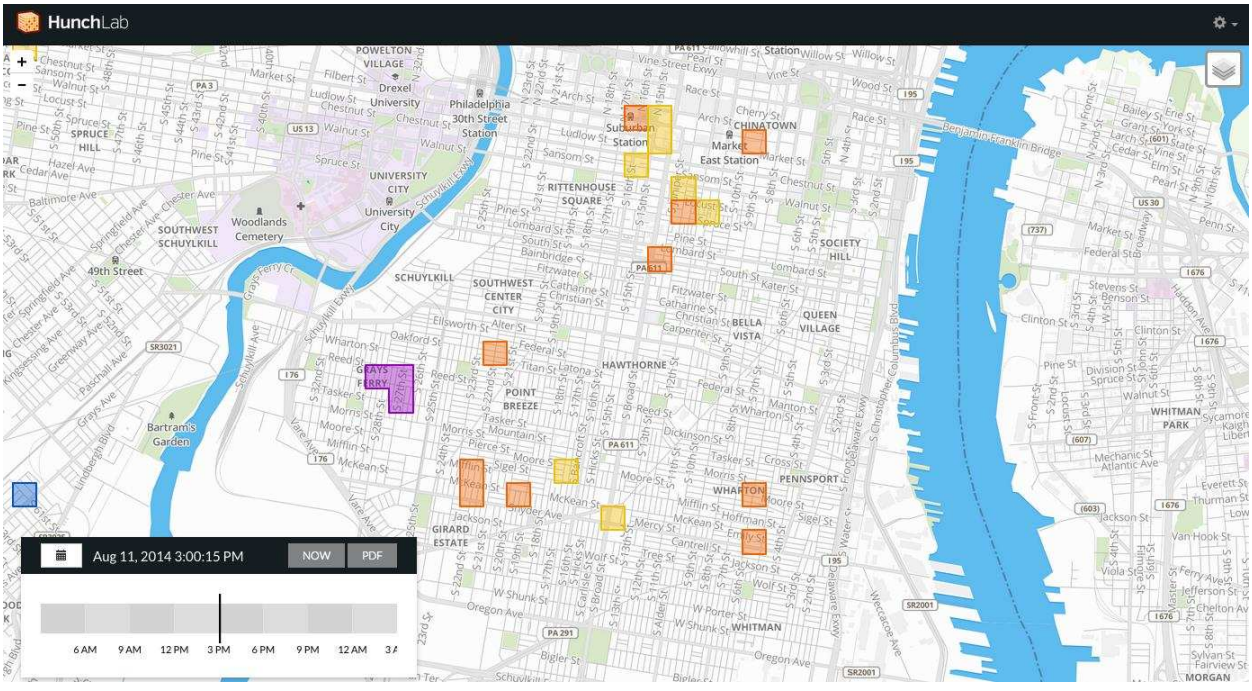


Figure 1: *HunchLab displays high-risk mission areas.*

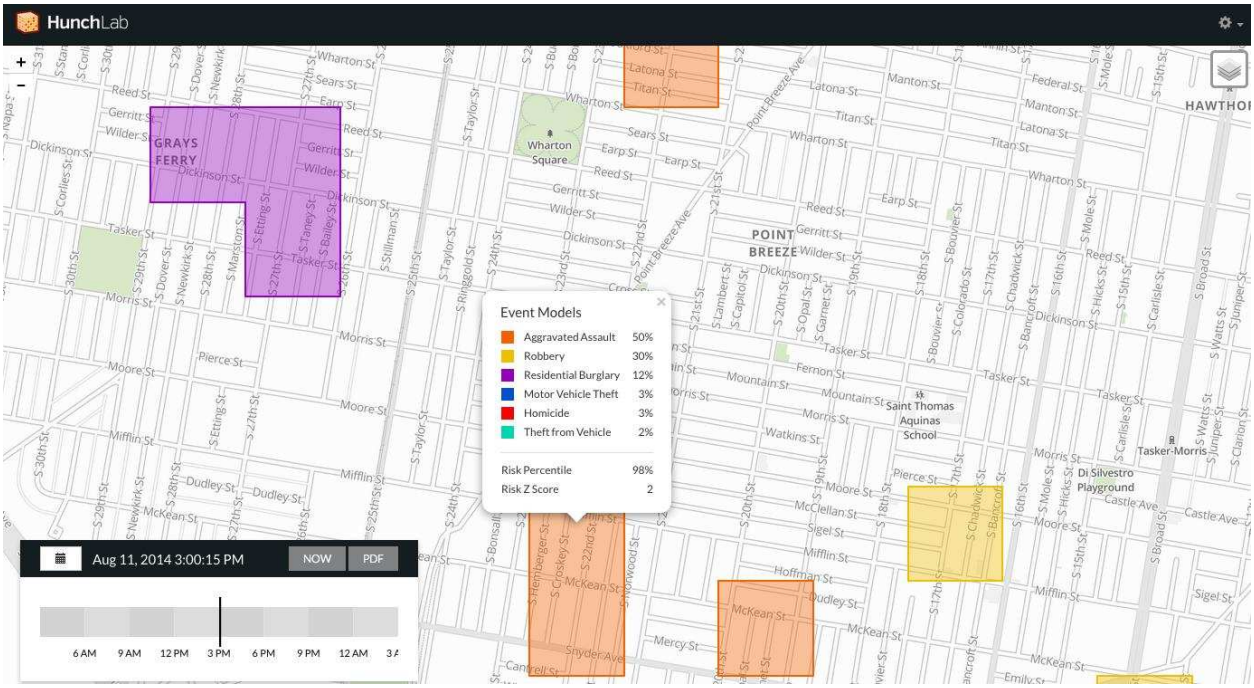


Figure 2: *Looking at the individual components of a mission area.*

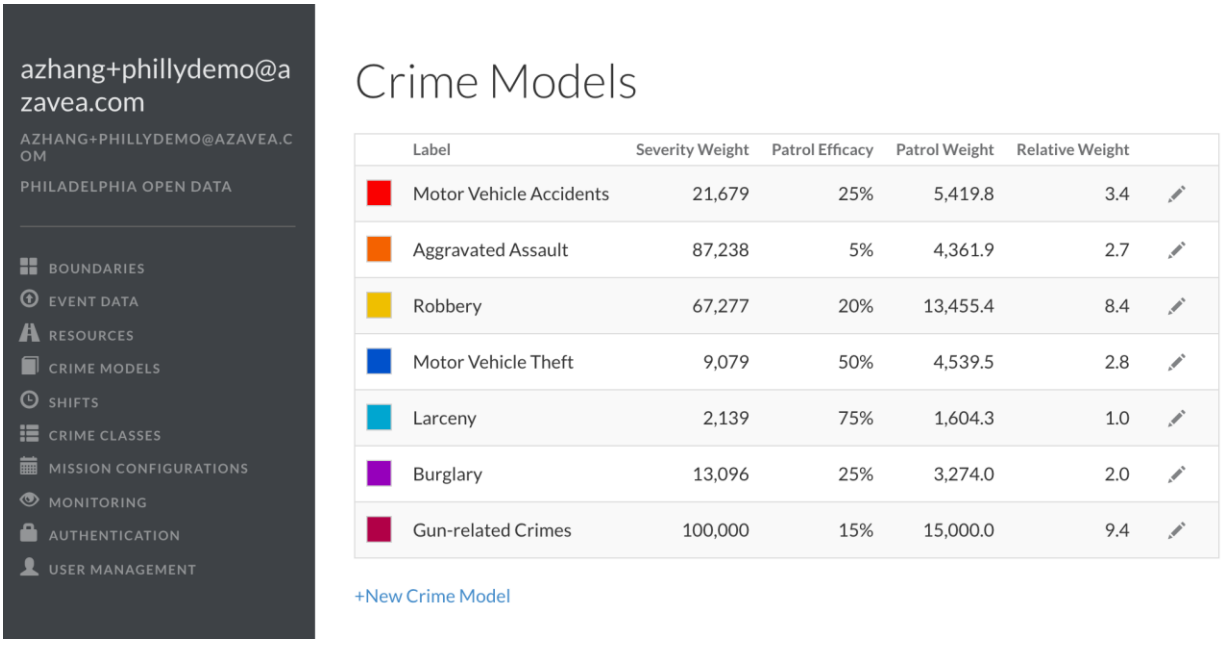


Figure 3: *The configuration of crime models is displayed.*

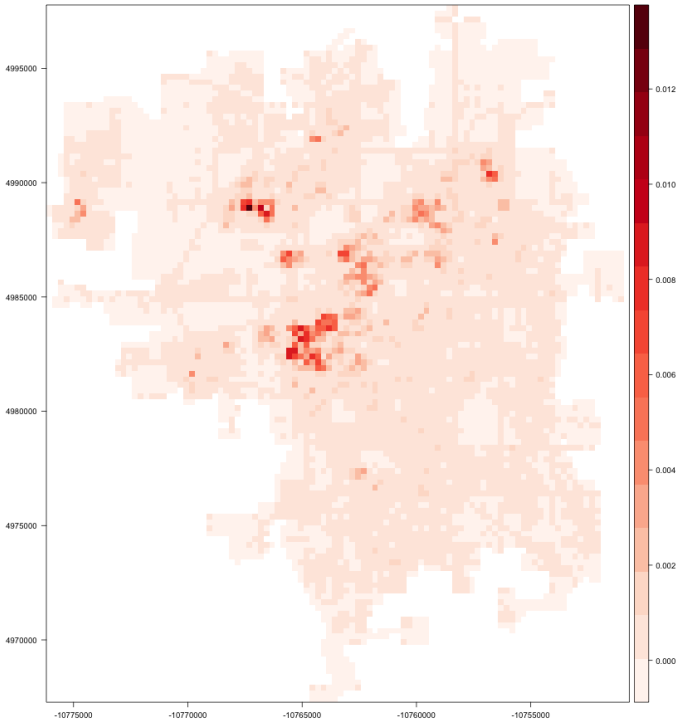


Figure 4: *HunchLab forecasts the expected count for each configured crime type within a shift for each small location within the jurisdiction. This figure shows an example of the map of one such set of forecasts.*

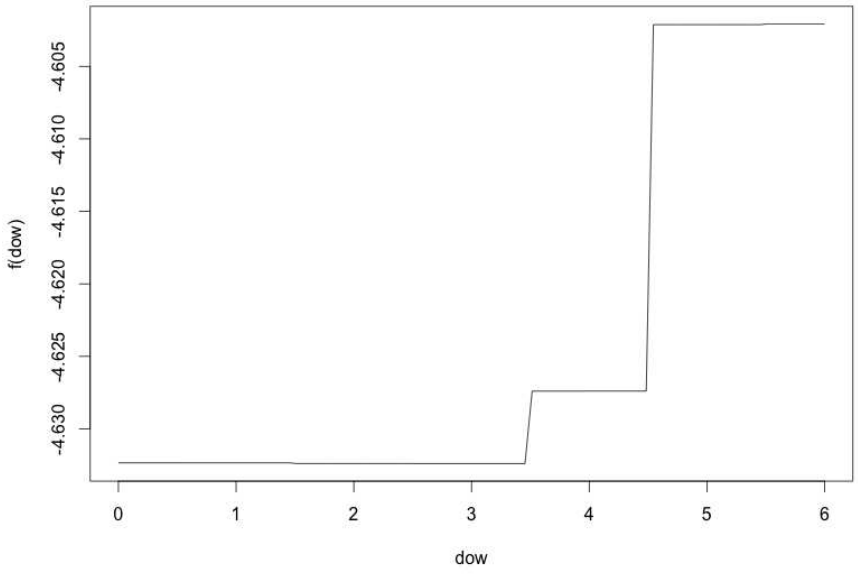


Figure 5: *The forecasting models can be examined to visualize what the system has determined effects the risk levels. In this case, the system learned how Friday, Saturday, and Sunday (4, 5, 6) have higher levels of assaults in Philadelphia.*

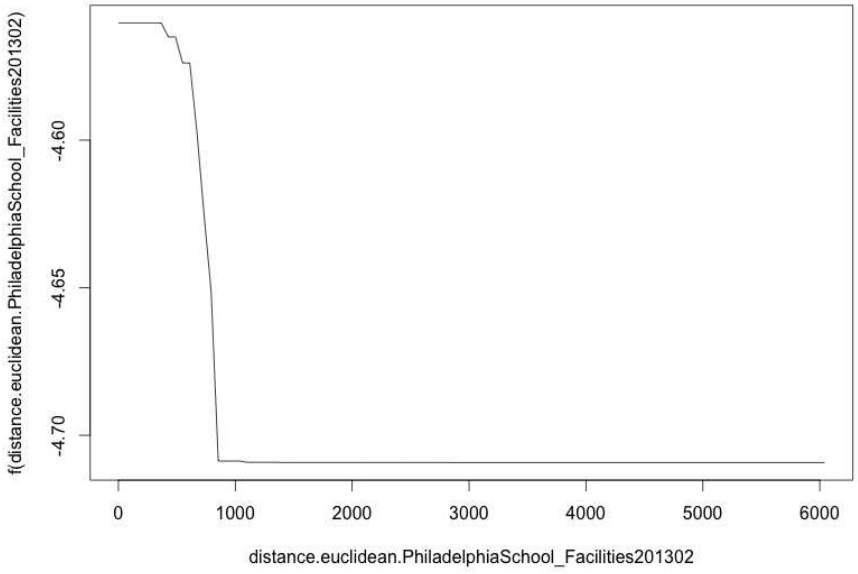


Figure 6: *In this example, the proximity to schools is shown to increase assault risks extending to about 900 meters.*

Figure 7: *A visualization of the different data sources contributing to forecasts for Motor Vehicle Theft. HunchLab had access to the same information in each city within this example, but uses them in each individual model to varying degrees based upon the local data.*

As a web-based application, HunchLab mission areas can be viewed through contemporary web browsers on devices ranging from desktop computers and laptops to tablets and MDTs. For viewing the data in the field, the simplest dissemination technique is a printable PDF report outlining mission areas for the shift. See Figure 8 for more details.

For departments with mobile broadband and GPS-enabled MDTs, smartphones, or tablets HunchLab also provides a location-based service called Sidekick. Sidekick provides officers with crime predictions about their current location, notifies them of mission objectives, and supports measurement of the dosage of field tactics to address crime problems. Access to specific system functionality is restricted based upon security roles. See figure 9 & 10 for more details. Please note that Sidekick is in revisions and will be ready in approximately 4 months.

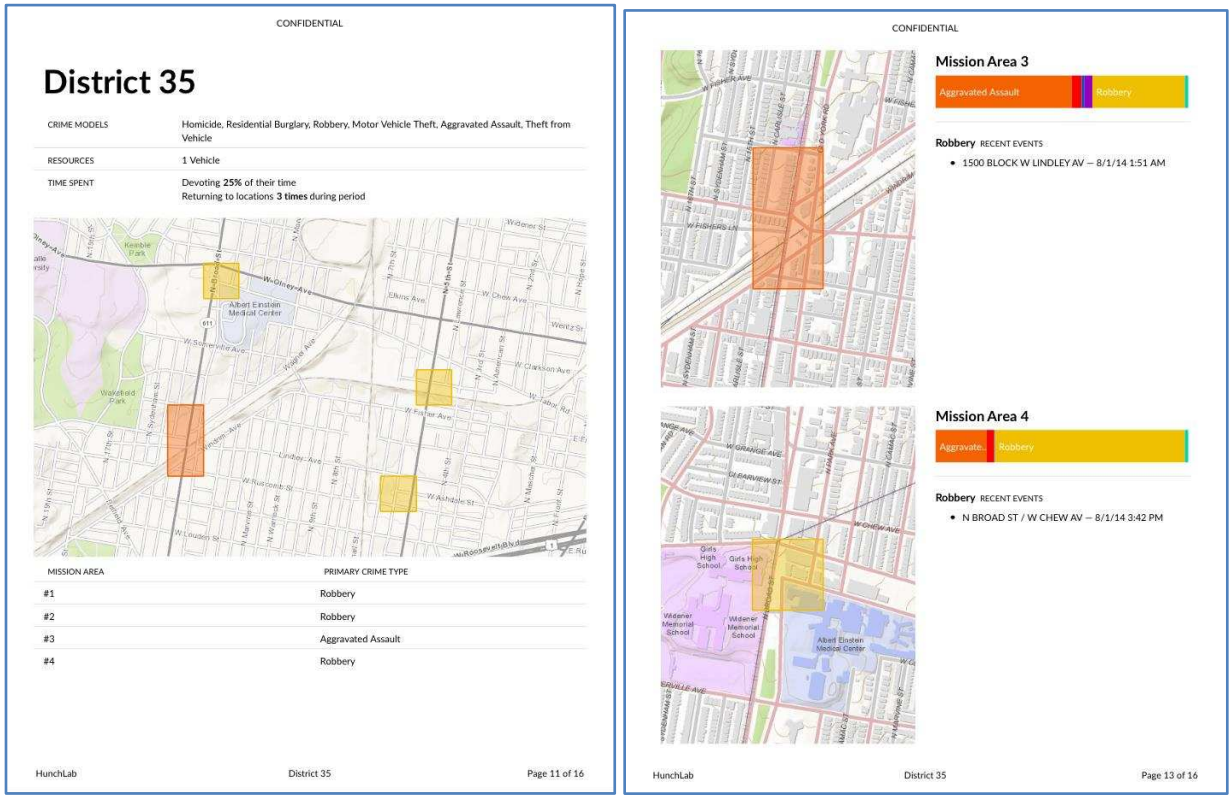


Figure 8: *Sample PDF reports that can be emailed or printed out and given to the officers right before patrol if Internet connectivity is not reliable in the field.*

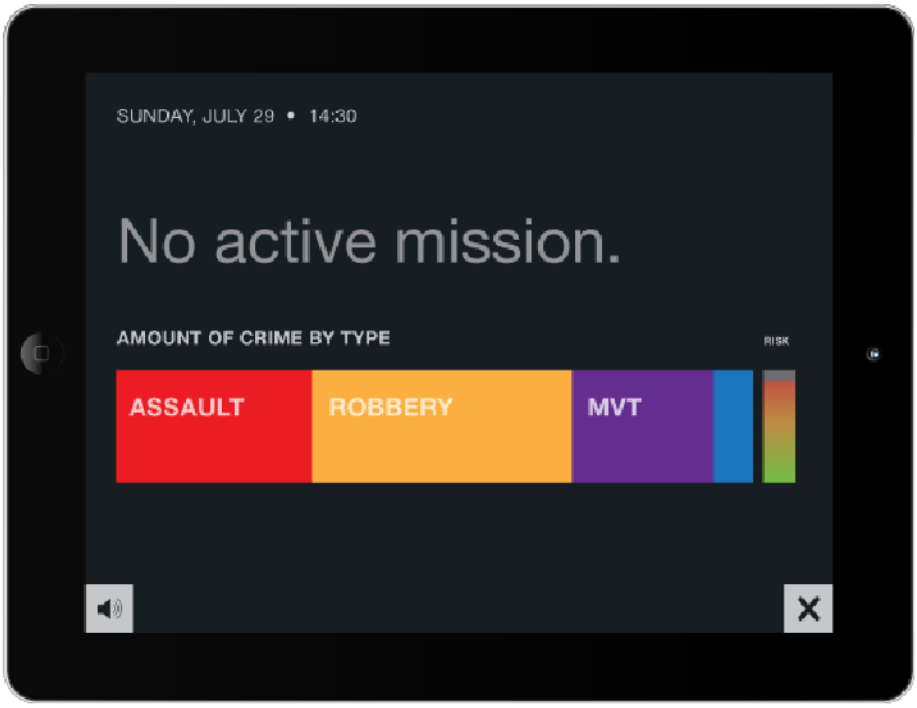


Figure 9: *The sidekick interface is location aware. GPS tracking displays relative risk for the current location on the tablet or MDT*

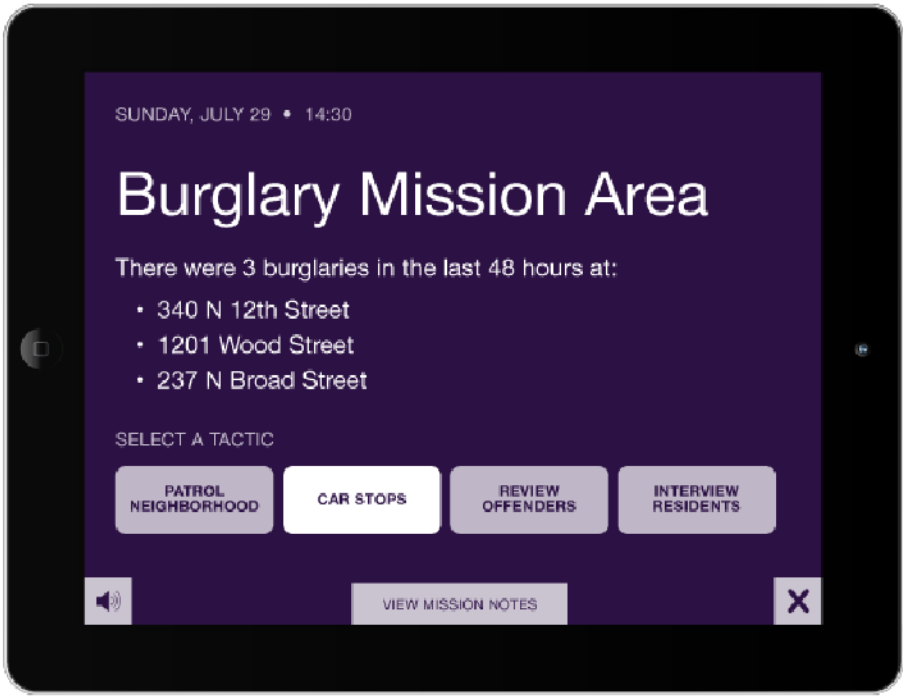


Figure 9: *When entering a predefined missions area, the user is notified of relevant information such as tactics configured for this crime type.*

The crime modeling process in HunchLab automatically calculates accuracy measures for each type of crime modeled by the system. These measures are available to clients and are most useful before the system is used operationally. Additionally, Azavea has modeled crime data from many jurisdictions that reflect diverse settlement patterns, densities, and crime problems to insure that the system can adapt as necessary. The system's accuracy varies based upon data quantity (more is better), quality (cleaner is better), and the nature of the crime type.

While it seems appealing to be able to include a glowing report on the effectiveness of predictive policing tools for deterring crime, these results are very difficult to prove. Software, after all, does not prevent crime on its own, but it can have a substantial preemptive impact, depending on how it is used by the agency that deploys it. For example, the same tool might be deployed at two different law enforcement agencies in the same geographic region. If the culture of the first agency is very data driven, they will likely adopt the tool to great success. If the command structure at the second agency does not value the tool or use it extensively, it is likely to have limited impact.

That said, the theoretical concepts for predictive policing that are collectively operationalized within HunchLab have been studied and documented as successful by academics for many years. Azavea has provided links to some research papers that we and others in the field have written on these techniques, including Risk Terrain Modeling, near repeat forecasting, and other methodologies. Individually, these methodologies have demonstrated their effectiveness in a number of police around the world.

Near repeat pattern analysis

Haberman, CP and Ratcliffe, JH (2012) The predictive policing challenges of near repeat armed street robberies, Policing: A Journal of Policy and Practice.
http://jratcliffe.net/papers/Haberman_Ratcliffe_2012_Predictive%20policing%20challenges%20of%20armed%20street%20robberies.pdf

Ratcliffe, JH and Rengert, GF (2008) Near repeat patterns in Philadelphia shootings, Security Journal. Volume 21, issue 1-2: 58-76.
http://jratcliffe.net/papers/Ratcliffe_Rengert%20(2008)%20Near%20repeat%20patterns%20in%20Philadelphia%20shootings.pdf

Risk Terrain Modeling

Heffner, J. (2013). Statistics of the RTMDx Utility. In J. Caplan, L. Kennedy, and E. Piza, *Risk Terrain Modeling Diagnostics Utility User Manual (Version 1.0).* Newark, NJ: Rutgers Center on Public Security.
http://www.rutgerscps.org/software/index.html

Additional resources (publications, software, manuals):

http://www.rutgerscps.org

Seasonality

Wilpen Gorr , Andreas Olligschlaeger , Yvonne Thompson (2003) Short-term Forecasting of Crime, International Journal of Forecasting 19
http://forprin.dev.zoe.co.nz/files/pdf/Gorr_Olligschalger_and_Thompson,_Short-term.pdf

Appendix B: System Requirements

Hardware Requirements

The HunchLab application is hosted as a multi-tenant Software-as-a-Service (SaaS) application within the AWS infrastructure. Azavea will manage the hosting infrastructure, security updates, and 2nd tier support. This cloud-based approach enables HunchLab to leverage significant amounts of computing power in an elastic manner, a critical requirement for providing the advanced statistical algorithms the system employs. Replicating a similar environment on-premise would entail a substantial outlay of capital to provide servers that are utilized only in bursts. To provide a secure application, we have consulted the FBI's CJIS guidelines to apply as many guidelines as possible in designing our system architecture, including risk mitigation techniques such as encrypting data connections and optional 2-factor authentication.

The application requires network connectivity from the user to the HunchLab service. Bandwidth requirements are modest, as most application assets are cached locally in the browser.

HunchLab 2.0 is hosted within the Amazon Web Services (AWS) infrastructure. AWS provides best-of-breed security and flexibility for building robust and secure SaaS applications.

Software-Related Information (Including Support and Upgrades)

HunchLab's yearly subscription includes application hosting, updates (fixes and new functionality within the place-based module of HunchLab 2.0), 2nd tier support, and ongoing training resources. This pricing model allows unlimited users and devices to access the application. All support services are coordinated and provided from Azavea's Philadelphia office and will include incident-based and troubleshooting support services by experienced Azavea staff through e-mail or phone during business hours, Monday to Friday, 9am – 5pm, EST (exclusive of designated US federal holidays). Additional support options outside of business hours are available for discussion as needed.

Azavea develops HunchLab through an agile Scrum methodology whereby work is planned in 2-week increments. This structure enables us to quickly develop iterative improvements to the application. New functionality and any necessary operating system updates or patches are deployed on a schedule designed to minimize downtime. For instance, most software updates result in about 0 – 15 minutes of downtime. System updates require no work from the client as Azavea staff manages the deployment process. The application is hosted as a multi-tenant application, so an update by Azavea for the US hosting environment will update all clients hosted within that environment simultaneously.

Clients are expected to continue to maintain modern, up-to-date web browsers on the devices that will be accessing the system. More information about our browser support policy is provided below.

Database Requirements

HunchLab does not require a client to have any particular database available. HunchLab does require event data (crimes or calls for service records) to already be geocoded with basic attribute data such as the date and time of the event (or time range), event classification, unique identifier, etc. More information about the data interface requirements is in section 4.

Interface Requirements

In a production environment, HunchLab will mirror authoritative data repositories, such as CAD and RMS systems. The manner in which this data is transferred to HunchLab varies from client to client. A typical process will consist of transferring records to HunchLab as an extension of existing crime mapping and analysis Extract, Transform, Load (ETL) processes. For example, this might be an extract process used to get data out of an RMS for crime mapping purposes.

Alternatively, Azavea can configure an upload process that draws data from an Open Database Connectivity (ODBC) connection to a read-only database view to fetch data that has changed since the last import was conducted. Most agencies schedule this import process on a daily or hourly basis, but HunchLab can also be configured to import changes on a more frequent basis such as every few minutes. Ideally data is provided for a 5-year historic period to allow robust predictive modeling.

Event data (crimes, calls for service, etc.) are transferred to HunchLab in a simple CSV format via a Representational State Transfer (or RESTful) HTTP Application Programming Interface (API) endpoint. A RESTful API is a framework that allows one to push and pull data to and from a system in a simple and secure manner. Clients can directly push data into this API endpoint or Azavea can support its use. CSV uploads contain column headers and basic attribute values such as the location and time of an event. Formatted CSV files can also be uploaded directly via the HunchLab administrative UI.

Alternatively, Azavea can fetch data from other RESTful APIs such as those provided by ArcGIS Server. If the endpoint is available via the Internet (with proper authentication), then no on premise utility needs to be configured as HunchLab can directly fetch updates. If the endpoint is behind a firewall, then the extraction process would be setup within the client environment and push updates to the HunchLab server.

Desktop Requirements

Staff access HunchLab through a web-browser. As an advanced web-application, HunchLab supports the following major desktop web-browsers and contemporary operating systems:

- Chrome (last two versions)
    - Windows XP or newer
    - Mac OS
    - Linux
- Firefox (last two rapid release versions and supported extended release versions)
    - Windows XP or newer

    - Mac OS
    - Linux
- Internet Explorer (last two versions; IE 10 and 11 as of September 2014)
    - Windows 7 or newer

Note: Window Vista and older Windows operating systems do not support secure versions of the TLS protocol (1.2+) within Internet Explorer. To support these older versions of Windows, we recommend the use of Chrome or Firefox (both free for installation) on these machines since they do support these newer versions of TLS.

We also support the following major mobile browsers:

- Safari
    - iOS 7 or newer
- Chrome (current version)
    - Android

In all cases, the default HunchLab configuration requires operating systems and browsers that support Transport Layer Security version 1.2. This requirement is to prevent known attacks against SSL traffic that impact TLS v1.0 and older protocols. Our supported browsers provide the correct version of TLS either automatically or with minor configuration changes (such as checking a box within the settings panel). If an agency is unable to support TLS 1.2 connections, it may necessitate the creation of a separate access point to the HunchLab system, which can be discussed on a case-by-case basis.

We realize that, at times, mobile data terminals (MDTs) are unable to be updated regularly. For 2015, we are supporting Internet Explorer 9 for the pages that MDT users would access daily (Map and Sidekick pages) within HunchLab.

For the Sidekick interface of HunchLab, we need access to the GPS location of the device. This information is accessed through the HTML5 geolocation API that is supported by all browsers listed above. Access to the GPS location on a specific device depends on the configuration of the device itself. For instance, nearly all tablets and smartphones with GPS chips provide access to location via this HTML5 API. Mobile data terminals may or may not have access to this feed depending on their specific configuration.

Appendix C: Data Guide

Introduction

HunchLab is a predictive policing solution that helps police departments to use their resources more effectively by leveraging advanced forecasts of crime. HunchLab's forecasting methodology fuses many crime theories and data sets into one picture of risk. The system automatically determines how to incorporate concepts such as recent crime events, temporal cycles such as day of week and season, the weather, and geographic locations such as bars and schools to produce a single forecast. The system uses these crime patterns when appropriate without requiring a police department to have a statistician on staff. This approach not only generates robust forecasts of crime but also provides insights into the dynamics of crime patterns.

The forecasting engine uses ensemble machine learning approaches that can incorporate the following crime patterns into a single prediction of criminal risk:

- Baseline crime levels
  - Similar to traditional hotspot maps
- Near repeat patterns
  - Event recency (contagion)
- Risk Terrain Modeling
  - Proximity and density of geographic features (points, lines, and polygons)
- Routine activity theory
  - Offender: proximity and concentration of known offenders
  - Guardianship: police presence (historic AVL / GPS data)
  - Targets: measures of exposure such as population, parcels, or automobiles
- Collective Efficacy
  - Socioeconomic indicators, neighborhood heterogeneity, etc.
- Temporal cycles
  - Seasonality, time of month, day of week, time of day, etc.
- Recurring temporal events
  - Holidays, sporting events, etc.
- Weather
  - Temperature, precipitation, etc.

Types of Information

Event Data

To forecast a space-time event such as a crime, HunchLab requires several years of historic data for the event to build both the outcome variable to be forecasted and several input covariates. At a minimum, HunchLab requires 5 years of crime (event) data for any event being modeled. This quantity of data is necessary to (1) "warm-up" variables that reach into the past up to 1 year, (2) include 3 years of examples to properly model seasonal patterns, and (3) hold back recent data to test accuracy.

This is the only required data set. Reasonably accurate models of crime can be generated with simply this data, but such models do not reveal insights into crime dynamics beyond crime events leading to more crime events.

Event data should be provided for at least the entire area for which forecasts will be used. If data can be provided for a buffer around this region, this can also be included. A buffer of up to 1000m can be useful within the modeling process. A reason to include additional data from nearby areas is that it may increase the overall data volume increasing predictive power. For instance, a small jurisdiction may not incur many violent crimes, but by including violent crimes from nearby jurisdictions more information is presented to the modeling process. Keep in mind that other data sets used in modeling must also be available for the buffered area.

Geographic Data

Geographic layers provide environmental context to the locations at which crimes occur. These datasets change slowly over long periods of time. While HunchLab's analysis is based on a raster format, geographic layers can be provided as points, lines, or polygon layers. HunchLab then builds variables based upon the distance to and concentration of these features. A given geographic layer may be split into multiple layers for the purposes of building covariates. For instance, given a street network for a city, each street segment may be of a different type – highways, highway onramps, residential streets, footpaths, etc. The distance to any street network feature may be a useful feature overall, but building variables for the distance to the nearest highway onramp or footpath may be useful as distinct variables. HunchLab can automatically split geographic layers on 'type' attributes to support this concept.

Implementation staff can easily take static extracts of geographic layers in ShapeFile format for inclusion in HunchLab. This approach requires no integration and therefore incurs no integration fees. Alternatively, a client may desire HunchLab to directly ingest GIS layers from a source such as an ArcGIS Server instance or other web API in GeoJSON format. In such cases, each system from which HunchLab pulls is considered one data connection and the relevant integration fee applies. Ingesting multiple GIS layers from a system does not incur additional fees.

While most geographic data provided by clients is in vector format, HunchLab can also leverage raster layers as variables. For instance, a city may have land cover data in raster format. Such data sets are transformed into a set of covariates and are resampled at the resolution of the HunchLab analysis.

Temporal Data

Temporal data sets provide information about the state of the entire jurisdiction and are considered "global" across the jurisdiction. For instance, a temporal data set may represent when the public school system is in session or the current air temperature. These data sets are provided in CSV format with the relevant time period, variable name, and a numeric value. School being in session may be represented as binary values with a value of 1 when in session and 0 when not in session. The air temperature may be represented as a numeric value in degrees Fahrenheit. Alternatively, the severity of activity between

two feuding gangs may be represented as integers: 0 for no activity, 1 for mild activity, 2 for severe activity.

It is important to realize that any temporal data used in forecasts must be available both historically for several years and for at least 48 hours into the future. The need for future values of the variable necessitates the use of variables that can be manually uploaded far in advance (such as the school schedule) or automation of updates (such as for weather). Temporal data sets that are uploaded into HunchLab manually do not incur integration fees. If instead, HunchLab was configured to automatically pull temporal data from a custom source, then a data connection fee would apply.

Other Variables

HunchLab also leverages variables that are not based upon specific data sets but are, instead, calculated. For instance, the day of the week and day of the month are simply calculated from the date. The moon phase, sunrise and sunset time, and season are other examples of variables calculated in a similar manner.

HunchLab Provided Data

HunchLab has processes in place to automatically manage the inclusion of common data sources if desired by the client. It should be noted that the use of these data sets is not required. For instance, a client may not desire any socioeconomic variables to be used in the forecasts even if academic research suggests it is useful.

Natural Terrain

Elevation data can be automatically loaded into HunchLab. This data set is transformed into several variables that describe the nature of the physical terrain such as the slope and aspect. This data is useful in identifying natural geographic structures that impact settlement patterns.

US Census

The US Census Bureau's American Community Survey provides up-to-date information about the US population based upon a sampled survey of residents. The data is available at the Census blockgroup level. HunchLab can automate the transformation of this data into relevant variables. For instance, this data set can provide measures of the collective efficacy and social cohesion of a neighborhood based upon socioeconomic indicators such as income and the prevalence of renters. The data set also includes information about potential targets of crimes such population density, automobile ownership, and home values.

Weather

Weather data provides a rich source of information about the conditions in a jurisdiction. For instance, seasonal patterns are often found in violent crimes, but these patterns may be more due to the conditions outside (warm temperatures) than the time of year itself. HunchLab can maintain historic

weather data and upcoming forecasts automatically for inclusion in models. Variables include such items as the air temperature, humidity, perceived temperature, and precipitation.

Open Street Map

Open Street Map (OSM) is an online, collaborative project to create an editable map of the world. The OSM database includes detailed information about street networks and major points of interest such as schools, libraries, and transportation hubs. HunchLab can use this data if such layers are not readily available from the client.

Thinking About Data Sets

In addition to the above data sets, clients can provide geographic and temporal data for inclusion in HunchLab's models. While more information is often better in building predictive models, a few well-chosen data sets can go a long way to building an accurate and insightful predictive model of crime. We encourage clients to think about this process in an iterative manner as additional data sets can be added over time.

When evaluating a potential data set for inclusion in HunchLab, there are a few key questions to ask:

- Is the data already available? If not, what will be the cost to generate and maintain the data?
    - For instance, a geographic layer that changes infrequently may cost little to maintain while one that changes more often may be burdensome.
- How strongly connected to crime is the data?
    - For instance, if the crime within a jurisdiction drastically changes based upon changes in the student population at a local university, then data sets related to the university are likely quite important.
- Are there synergies between this and other data sets? In other words, does 1 + 1 = 3?
    - The locations of public schools may be useful by itself. The school schedule may also be useful by itself. By providing both school locations and the school schedule, the system can fully identify when and where school may be having an impact. Such related data sets may warrant evaluation as a group.
- Does one set of data represent many ideas?
    - For instance, a city's parcel database may include zoning or land use information that provides information about residential developments, hotels, fast food locations and more.

Ideas for Data Sets

Here are some ideas of data sets that may be useful to include within HunchLab:

- Where people congregate
  - Restaurants, fast food, bars, liquor licenses, nightclubs, places of worship, tourist attractions, movie theaters, exotic clubs
- Where people live
  - University dorms, fraternities, public housing, apartment complexes
- How people move around
  - Bus stops, bus stations, train stations, recreational paths, highway onramps
- Venues for particular types of crimes
  - Pawn shops, retail stores, malls, convenience stores, motels/hotels, ATMs, banks, parking lots, bike parking
- Government buildings
  - Police and fire stations, libraries, post offices
- Problem places
  - Abandoned buildings, vacant lots, foreclosed houses

Additional ideas may be gleaned from the literature reviews of relevant factors for each crime type available for download from the Rutgers University website at http://rutgerscps.weebly.com/publications.html

Appendix D: Application Architecture

Introduction

HunchLab is a web-based server application provided under a software-as-a-service (SaaS) model. While the SaaS model of software deployment abstracts architectural decisions behind a simple client-facing web application, we realize that transparency is necessary within the law enforcement community.

The HunchLab application is hosted as a multi-tenant SaaS application within the AWS infrastructure. The application leverages a broad array of open source projects including operating systems, application frameworks, and statistical packages. Further, the application leverages AWS-specific technologies to provide scalability, redundancy, and security. Finally, the application is architected into discrete tiers allowing the logical separation of components.

Open Source Technologies

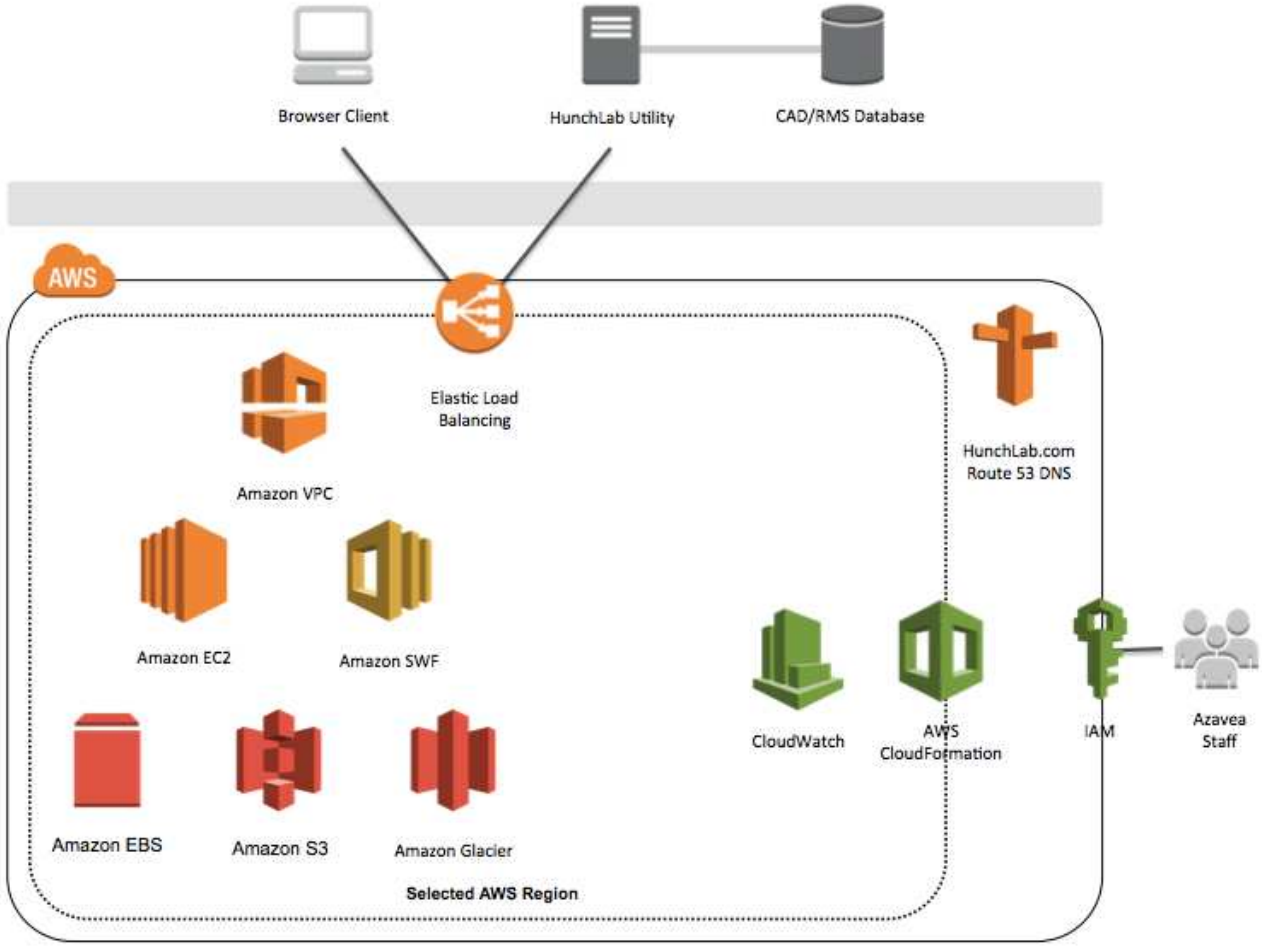
The application consists of a client-side, standards-based web GUI application implemented in JavaScript using the Angular JS framework. This GUI application speaks to a set of RESTful APIs implemented in the Django web application framework with data persistence provided by an AWS-managed PostgreSQL database with geographic queries supported by the PostGIS extension. Additionally, the system uses Azavea's open source GeoTrellis framework for high performance geographic processing and the R framework for state-of-the-art machine learning algorithms.

Amazon Web Services Technologies

The HunchLab SaaS application was designed to take advantage of the breadth of AWS services to provide a secure and scalable application. The application uses the following AWS technologies:

- Route 53
    - DNS for the hunchlab.com domain is managed through the distributed and redundant Route 53 service.
- Virtual Private Cloud (VPC)
    - VPC allows the isolation of application components on individual subnets, enforces network-level traffic rules, and provides both inbound and outbound firewalls.
- Elastic Load Balancing (ELB)
    - The SSL encrypted web traffic for the application is terminated by elastic load balancers which provide secure management of the signed HunchLab SSL certificate and performance under increased application loads.
- Elastic Compute Cloud (EC2)
    - Servers provided by EC2 are used for the web application, database, and machine learning tiers of the application. Many AWS services utilize EC2.
- Elastic Block Storage (EBS)
    - EBS volumes back the root partitions of EC2 instances and are used to store client-specific data.
- Relational Data Store (RDS)

    - RDS provides a managed relationship PostgreSQL database to HunchLab. The RDS instance is configured for real-time replication and automatic failover between availability zones.
- Simple Storage Service (S3)
    - Additional application artifacts are stored in the S3 service using the server-side encryption option. These artifacts include data sets undergoing processing, analytic models and results, and backup files.
- Glacier
    - Long-term backup archives are hosted in the Glacier service.
- ElastiCache
    - An in-memory application cache is provided by the Redis functionality of ElastiCache.
- Simple Workflow Service (SWF)
    - Machine learning and prediction processes are managed via the SWF service allowing distribution of tasks among a cluster of compute instances that scales to meet client needs.
- CloudWatch
    - CloudWatch metrics and alarms are used to scale application resources to meet demand and to notify Azavea staff of failures.
- CloudFormation
    - CloudFormation is used to securely manage and update the application stack with discrete application components isolated from one another and designed to automatically scale to meet user load.
- Identity and Access Management (IAM)
    - IAM is used to provide individual credentials to Azavea staff tasked with supporting the application. IAM security policies require the use of 2 factor authentication tokens when interacting with the AWS infrastructure. Additionally, IAM security roles are used within the application stack to provide credentials to application components.
- CloudTrail
    - CloudTrail provides audit logs of interactions with AWS management commands. These logs are stored securely within S3.

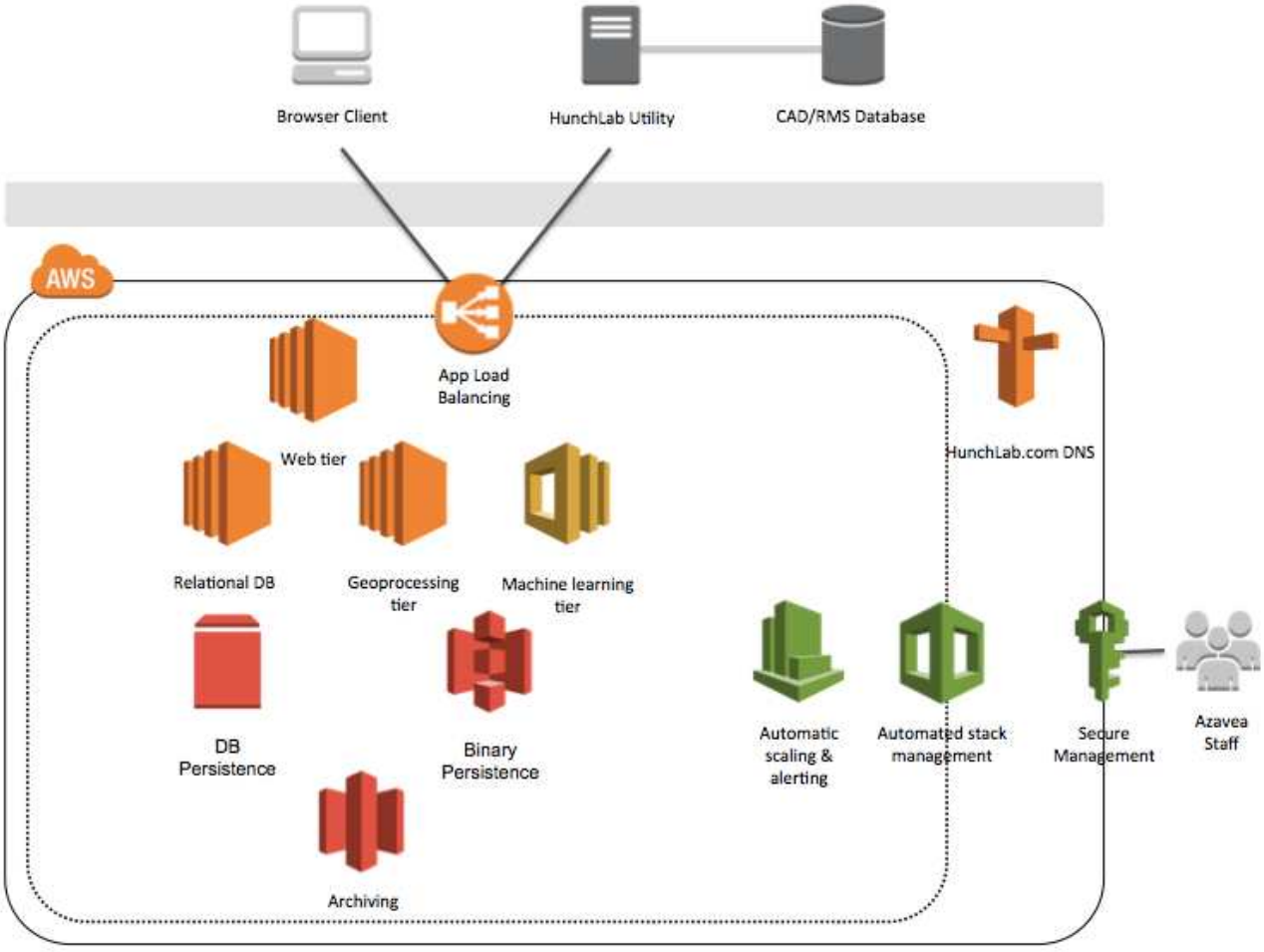


Application Components

The HunchLab SaaS application is designed as a set of loosely coupled components that work together to service the user. The main components of the application include:

- Client-side
  - Browser-based application
    - JavaScript application that provides the graphical user interface
  - HunchLab data upload (varying formats based upon client needs)
    - Data integration utility for crime data
- Server-side
  - Web application tier
    - Serves static files and provides RESTful APIs consumed by the browser application and integration utility
  - Geographic processing tier
    - Conducts geographic processing to support requests from the web tier
  - Machine learning tier
    - Batch processing for creating statistical models and generating crime predictions

- Persistence tier
  - Relationship persistence for the web tier and file persistence for objects shared among tiers



Appendix E: Application Security

Azavea has a long history of handling sensitive law enforcement data sets. The new version of HunchLab is delivered as a secure cloud-based subscription service using Amazon Web Services (AWS). As we designed this new version, we focused on incorporating security best practices into our development process. While most deployments of HunchLab contain local department data sets that do not technically require compliance with the FBI's Criminal Justice Information Systems (CJIS) guidelines, we are using the CJIS requirements and recommendations to guide our decision-making process and system architecture. Here are some of the security features and policies available within the new HunchLab.

Overview

AWS data centers maintain strict physical access controls including 24x7, trained security. Authorized staff must pass two-factor authentication a minimum of two times to access data center floors. AWS staff members pass criminal background checks prior to employment.

Further, the AWS platform regularly passes third-party evaluations. AWS has achieved ISO 27001 certification and has been validated as a Level 1 service provider under the Payment Card Industry (PCI) Data Security Standard (DSS). AWS annually publishes SOC 1, 2 and 3 audits. AWS is also a FedRAMP Compliant Cloud Service Provider (CSP) with validation at the Moderate level. This validation covers both the regular US regions and the GovCloud region. AWS has been successfully evaluated at the FISMA Moderate level for US federal government systems as well as DIACAP Level 2 for US DoD systems.

AWS Compliance information: http://aws.amazon.com/compliance/

Availability

The AWS platform provides robust services to maintain application availability even in the face of infrastructure failure. Within each AWS region, multiple availability zones allow an application to remain available even with the complete failure of an individual data center. Power and network connectivity systems are designed for redundancy with onsite backup power generation.

The HunchLab application is designed to use multiple availability zones within a region to provide availability even in the face of the loss of a complete zone. For instance, if a client's HunchLab application is hosted within the US East region, client data is replicated between multiple availability zones within the region. Availability zones are independent data centers within the region. The application is designed to survive the complete failure of an availability zone (a complete data center) without manual intervention by Azavea.

Data Residency

Distinct AWS geographic regions allow applications to be deployed to different parts of the world. This allows HunchLab clients to select a region based upon applicable privacy laws. Data placed within a region is not automatically replicated to other regions by AWS.

Clients can select from residency in:

- North America
  - US East (Northern Virginia)
  - GovCloud (US)
- Europe / Middle East / Africa
  - EU (Ireland)
  - EU (Frankfurt)
- Asia Pacific
  - Asia Pacific (Tokyo)
  - Asia Pacific (Sydney)

Additional fees may apply for all data centers except US East (Northern Virginia).

Access

Logical access to the HunchLab AWS hosting account is limited to Azavea personnel working on the application. Access to the infrastructure is granted via 2-factor authentication using individual credentials for each employee. System development and testing occurs in a separate hosting account so that contact with client data is minimized. Client data is not copied outside of the AWS infrastructure without the explicit consent of the client. Statistical models and other diagnostic data that does not include disaggregated criminal justice information (CJI) may be accessed and examined outside of AWS by Azavea personnel for troubleshooting and support purposes.

More details of the AWS platform can be found in the current version of the *Amazon Web Services: Overview of Security Processes* document available for download at http://aws.amazon.com/security/.

Azavea's Data Use and Security Agreement

By default, Azavea agrees to solely use the law enforcement data to provide the agreed upon HunchLab service to the department including using the data for system testing, troubleshooting, and lives operations. Separately, Azavea may seek permission to use the data for research purposes that further the product and crime analysis in general. At no time will Azavea hold any claims to the data nor will Azavea use the data for other commercial purposes. Upon written request, Azavea will purge a customer's law enforcement data from its operational systems. Deletion from operational systems will occur within 7 days. The application maintains automated backup files for the last several weeks. Client data will expire out of these automated backups within 28 days from the request. If requested, Azavea will certify that client data has occurred.

Azavea will gladly sign a CJIS Security Addendum as specified in CJIS v5.3 section 5.1.1.5.

Security Awareness Training

Azavea hires technical staff with an eye toward building reliable and secure web applications. Part of the Azavea onboarding process is acknowledgement of company security practices as well as signing a separate agreement regarding confidentiality of client data. Additionally, staff members with access to the HunchLab system undergo biennial training on best practices when dealing with criminal justice information as outlined in CJIS v5.3 section 5.2.

Reliability and Security Incident Management

The HunchLab service is designed to be resilient to failure with redundancy built into the system architecture. Additionally, Azavea has implemented automatic monitoring of system uptime and incident alerts to provide timely resolution of system issues. In the event of a suspected or confirmed security breach, Azavea will proactively notify the law enforcement agency of the breach in a timely manner as specified in CJIS v5.3 section 5.3.2.

System Auditing

The HunchLab system keeps a running system log of activity by users including log-on attempts and information retrieval. These records are retained for at least 365 days. The auditing system is designed to comply with CJIS v5.3 section 5.4. Additionally, Azavea employs AWS services that log the logical access and control of the AWS environment.

Role-based Security

Access to system functionality is restricted based upon security roles. For instance, only a few users need administrative access to the system. This approach reflects the guidelines in CJIS v5.3 section 5.5.2.

Authentication Credentials

HunchLab can delegate credential management to 3rd party directory services such as Active Directory through the SAML standard. In that case, HunchLab assumes that the 3rd party directory service provides a CJIS compliant security model. Additionally, HunchLab can provide a stand-alone authentication system that complies with both the standard authentication and advanced authentication specifications in CJIS v5.3 sections 5.6.2.1 and 5.6.2.2. Our advanced authentication option provides 2-factor authentication using time-based tokens generated locally by mobile applications for mobile devices. Additional costs may apply if Azavea is managing 2-factor authentication on behalf of the client.

Password Management and Login Failures

If operating in stand-alone authentication mode, HunchLab stores user passwords in a salted cryptographic hash format which increases the computing power necessary to reverse engineer a user's password even if our database is comprised. Additionally, to prevent external attacks on user

credentials, the system keeps track of unsuccessful login attempts and locks the account for progressively longer periods of time. This policy is recommended in CJIS v5.3 section 5.5.3.

Session Lock

When a user logs into HunchLab, a temporary security token is kept within their local browser memory. HunchLab assumes that devices logging into the system will employ sessions locks or screensavers that meet the guidelines in CJIS v5.3 section 5.5.5.

Data Protection

The HunchLab service is hosted within Amazon Web Services (AWS) data centers. These data centers implement state-of-the-art security practices that protect the physical access to data within HunchLab as recommended in CJIS v5.3 section 5.9. Additionally, AWS continuously monitors their infrastructure against denial of service attacks and penetration vulnerabilities.

Within the HunchLab architecture, Azavea has utilized several security features of the AWS platform to harden the system. For instance, all inbound traffic to HunchLab is encrypted via SSL and terminates at a set of load balancers. These load balancers only allow secure HTTPS traffic with specific versions of the TLS protocol (TLS 1.2+) and specific encryption algorithms (AES) and proxy all traffic to the application. Each component of the application is isolated from all others with only the minimum required network traffic for each server instance granted. This security is enforced as inbound and outbound firewall rules on each server as well as redundantly at the network level.

While the physically secure AWS infrastructure constitutes a physically secure location and therefore encryption is not required, Azavea has decided to encrypt data in transit and at rest as much as feasible. All data in transit within the application is encrypted. Data stored on the elastic block storage devices attached to HunchLab servers and within the AWS S3 service is encrypted at rest. Additionally, data stored in the relational database provided by Amazon RDS is encrypted at rest.

These design approaches seek to conform to CJIS v5.3 section 5.10.

[Note: As of December 2015, there is one pending security features referenced above. We have added a caching layer within the application. This presently transmits data within the secure environment without encryption. We are working to encrypt the data being stored within the cache.]

Personnel

Upon request, Azavea will cooperate with the screening of Azavea personnel with access to the HunchLab system in line with CJIS v5.3 section 5.12.

CJIS Policy v5.3 Review

The following review of CJIS Security Policy version 5.2 outlines how HunchLab aligns with these guidelines.

Section	Requirement	Alignment
4.1	Defines Criminal Justice Information (CJI)	The required data set within HunchLab consists solely of crime event data. This data set does not include personally identifiable information. The most sensitive component of the data set is the location of incidence, but this section of the CJIS guidelines exempts property data when it is not accompanied by PII. As such, CJIS does not technically apply.
5.1.1.5	Private contractors are subject to the CJIS Security Addendum when handling CJI.	Azavea will gladly execute agreements in regards to the handling of CJI.
5.2	Security awareness training shall be required within six months of assignment and biennially thereafter for all personnel with access to CJI.	Azavea already conducts new employee briefs on guidelines and responsibilities in handling client data. Specifically to the team responsible for HunchLab, we are implementing focused training to comply with the minimum topics outlined in the CJIS guidelines.
5.2.2	Records of security training	Azavea shall keep records of security training for staff involved in HunchLab projects.
5.3.1	Security events shall be promptly reported.	Azavea shall promptly report security related events to the relevant clients.
5.4.1	Information systems shall generate audit records for specified events.	The HunchLab API logs user interactions that include the event types specified within the CJIS guidelines. Additionally, the AWS environment generates audit logs of management interactions with the hosting environment through the use of the AWS CloudTrail service.
5.4.3	Audit monitoring shall be conducted at minimum once a week by designated personnel.	The HunchLab environment generates system alerts upon suspicious activity with a view toward maintaining continuous monitoring of suspicious activity. For instance, increased levels of API requests that fail authentication generate alerts to the HunchLab team.
5.4.5	Protection of audit information from modification, deletion, and unauthorized access.	AWS level audit logs are kept in a secure S3 bucket with modification and deletion access limited to a subset of the HunchLab team. HunchLab API audit logs are kept securely within the hosting environment and end users are prevented from modifying or deleting these

		records.
5.4.6	Audit records shall be retains for at least one year.	Azavea will retain audit logs for at least one year.
5.5.1	Account management shall be in place to validate system accounts and permissions.	HunchLab client agencies manage user access to the system. Administrative access to the hosting environment by Azavea staff is reviewed regularly with only members of the team granted access.
5.5.2	Access enforcement shall be enforced to limit access to privileged functions.	HunchLab application functions are accessible via role-based system that limits access to administrative features within an organization's account. Additionally, components of the HunchLab application are only granted permissions within the AWS environment for systems that they need access to.
5.5.3	Unsuccessful login attempts shall be limited to no more than 5 consecutive invalid attempt per user followed by an automatic lock on the account for 10 minutes.	This login restriction is in place within the HunchLab application.
5.5.5	Session locks shall be in place to prevent access to the system after inactivity.	HunchLab assumes that client managed devices will implement screen locks or appropriate measures to meet this requirement.
5.5.6	Remote access shall be monitored and controlled.	By its nature a cloud service provides access over an untrusted network. Access to the application is controlled through login requirements. Access to the hosting environment itself is severely limited and requires multi factor authentication and cryptographic keys.
5.6.1	Identification policies should uniquely identify each user or administrator of the system.	All HunchLab users login with a unique identifier. The AWS environment is also managed through unique credentials assigned to each Azavea team member.
5.6.2.1.1	Passwords shall comply with stated attributes.	The AWS environment is managed through unique credentials assigned to each Azavea team member. These credentials include a password (that meets the stated requirement).

		HunchLab users can have password restrictions assigned to their accounts. Alternatively, if HunchLab is delegating authentication to another system, then that system would enforce such requirements.
5.6.2.2	Advanced authentication is required for publicly accessible services where the authenticity or security of the requesting device can not be established	HunchLab can either provide 2-factor authentication to end-users directly or can delegate authentication to a client agency to provide a compliant authentication methodology. Additional fees may apply. The AWS environment requires both a password and token to be entered for Azavea staff to access the hosting system.
5.8.1	Electronic and physical media shall be stored within physically secure locations. If not, then the data shall be encrypted.	The AWS hosting environment is a physically secure environment therefore data encryption is not required.
5.8.3	Electronic media shall be sanitized prior to reuse or disposal.	Media within the AWS hosting environment is sanitized before allocation to new customers. Additionally, AWS destroys all media that leaves its data centers for disposal.
5.9	Physically secure locations shall meet stated guidelines.	AWS provides details of its security policies for its data centers. Even Azavea as customers of the service are not permitted physical access to the environment.
5.10.1	The network infrastructure shall control the flow of information between connected systems.	The HunchLab application is comprised of distinct functional units. Each unit can only speak to the other units of the application that are necessary for it to complete its functions. Each server has a firewall that only allows inbound and outbound communication as needed. Additionally, the network enforces traffic controls to specific allowed ports. External access to the environment is limited to a single bastion server accessible only by Azavea.
5.10.1.2	Encryption shall protect data outside of the boundary of a physically secure location when being transmitted or encrypted. Cryptographic modules shall be	External access to HunchLab is over HTTPS. The application only permits TLS 1.2 due to flaws in earlier TLS versions. The server is configured to use either 128/256bit GCM or 256bit CBC AES encryption and prefers ephemeral key exchange that provides forward secrecy (ECDHE).

	certified to meet FIPS 140-2 standards	The application uses Amazon's Elastic Load Balancers to terminate inbound SSL connections. These load balancers do not use certified cryptographic modules unless the application is hosted within the GovCloud environment, which is available for an additional fee.
5.10.1.3	Intrusion detection shall be implemented.	AWS manages intrusion detection and abuse of their environment. Additionally, HunchLab logs inbound requests to monitoring servers that provide Azavea staff with a real time view of activity.
5.10.1.5	The metadata derived from CJI shall not be used by any cloud provider for any purposes.	Azavea will not use CJI for any purposes other than to provide this service. AWS also agrees to not use client data for any purposes.
5.10.3.1	Partitioning shall separate user functionality from information management functionality.	The HunchLab application is broken up into discrete segments that separate functionality. For instance, an inbound request for data first arrives at a load balancer, which terminates the inbound SSL connection, parses the request, and then wraps the request in a new SSL connection to pass to the web servers. The web servers then receive the request, validate the user's credentials and query the database for needed data. The database also resides on a separate virtual machine.
5.10.3.2	Virtualization shall be implemented to isolate machines.	Firewalls are in place that restrict access to each machine within the environment. For instance, the load balancers may not directly communicate with the database server. Requests from the load balancers to the web servers and from the web servers to the database server are encrypted. Requests from all servers to the S3 object store are all enforced as encrypted. Log files on each machine are centrally aggregated for monitoring.
5.10.4.1	Patches shall be maintained.	Azavea maintains a staging environment to validate updates to software. The application utilizes OS releases that are currently supported with security patches. OS security patches are

		applied upon each deployment of the software via golden images for machines. Additionally, Azavea updates other software packages on a regular basis based upon the severity of the patch.
5.10.4.2	Malicious Code Protection	HunchLab utilizes only Linux based software. It is atypical to run antivirus software on such systems due to the security design of the systems. Additionally, the hosting environment is designed for the rapid replacement of server instances based upon golden images. For instance, no persistent data is stored on web application servers. Upon every deploy of an update, the existing servers are destroyed and replaced with new servers running from a clean image. This approach eliminates the likelihood of an infection of maintaining itself.
5.12.1	Personnel will have fingerprint-based record checks.	Azavea is happy to have relevant staff cleared through these processes.
5.12.2	Upon termination, access to CJI shall be terminated immediately	Azavea maintains a checklist of termination practices, which includes removal of access to the HunchLab environment.

Appendix F: Support and Service Level Agreement

Service Level

During the Term of the HunchLab subscription, Azavea shall make commercially reasonable efforts to maintain the operation and availability of the application to the Customer at least 99.9% of the time as measured over the course of a calendar month. This SLA level corresponds with approximately 44 minutes of unplanned downtime per month. Scheduled downtime will not be included in these calculations but generally will amount to less than 1 hour per month. If the application does not meet the SLA, the Customer may request Service Credit as described below. This SLA states the Customer's sole and exclusive remedy for any failure by Azavea to provide the service.

Definitions

The following definitions shall apply to the SLA:

- *"Downtime"* means that the application and API are unavailable as measured by availability and a valid response being received from an external monitoring service maintained by Azavea.
- *"Downtime Period"* means a period of two consecutive minutes of Downtime. Intermittent Downtime for a period of less than two minutes will not be counted towards any Downtime Periods.
- *"Scheduled Downtime"* means those times where Azavea notifies Customer of periods of Downtime at least five calendar days prior to the commencement of such Downtime. There will be no more than eighteen hours of Scheduled Downtime per calendar year. Scheduled Downtime is not considered Downtime for purposes of this SLA, and will not be counted towards any Downtime Periods.
- *"Monthly Uptime Percentage"* means total number of minutes in a calendar month minus the number of minutes of Downtime suffered from all Downtime Periods in a calendar month, divided by the total number of minutes in a calendar month. For purposes of the Server Uptime Level, a lapse in server availability is calculated from the time Azavea detects or otherwise becomes aware of an incidence of a service interruption and ending when the service is restored, regardless of where the outage originated.
- *"Service Credit"* means the following:

Monthly Uptime Percentage	**Service Credit added to the end of the Service Term, at no charge to Customer**
< 99.9% and ≥ 95.0%	1 additional week added to the subscription
< 95.0%	2 additional weeks added to the subscription

In order to receive any of the Service Credits described above, Customer must notify Azavea within thirty days from the time Customer becomes eligible to receive a Service Credit. Failure to comply with this requirement will forfeit Customer's right to receive a Service Credit.

Service Credits

Service Credits may not be exchanged for, or converted to, monetary amounts.

Monitoring

Azavea shall maintain a monitoring service external to the data center housing the equipment supporting the application and shall monitor the service with reasonable frequency and duration.

Scheduled Downtime

Azavea shall provide at least 5 calendar days or more notice to the Project Contact and IT Contact if there is planned downtime. Tasks performed during planned downtime may include:

- Application of security patches
- Upgrades to software
- Updates to the database
- Other activities as necessary to maintain the integrity, stability and performance of the web services.

SLA Exclusions

The SLA does not apply to unavailability or any performance issues:

(i) caused by factors outside of Azavea's reasonable control including without limitation, acts of God, acts of government, flood, fire, earthquakes, civil unrest, acts of terror, strikes or other labor problems.; or

(ii) caused by a malicious internet attack including a denial of service attack; or

(iii) that resulted from Customer's equipment or activity or third party equipment or activity, or both (not within the primary control of Azavea)

Changes to Service

Azavea may make commercially reasonable modifications to the Service, or particular components of the Service, from time to time. Azavea will use commercially reasonable efforts to notify Customer of such changes.

Security Incident Management

In the event of a suspected or confirmed security breach, Azavea will proactively notify the Project Contact and Security Contact (these roles are defined in the Implementation Narrative on page 17) of the breach in a timely manner as specified in CJIS v5.3 section 5.3.2.

Customer Support

Azavea shall use commercially reasonable efforts to provide the following support services for customers:

- Provide telephone, web and/or email support to customers during normal business hours (9am – 6pm, Eastern Time); and

Respond to customer support queries regarding within one to five business days, depending on the severity of the issue.