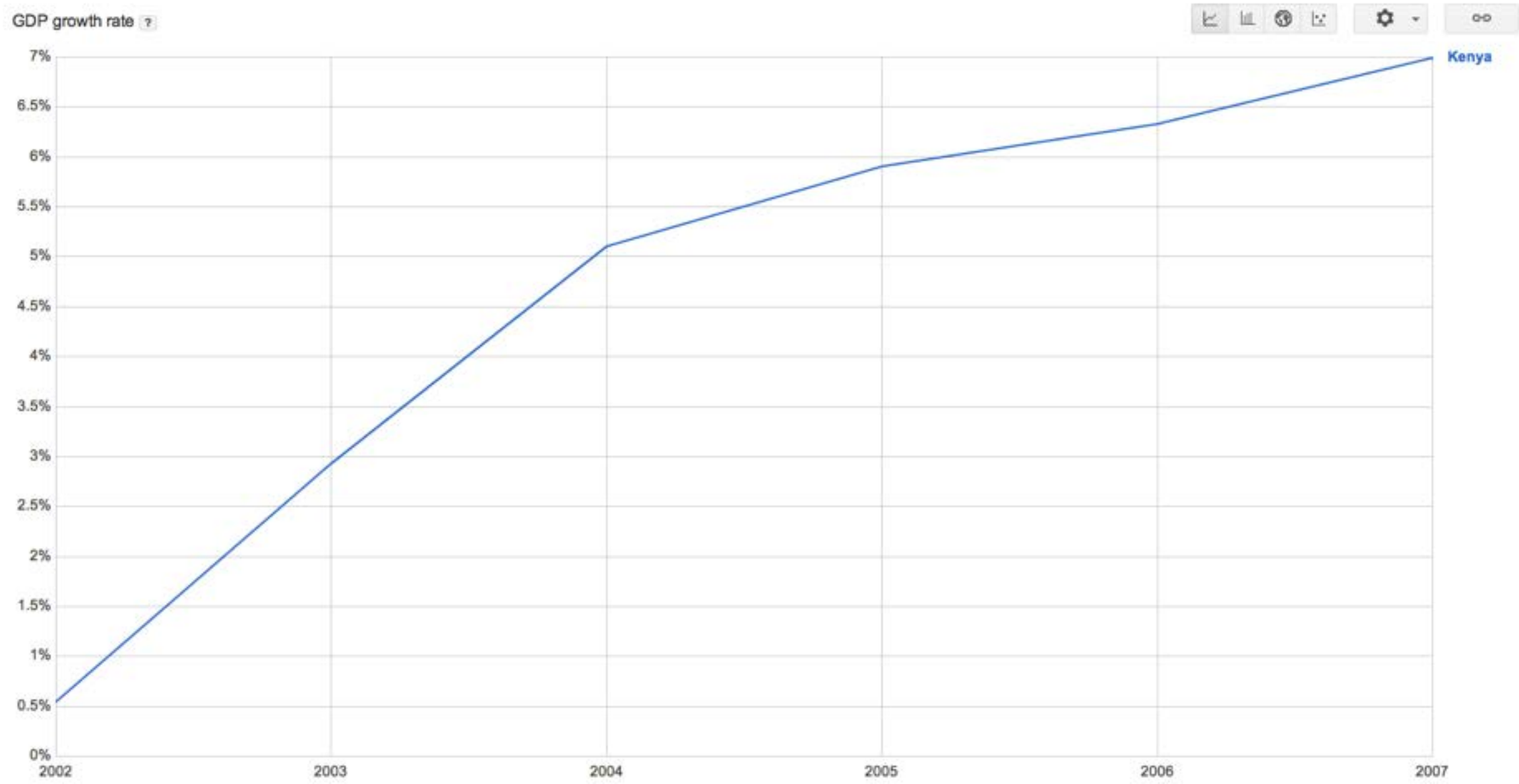
Ushahidi


GDP growth rate
Kenya
7%
6.5%
6%
5.5%
5%
4.5%
4%
3.5%
3%
2.5%
2%
1.5%
1%
0.5%
0%
2002
2003
2004
2005
2006
2007

2008

underreported

unreported

Ushahidi

We build tools for **democratizing** information, increasing **transparency** and lowering the barriers for individuals to share their stories.
We're a **disruptive** organization that is willing to take risks in the pursuit of changing the traditional way that information flows in the
openness | innovation | community

159 Countries, 49 Languages

Afghanistan, Albania, Argentina, Austria, Australia, Bosnia and Herzegovina, Barbados, Bangladesh, Belgium, Bulgaria, Bahrain, Benin, Bolivia, Brazil, Bahamas, Bhutan, Botswana, Burundi, Belarus, Canada, Cape Verde, Democratic Republic of Congo, Switzerland, Cote d'Ivoire, Chile, Cameroon, China, Colombia, Czech Republic, Germany, Denmark, Dominican Republic, Algeria, Ecuador, Estonia, Egypt, Spain, Ethiopia, Finland, France, United Kingdom, Ghana, Guadeloupe, Greece, Guatemala, Guyana, Honduras, Croatia, Haiti, Hungary, Indonesia, Ireland, Israel, India, Iraq, Iran, Islamic Republic of, Italy, Japan, Kenya, Cambodia, Korea, Republic of, Kuwait, Kazakhstan, Lao People's Democratic Republic, Lebanon, Lesotho, Sri Lanka, Liberia, Lithuania, Latvia, Liberia, Libya, Morocco, Moldova, Republic of, Montenegro, Macedonia, Mozambique, Mali, Myanmar, Malta, Malawi, Mexico, Malaysia, Namibia, New Caledonia, Niger, Nigeria, Nicaragua, Netherlands, Norway, Nepal, New Zealand, Oman, Peru, Papua New Guinea, Philippines, Pakistan, Poland, Portugal, Paraguay, Qatar, Reunion, Romania, Rwanda, Serbia, Russian Federation, Saudi Arabia, Sudan, Sweden, Singapore, Slovenia, Sierra Leone, Senegal, Swaziland, Somalia, South Africa, El Salvador, Syria, Chad, Thailand, Tajikistan, Timor-Leste, Tunisia, Turkey, Taiwan, Tanzania, United Republic of, Ukraine, Uganda, United Arab Emirates, United States, Venezuela, Samoa, Yemen, Zambia, Zimbabwe

Made in frica

Disaster Response

March 2011 - 4,000+ reports, 144,974 views
Monitor fallout of tsunami and nuclear crisis

sinsai.info/ushahidi

January 2010 - 3,584 reports, 500,000 views
Allowed Haitians to report location and needs

haiti.ushahidi.com

Ushahidi

TIME LINE

From: Feb 2011 To: Apr 2011 PLAY

Jan 2011 Apr 2011 Jul 2011 Dec 2011

2000 1500 1000 500 0

3 Mar 8 Mar 13 Mar 18 Mar 23 Mar 28 Mar

FILTERS:

BOOLEAN OPERATORS:

OR What's this?

AND What's this?

CATEGORIES:

全カテゴリ

お知らせと情報

その他

信頼できる情報

利用可能なサービス

インフラの状態

↓ 他のレイヤー [見る]

Scale = 1 : 7M | 123.96638, 34.61008 | EPSG:900913

100 km

100 mi

English (US)

خريطة التحرش الجنسي

HARASSmap

SMS 0169870900

SUBMIT A REPORT

HOME | REPORTS | SUBMIT A REPORT | GET ALERTS | CONTACT US | HOW TO HELP | ABOUT | VOLUNTEER

FILTERS → REPORTS | NEWS | PICTURES | VIDEO | ALL

FULL SCREEN MAP

From: Oct 2004 To: Mar 2011 Play

Oct 2004 Jun 2006 Feb 2008 Mar 2011

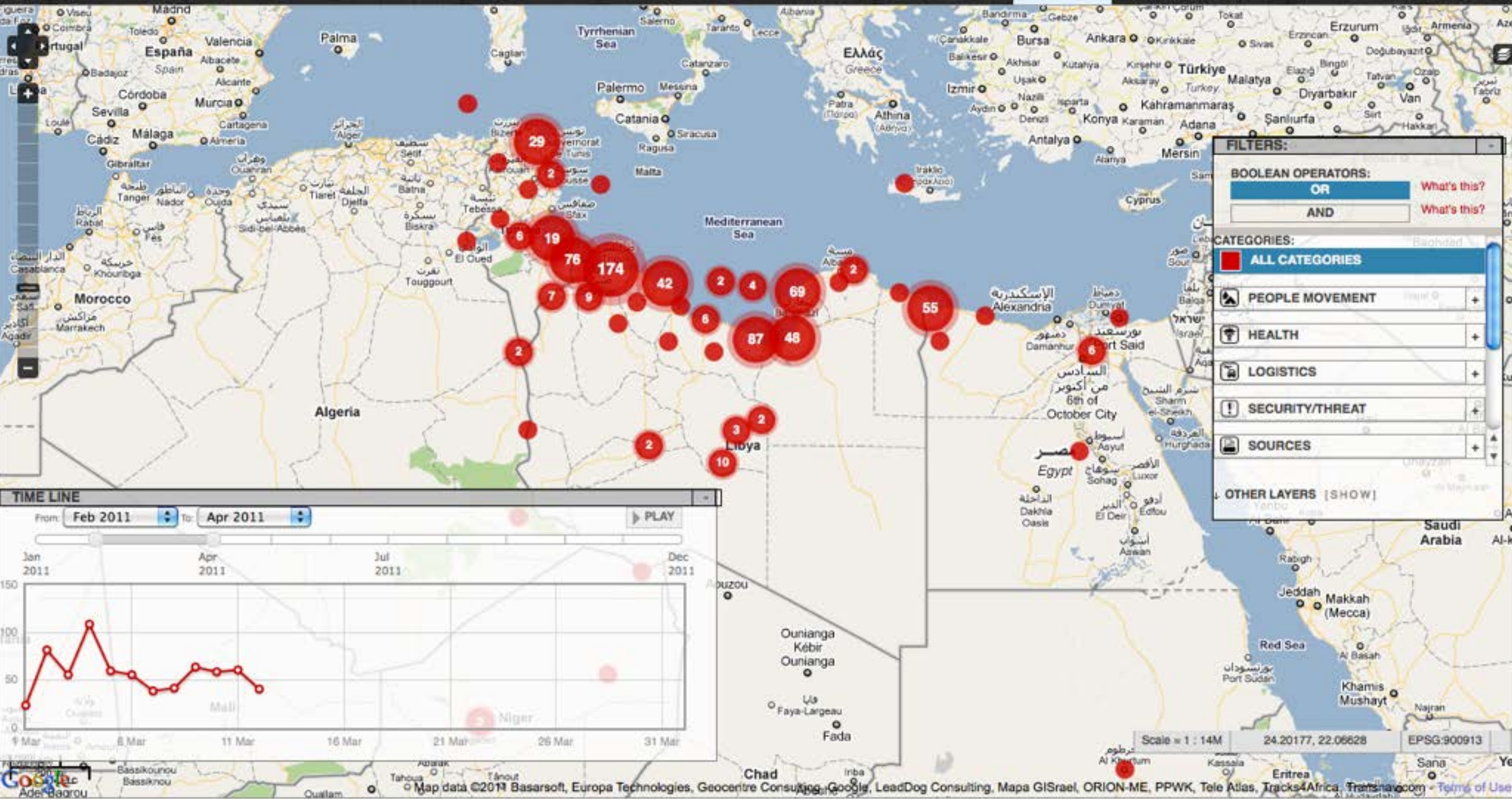
2004 2005 2006 2007 2008 2009 2010 2011

CATEGORY FILTER [HIDE]

- ALL CATEGORIES
- TOUCHING
- CATCALLS
- OGLING
- COMMENTS
- STALKING OR FOLLOWING
- PHONE CALLS
- INDECENT EXPOSURE
- SEXUAL INVITES
- FACIAL EXPRESSIONS
- RAPE / SEXUAL ASSAULT

How to Report

1. By sending a message to **0169870900**


FILTERS:
BOOLEAN OPERATORS:
OR
What's this?
AND
What's this?
CATEGORIES:
ALL CATEGORIES
PEOPLE MOVEMENT
HEALTH
LOGISTICS
SECURITY/THREAT
SOURCES
OTHER LAYERS [SHOW]
TIME LINE
From: Feb 2011
To: Apr 2011
PLAY
Jan 2011
Apr 2011
Jul 2011
Dec 2011
Mediterranean Sea
Algeria
Libya
Egypt
Chad
Mali
Niger
Saudi Arabia
Red Sea
Scale = 1 : 14M
24.20177, 22.06628
EPSG:900913
Map data ©2011 Basarsoft, Europa Technologies, Geocentre Consulting, Google, LeadDog Consulting, Mapa GISrael, ORION-ME, PPWK, Tele Atlas, Tracks4Africa, Transnavi.com - Terms of Use

Technology is only 10% of the story

UCHAGUZI

Hivos
people unlimited

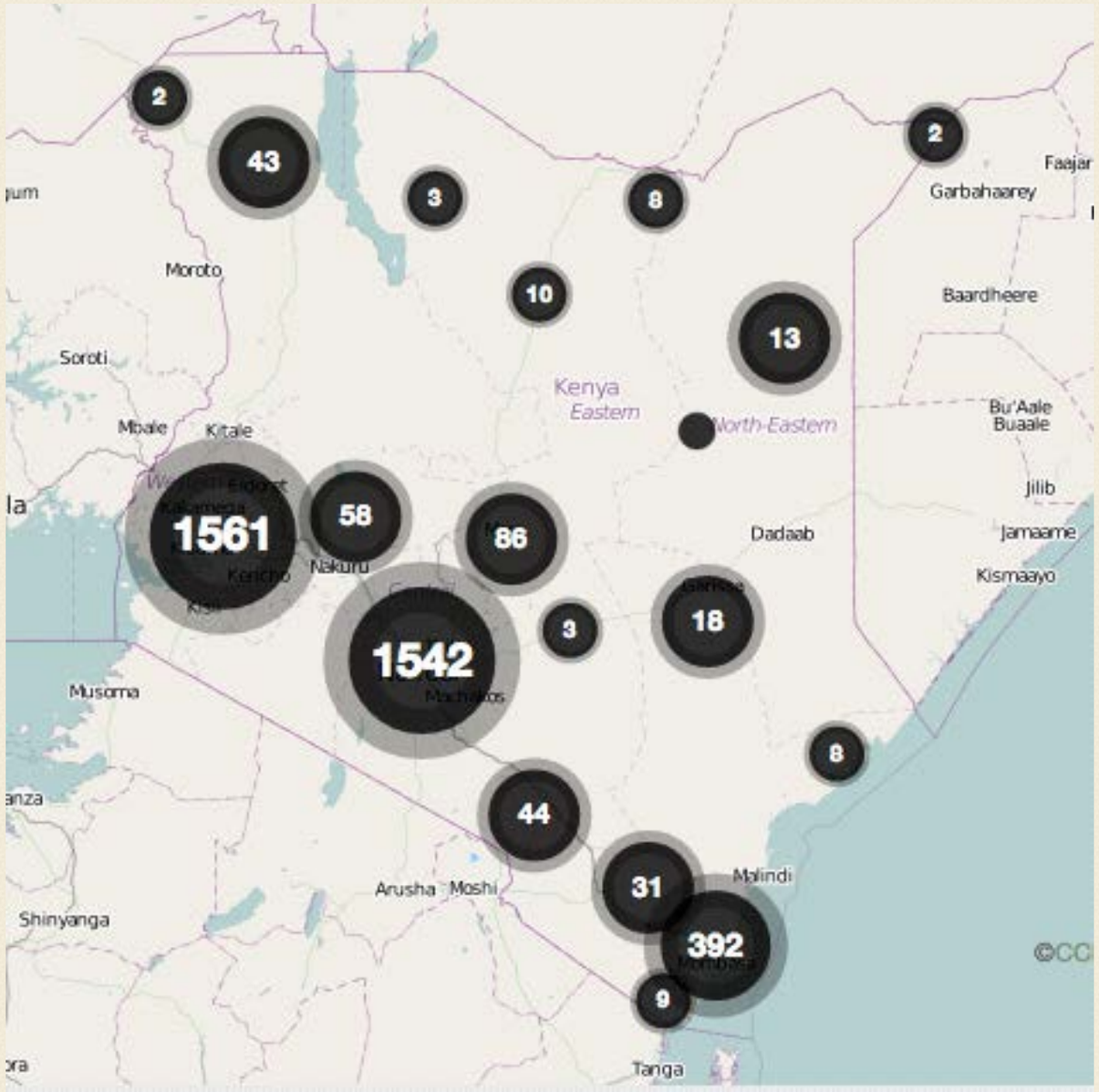
CONSTITUTION AND REFORM EDUCATION CONSORTIUM
CRE-CO
MUUNGANO WA ELIMU YA KATIBA

IN SYNERGY THERE IS STRENGTH
SODNET
SOCIAL DEVELOPMENT NETWORK

Ushahidi

Ushahidi

Uchaguzi Kenya 2013



- 5313 Messages Received
- 5140 Reports Created
- 4502 Reports Approved
- 2700 Verified

Ushahidi

uchaguzi.co.ke

DATA

↓

VERIFICATION

↓

RESPONSE

Ushahidi

Linda KURA Yako

Uchaguzi: On The Ground

- 259 Uchaguzi Supervisors
- 777 Uchaguzi Observers
- 5 Regional Clusters
- Training: Mombasa, Kisumu, Meru, Nairobi, Eldoret, Wajir, Turkana
- 66 KNCHR observers
- 35 CUEA observers

Ushahidi

Digital Humanitarians

Ushahidi

11 Active Situation Room Teams

- SMS
- Media Monitoring
- Geolocation
- Translation
- Reports
- Verification,
- Analysis & Research
- Quality Assurance
- Technology
- Communications
- Uchaguzi Strategy Team

UCHAGUZI

We've been posting numbers all week about reports, types of reports, awaiting approval, awaiting verification etc. But the big community story is how we collectively made this possible.

11 Number of Skype Subteams Active

50 Number of Training and Meeting Sessions with Field Partners

14 Number of Community Working Group and Digital training team sessions.

239 Number of people trained for digital teams from Feb 11th - Mar 3rd

218 Number of people added to skype teams from Mar 3-7th

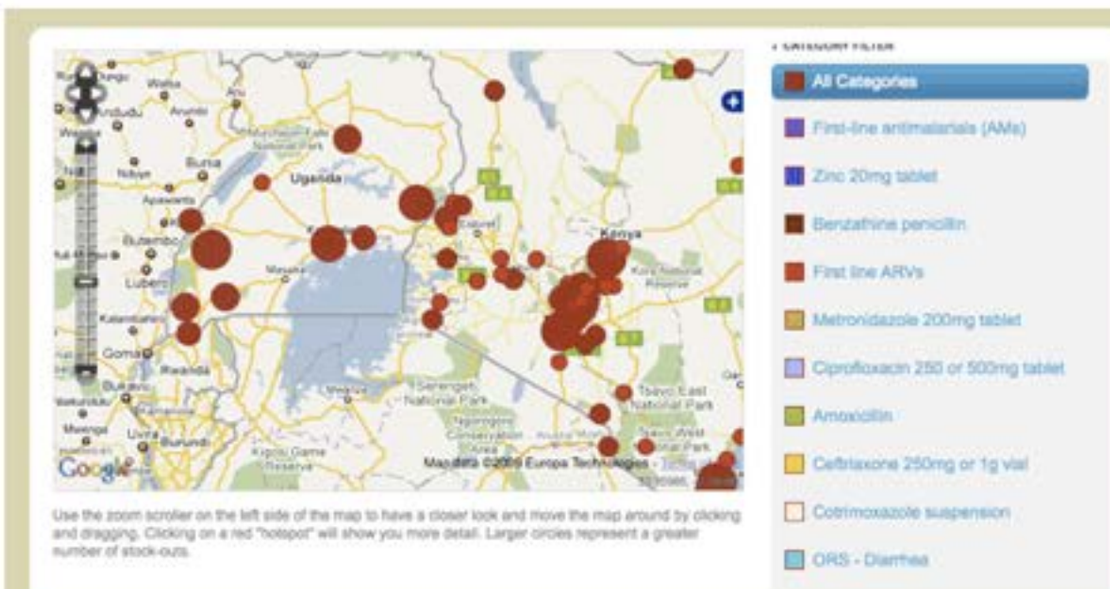
UCHAGUZI COMMUNITY

24/7

Live mode: Mar 2-5th 2013
Monitoring mode: Mar 1, Mar 5 - ongoing

Ushahidi

Snap survey reveals medicine stockouts still plague Kenya, Malawi, Uganda and Zambia



Use the zoom scroller on the left side of the map to have a closer look and move the map around by clicking and dragging. Clicking on a red "hotspot" will show you more detail. Larger circles represent a greater number of stock-outs.

During 'pill check week', researchers visited public health institutions countrywide and checked on the availability of a list of 10 essential medicines. These are medicines that should be readily available in public

EVENTS

Kisumu residents demand for essential medicines
Tuesday 14 July 2009 Kisumu: People thronged the Pollyview Hotel in Kisumu, the third...

SMS pill check findings released to Kenyan media
The Kenyan team held a successful press conference on 30/06/09 with 22 journalists...

Mombasa forum a huge success!
The Kenyan team organised a successful public forum in Mombasa, the second largest...

More Events

BE UPDATED

Sign up to receive breaking news as well as receive other site updates!

Enter your email address... GO

STOP
STOCK-OUTS!

CRISIS
NET

I want social media reports of bombings in Syria

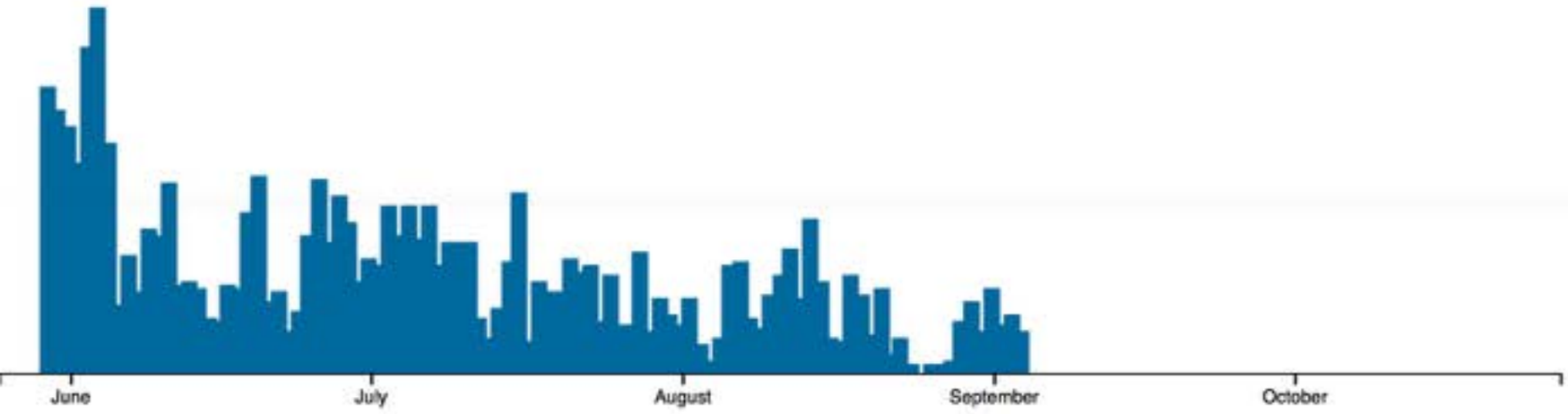
```
// example source http://censorbugbear.org/farmitracker/api?
task=incidents
// data['payload']['incidents'][0]['incident']['incidenttitle']

var Sucka = require("./sucka")
  , logger = require("winston")
  , moment = require("moment")
  , _ = require("underscore")
  , _s = require("underscore.string")
  , request = require("request")
  , async = require('async');

var UshahidiSucka = function() {};
UshahidiSucka.prototype = Object.create(Sucka.prototype);

UshahidiSucka.definition = {
  internalID: '0a016230-d72a-4f48-bcf6-0964caeafa70',
  sourceType: "ushahidi",
  frequency: "repeats",
  repeatsEvery: "hour",
  startDate: moment('2014-03-30', 'YYYY-MM-DD'),
  endDate: moment('2015-04-05', 'YYYY-MM-DD')
};

UshahidiSucka.prototype.getInstanceIDs = function(callback) {
  var that = this;
  var url = 'http://tracker.ushahidi.com/list';

  var propertiesObject = {
    return_vars: "url",
    limit: "0,5000",
    minpopularity: 100
  };

  request({url:url, qs:propertiesObject, json:true}, function(err,
response, body) {
    var keys = _(response.body).keys();
    var tuples = keys.map(function(key) { return [key,
response.body[key].url + 'api'] });
    callback.call(that, err, tuples);

    response.body = null;
  });
};

UshahidiSucka.prototype.getInstanceData = function(err, tuples) {
  var that = this;
  var propertiesObject = {
    task: 'incidents'
  };

  var funcs = [];
  that.lastRetrieved = that.lastRetrieved || {};
  var totalRetrieved = tuples.length;
  var totalProcessed = 0;

  _(tuples).each(function(tuple) {

    if(that.lastRetrieved[tuple[0]]) {
      propertiesObject.by = 'sinceid';
      propertiesObject.id = that.lastRetrieved[tuple[0]];
    }

    request({url: tuple[1], qs: propertiesObject, json:true},
function(err, response, body) {
      //that.returnData({'remoteID': tuple[0], 'source':'test'});
      if(response.body && response.body.payload &&
response.body.payload.incidents) {
        that.transform(response.body.payload.incidents, tuple[0]);
      }
      totalProcessed++;

      if(totalProcessed >= totalRetrieved) {
        that.allFinished(that.lastRetrieved);
      }
      return null;

    });
  });

};


UshahidiSucka.prototype.suck = function() {
  var that = this;
  that.getInstanceIDs(that.getInstanceData);
};


/**
* Transform a list of tweets to items matching the CrisisNET schema.
*/
UshahidiSucka.prototype.transform = function(inputData, instanceID) {
  var that = this;

  //console.log(inputData);

  var outputData = _(inputData).map(function(item) {
    var transformed = {
      remoteID: instanceID + "-" + item.incident.incidentid,
      publishedAt: new Date(item.incident.incidentdate),
      content: item.incident.incidentdescription,
      summary: item.incident.incidenttitle,
      lifespan: "temporary",
      geo: (function() {
        var data = {
          addressComponents: {
            formattedAddress: item.incident.locationname
          }
        };

        if(item.incident.locationlongitude &&
item.incident.locationlatitude) {
          data.coords = [
            item.incident.locationlongitude,
            item.incident.locationlatitude
          ]
        }
        return data;
      })(),
      source: "ushahidi",
      tags: (function() {
        return item.categories.map(function(category) {
          return {
            name: category.category.title,
            confidence: 1
          }
        });
      })()
    };

    return transformed;
  });

  inputData = null;

  _(outputData).each(function(item) {
    that.returnData(item);
  });


  var lastRetrievedDoc = _(_(outputData).sortBy(function(doc) {
    return (new Date(doc.publishedAt)).getTime();
  })).last();

  that.lastRetrieved[instanceID] = lastRetrievedDoc.remoteID;

  return outputData;
};

module.exports = UshahidiSucka;

var Sucka = require("./sucka")
  , logger = require("winston")
  , moment = require("moment")
  , _ = require("underscore")
  , _s = require("underscore.string")
  , request = require('request')
  , fs = require('fs')
  , unzip = require('unzip')
  , validator = require('validator')
  , Promise = require("promise");


var ReliefSucka = function() {};
ReliefSucka.prototype = Object.create(Sucka.prototype);

ReliefSucka.definition = {
  internalID: '883885b6-7baa-46c9-ad30-f4ccc0945674',
  sourceType: "reliefweb",
  frequency: "repeats",
  repeatsEvery: "day",
  startDate: moment('2014-03-30', 'YYYY-MM-DD'),
  endDate: moment('2014-04-01', 'YYYY-MM-DD')
};


ReliefSucka.prototype.suck = function() {
  var that = this;

  var url = 'http://api.rwlabs.org/v0/disaster/list';
  var propertiesObject = {
    fields: {
      include:[
        'name',
        'description',
        'primary_country',
        'date',
        'country',
        'primary_type',
        'type'
      ]
    },
    limit: 100
  };

  if(!_.isUndefined(that.source.lastRetrieved)) {
    propertiesObject.filter = {
      field: "date.created",
      value: {
        from: (new
Date(that.source.lastRetrieved.publishedAt)).getTime()
      }
    }
  }

  var collectData = function(offset, memo, total) {
    if(total && offset >= total) {
      that.transform(memo);
      return;
    }

    request({url:url, qs:propertiesObject, json:true}, function(err,
response, body) {
      if(err) { that.handleError(err); return; }

      var data = response.body.data;

      collectData(offset + data.count, memo.concat(data.list),
data.total);

    });
  };

  collectData(0, []);
};


/**
* Transform a list of tweets to items matching the CrisisNET schema.
*/
ReliefSucka.prototype.transform = function(inputData) {
  var that = this;

  var outputData = [];

  var transformItem = function(item) {
    var returnData = {
      remoteID: item.idToUse,
      publishedAt: new Date(moment(item.fields.date.created)),
      lifespan: "temporary",
      content: item.fields.description || item.fields.name,
      summary: item.fields.name,
      geo: {
        addressComponents: {
          adminArea1: item.singleCountry
        }
      },
      language: {
        code: 'en',
      },
      source: "reliefweb",
      tags: (function() {
        return item.fields.type.map(function(tag) {
          return { name: _s.slugify(tag.name), confidence: 1 }
        });
      })()
    };

    return returnData;

  };


  // Data from ReliefWeb can be associated with multiple countries.
The
  // occurence of this disaster in each country is considered a
separate
  // incident, we create a separate document.
  _(inputData).each(function(item) {
    _(item.fields.country).each(function(country, idx) {
      item.singleCountry = country.name;

      // Every item must have a unique remote id, so because we create
multiple
      // documents from a single returned record, we need to create a
modified
      // remoteid for "child" documents
      if(idx === 0) {
        item.idToUse = item.id;
      }
      else {
        item.idToUse = item.id + "-" + idx;
      }

      outputData.push(transformItem(item));
    });
  });

  _(outputData).each(function(item) {
    that.returnData(item);
  });

  var lastRetrievedDoc = _(_(outputData).sortBy(function(doc) {
    return (new Date(doc.publishedAt)).getTime();
  })).last();

  that.allFinished(lastRetrievedDoc);

  return outputData;
};

module.exports = ReliefSucka;

var Sucka = require("./sucka")
  , logger = require("winston")
  , moment = require("moment")
  , csv = require("fast-csv")
  , _ = require('underscore')
  , _s = require("underscore.string")
  , fs = require('fs')
  , unzip = require('unzip')
  , request = require('request')
  , logger = require('winston')
  , validator = require('validator')
  , Promise = require("promise");


var Gdelt = function() {};
Gdelt.prototype = Object.create(Sucka.prototype);

Gdelt.definition = {
  internalID: '6589847b-57f0-425e-bb6b-c0ce4fd1bafa',
  sourceType: "gdelt",
  frequency: "repeats",
  repeatsEvery: "day",
  startDate: moment('2014-03-19', 'YYYY-MM-DD'),
  endDate: moment('2014-03-22', 'YYYY-MM-DD')
};

Gdelt.prototype.suck = function() {
  var that = this;

  logger.info("Gdelt.suck starting suck...");
  that.setupData();

  // Where we're putting the file we retrieve from Gdelt website
  var zipFilePath = __dirname + '/data/gdelt/latest-daily.zip';
  var today = moment().subtract('days',2).format('YYYYMMDD');
  var file = fs.createWriteStream(zipFilePath);
  var url = 'http://data.gdeltproject.org/events/'+ today +
'.export.CSV.zip';

  // Get the file and pipe it into our writeStream
  request(url, function(err, resp, body) {
    if(err) {
      that.handleError(err);
    }
    if(resp.statusCode !== 200) {
      return logger.warn('Gdelt.suck did not retrieve anything from '
+ url + ' code:' + resp.statusCode);
      this.allFinished();
    }

    logger.info('Gdelt.suck retrieved ' + url + ' with code ' +
resp.statusCode);
  }).pipe(file);

  // Once the file has finished writing we'll read it, unzip it, and
parse the
  // csv file in the extracted directory
  file.on('finish', function() {
    logger.info("Gdelt suck finished writing zip");
    file.end();
    file = null;

    that.processData(zipFilePath, null, null, today);
  });
};

/**
* Unzip the file, find relevant records in the csv and stash those in
a
* new file (so we can stream them - it's 10k+ records), read the new
* relevant records csv and transform each row. When we're done reading
the
* relevant records file we let the main process know we're finished
with
* this suck.
*/

Gdelt.prototype.processData = function(zipFilePath, newFilePath,
relevantOutputPath, today) {
  var that = this;

  that.unzipFile(zipFilePath, today).then(function(newFilePath) {
    that.findRelevantRecords(newFilePath, relevantOutputPath)
      .then(function(relevantFilePath) {
        logger.info('Gdelt.processData have relevant records');
        var relevantStream = fs.createReadStream(relevantFilePath);
        var lastTransformed;

        csv.fromStream(relevantStream, {delimiter: "\t"})
        .on("record", function(record){
          lastTransformed = that.transform(record);
        })
        .on("end", function(){
          logger.info('Gdelt.suck finished transforming relevant
records');
          that.allFinished(lastTransformed);
        });
      });
  });

};

Gdelt.prototype.unzipFile = function(zipFilePath, today, newFilePath)
{
  return new Promise(function(resolve, reject) {
    fs.createReadStream(zipFilePath)
    .pipe(unzip.Parse())
    .on('entry', function (entry) {
      logger.info("Gdelt.unzipFile We have an entry");
      var fileName = entry.path;

      // We only want to deal with the CSV - it should be the only
thing in there
      if (fileName === today + ".export.CSV") {
        logger.info("Gdelt.unzipFile entry is a match")
        if(!newFilePath) {
          newFilePath = __dirname + '/data/gdelt/gdelt-latest.csv';
        }
        var newFile = fs.createWriteStream(newFilePath);
        entry.pipe(newFile);

        // Now that we've finished writing the extracted contents to
another
        // new file, let's read that and send each line to our
transform method
        newFile.on('finish', function() {
          logger.info("Gdelt.unzipFile zip file extracted");
          resolve(newFilePath);
          newFile.end();
          newFile = null;
        });
      }
      else {
        entry.autodrain();
      }
    });
  });
};

Gdelt.prototype.findRelevantRecords = function(csvFilePath,
outputPath, dateString) {
  var that = this;

  if(!dateString) {
    dateString = moment().subtract('days', 2).format('YYYY-MM-DD');
  }

  if(!outputPath) {
    outputPath = __dirname + '/data/gdelt/gdelt-latest-relevant.csv';
  }

  return new Promise(function(resolve, reject) {
    var stream = fs.createReadStream(csvFilePath);
    var outputStream = fs.createWriteStream(outputPath, {flags: 'a'});

    csv.fromStream(stream, {delimiter: "\t"})
    .on("record", function(record){
      recordObject = that.recordToObject(record);
      if(that.shouldTransform(recordObject, dateString)) {
        outputStream.write(record.join('\t') + '\n');
      }
    })
    .on("end", function(){
      logger.info('Gdelt.findRelevantRecords finished csv parsing');
      outputStream.end();
      outputStream = null;
      resolve(outputPath);
    });
  });
};

/**
* GDELT data is like a SQL dump. Many columns in the CSVs are foreign
key
* references to other tables. We're storing that additional GDELT
information
* in JSON files on the local filesystem.
*/
Gdelt.prototype.setupData = function() {
  var path = './data/gdelt/';

  if(!this.columns) {
    this.columns = require(path + 'column-names.json');
  }

  if(!this.ethnicities) {
    this.ethnicities = require(path + 'ethnicities.json');
  }

  if(!this.eventTypes) {
    this.eventTypes = require(path + 'event-types.json');
  }

  // Note the `.js` extension. This is due to a strange naming
convention
  // GDELT uses for event description codes. The initial zero in many
of the
  // codes is removed by default when the object is interpreted as
JSON.
  if(!this.eventDescriptions) {
    this.eventDescriptions = require(path + 'event-descriptions.js');
  }

  if(!this.religions) {
    this.religions = require(path + 'religions.json');
  }
};


/**
* Just like how it sounds. Takes a row from the CSV and assigns each
value
* to its corresponding header in an object. Just for convenience/
readability.
*/
Gdelt.prototype.recordToObject = function(record) {
  var data = {};

  if(_(this.columns).isEmpty()) {
    logger.warn("Gdelt.recordToObject no columns found");
  }

  _(this.columns).each(function(column, idx) {
    data[column] = record[idx];
  });

  return data;
};


/**
* GDELT does quite a bit of categorization already, noting any
religions,
* known groups, ethnicities, etc mentioned in an event. We layer our
own
* more generic categorization on top of that based on the event
description.
*/
Gdelt.prototype.determineTags = function(recordObject) {
  var that = this;

  var tags = [];

  var findValue = function(list, property) {
    return _(list).filter(function(item) {
      return item.code.toString() ===
recordObject[property].toString();
    });
  };

  var pushIfExists = function(list) {
    if(!_(list).isEmpty()) {
      tags.push(list[0].label);
    }
  };

  var tagsForProps = function(props, list) {
    _(props).each(function(prop) {
      var found = findValue(list, prop);
      pushIfExists(found);
    });
  };

  tagsForProps(['Actor1EthnicCode', 'Actor2EthnicCode'],
that.ethnicities);
  tagsForProps(['Actor1KnownGroupCode', 'Actor2KnownGroupCode'],
that.knownGroups);
  tagsForProps(['Actor1EthnicCode', 'Actor2EthnicCode'],
that.ethnicities);
  tagsForProps(['Actor1Religion1Code', 'Actor1Religion2Code',
    'Actor2Religion1Code', 'Actor2Religion2Code'], that.religions);
  tagsForProps(['Actor1Type1Code',
'Actor1Type2Code', 'Actor1Type3Code',
    'Actor2Type1Code','Actor2Type2Code','Actor2Type3Code'],
that.eventTypes);

  var codeToTag = {
    '14': ['protest', 'conflict'],
    '145': ['physical-violence', 'riot', 'conflict'],
    '1451': ['physical-violence', 'riot', 'conflict'],
    '1452': ['physical-violence', 'riot', 'conflict'],
    '1453': ['physical-violence', 'riot', 'conflict'],
    '1454': ['physical-violence', 'riot', 'conflict'],
    '1711': ['property-loss', 'conflict'],
    '1712': ['property-loss', 'conflict'],
    '173': ['arrest', 'conflict'],
    '174': ['deportation', 'conflict'],
    '175': ['physical-violence', 'conflict'],
    '181': ['physical-violence', 'abduction', 'conflict'],
    '18': ['physical-violence', 'conflict'],
    '1821': ['physical-violence', 'sexual-violence', 'conflict'],
    '1822': ['physical-violence', 'torture', 'conflict'],
    '1823': ['physical-violence', 'death', 'conflict'],
    '183': ['physical-violence', 'death', 'suicide-bombing',
'conflict'],
    '1831': ['physical-violence', 'death', 'suicide-bombing',
'conflict'],
    '1832': ['physical-violence', 'death', 'suicide-bombing',
'conflict'],
    '1833': ['physical-violence', 'death', 'conflict'],
    '185': ['physical-violence', 'death', 'assassination',
'conflict'],
    '186': ['physical-violence', 'assassination', 'conflict'],
    '19': ['conflict'],
    '191': ['restrict-movement', 'conflict'],
    '192': ['occupation', 'conflict'],
    '193': ['conflict', 'armed-conflict'],
    '194': ['conflict', 'armed-conflict'],
    '201': ['conflict', 'mass-violence'],
    '202': ['conflict', 'mass-violence', 'death', 'physical-
violence'],
    '203': ['conflict', 'mass-violence', 'death', 'physical-violence',
'ethnic-violence']
  };

  var tagForEventCode = function(props) {
    _(props).each(function(prop) {
      if(!_(codeToTag[recordObject[prop]]).isUndefined()) {
        tags = tags.concat(codeToTag[recordObject[prop]]);
      }
    });
  };

  tagForEventCode(['EventCode', 'EventRootCode', 'EventBaseCode']);

  return _(_.uniq(tags)).map(function(tag) { return {name: tag}});
};


/**
* Each daily CSV contains information indexed on that day, not events
that
* (necessarily) occurred on that day. We're only interested in recent
information
* so we're removing old data, and records that are too generic.
*/
Gdelt.prototype.shouldTransform = function(recordObject, dateString) {
  //logger.info("Gdelt.shouldTransform checking against " +
dateString);
  if(!moment(recordObject.SQLDATE,
'YYYYMMDD').isSame(moment(dateString, 'YYYY-MM-DD'), 'day')) return
false;
  if(_s.startsWith(recordObject.EventRootCode, '0')) return false;
  if(parseInt(recordObject.EventRootCode,10) < 14) return false;

  var descriptions = _(this.eventDescriptions).filter(function(desc) {
    return desc.code.toString() ===
recordObject.EventCode.toString();
  });
  if(_(descriptions).isEmpty() || _s.include(descriptions[0].label,
'not specified')) return false;

  return true;
};


Gdelt.prototype.transform = function(record) {
  var that = this;

  var recordObject = that.recordToObject(record);

  var transformedRecordObject = {
    remoteID: recordObject.GLOBALEVENTID,
    publishedAt: new Date(moment(recordObject.SQLDATE, 'YYYYMMDD')),
    lifespan: "temporary",
    content: (function() {

      var descriptions =
_(that.eventDescriptions).filter(function(desc) {
        return desc.code.toString() ===
recordObject.EventCode.toString();
      });

      if(!_(descriptions).isEmpty()) {
        return descriptions[0].label;
      }
      else {
        return null;
      }
    })(),
    geo: {
      addressComponents: {
        formattedAddress: (function() {
          if(!_(recordObject.Actor1Geo_FullName).isEmpty()) {
            return recordObject.Actor1Geo_FullName;
          }

          if(!_(recordObject.Actor2Geo_FullName).isEmpty()) {
            return recordObject.Actor1Geo_FullName;
          }

          return null;
        })()
      }
    },
    language: {
      code: 'en'
    },
    source: 'gdelt',
    tags: []
  };

  // Add coordinates if available
  if(!_(recordObject.Actor2Geo_Lat).isEmpty() || !
_(recordObject.Actor1Geo_Lat).isEmpty()) {
    var point = (function() {
      if(!_(recordObject.Actor1Geo_Lat).isEmpty()) {
        return [recordObject.Actor1Geo_Long,
recordObject.Actor1Geo_Lat];
      }
      else {
        return [recordObject.Actor2Geo_Long,
recordObject.Actor2Geo_Lat];
      }
    })();

    transformedRecordObject.geo.coords = point;
  }

  // Add fromURL if available
  if(!_(recordObject.SOURCEURL).isEmpty() &&
validator.isURL(recordObject.SOURCEURL)) {
    transformedRecordObject.fromURL = recordObject.SOURCEURL;
  }

  // No need to summarize this content, it's already short
  transformedRecordObject.summary = transformedRecordObject.content;

  // Set the initial tags
  transformedRecordObject.tags = that.determineTags(recordObject);

  this.returnData(transformedRecordObject);
  return transformedRecordObject;
};

module.exports = Gdelt;
```

681 lines of code before you can make anything...

… or **six** lines of code.

```
var url = 'http://crisis.net/item',
location = ’36.821946,-1.292066';
$.getJSON(url + '?location=' + location, function(data) {
  console.log(data);
});
```

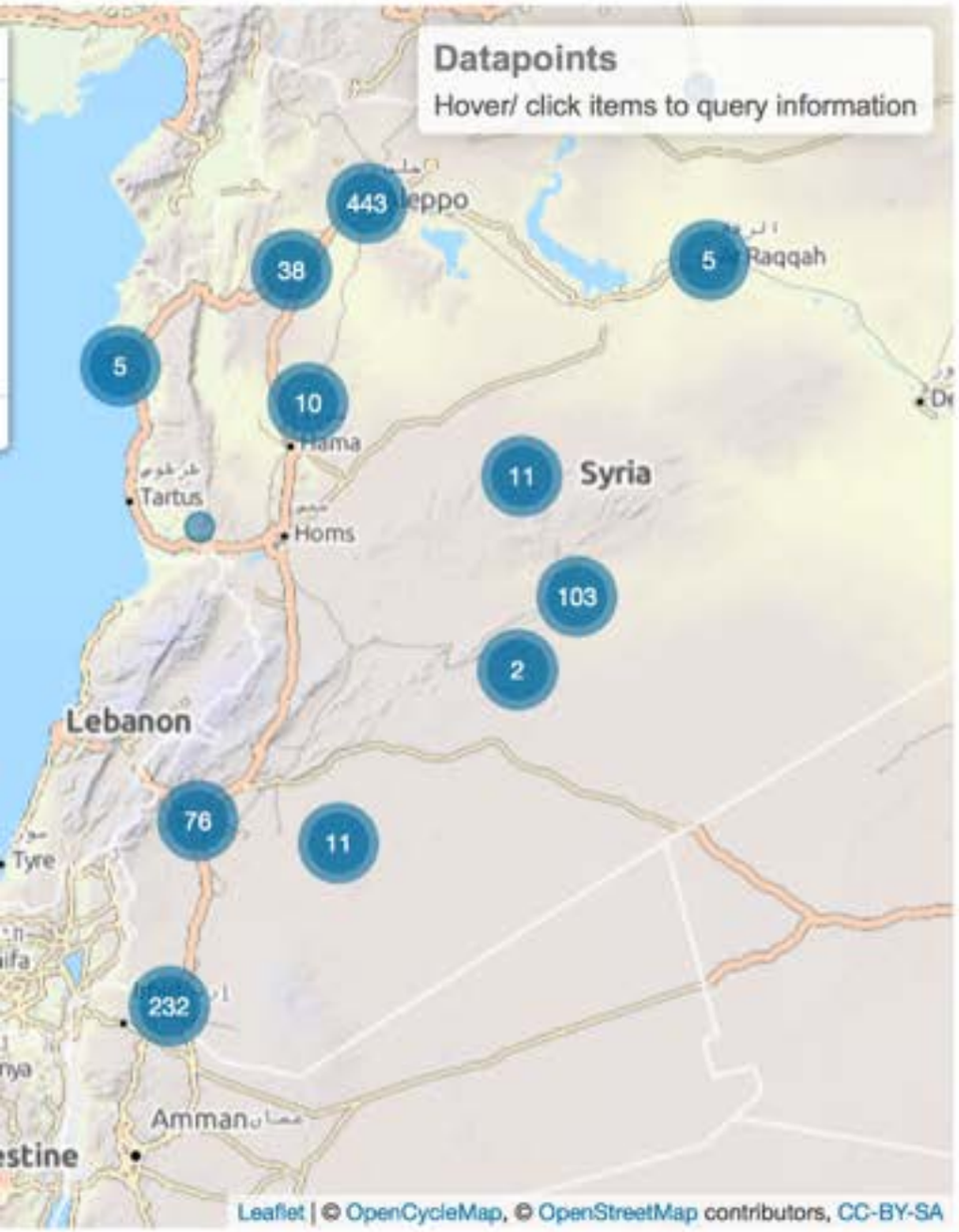
Syrian Civil War: An interactive timeline-mapping of 3000 social media reports



Filter by City

Filter by Attack

Reset Selection



Showing 1 to 10 of 3,000 entries

Search:

Date of Publication	Report	Author	Attached Image
August 01, 2014	**English translation**: [not available] **Original**: عاجل: مراسل حلب نيوز : سقوط برميل متفجر على جامع مالك بن انس في حي الهلك أسفر عن إصابة خمسة اشخاص أثناء خروجهم من الصلاة.	HalabNewsNetwork.HNN.2 (شبكة حلب نيوز H.N.N)	-
August 01, 2014	**English translation**: [not available] **Original**: عاجل:حلب: سقوط برميل متفجر على محيط منطقة البريج شمال شرق حلب.	syria.lattkia.m.l.l (مرج الحرية في ريف اللاذقية الإعلام الحرlattakia)	-
August 01, 2014	**English translation**: [not available] **Original**: درعـــــا #النـــخـــل المـــكـــتـــب الاعـــلامـــي الـــمـــوحـــد 1 -8- 2014 تصاعدة أعمدة الدخان جراء أستهداف مدينة إنخل برميل متفجر من قبل طيران الأسد المروحي	syria.news.F.N.N (شبكة فلاش سوريا | F.N.N)	
August 01, 2014	**English translation**: [not available] **Original**: ريف درعا الجيش الحر يفكك برميل لم ينفجر 1 8 2014	al.jazeera.hoooms (قناة الجزيرة الحمصية)	

Ebola Infection

311
Country Infections

431
Total Infections

24/9/2014

July 14 Reports from Macauley Street Satellite Hospital in Freetown, say they recently confirmed that a suspected Ebola who was admitted at the facility has escaped. Sources at the Macauley Street Satellite Hospital claim that a nurse known as Basiru from Daru arrived at the hospital claiming he had come to Freetown to sit to a public examination last week Saturday. However, by Monday this week... read more

- osm

Select country for statistics

- Guinea
- Liberia
- Sierra Leone

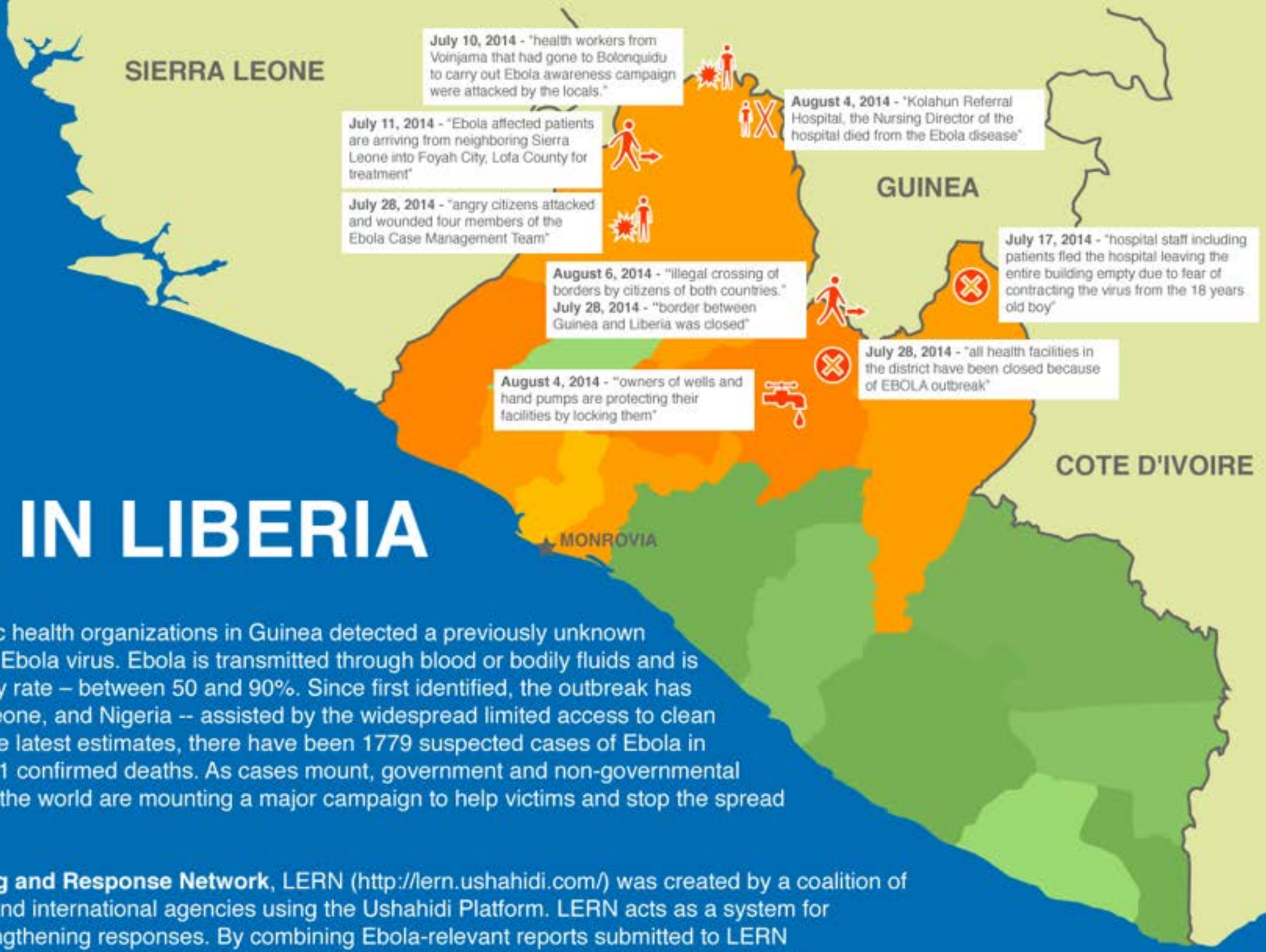
EBOLA IN LIBERIA

August 10, 2014

In March of this year, public health organizations in Guinea detected a previously unknown outbreak of the dangerous Ebola virus. Ebola is transmitted through blood or bodily fluids and is notable for its high mortality rate – between 50 and 90%. Since first identified, the outbreak has spread to Liberia, Sierra Leone, and Nigeria -- assisted by the widespread limited access to clean water and sanitation. By the latest estimates, there have been 1779 suspected cases of Ebola in West Africa, resulting in 961 confirmed deaths. As cases mount, government and non-governmental organizations from around the world are mounting a major campaign to help victims and stop the spread of the virus.

The **Liberia Early-Warning and Response Network**, LERN (http://lern.ushahidi.com/) was created by a coalition of civil society, government, and international agencies using the Ushahidi Platform. LERN acts as a system for identifying issues and strengthening responses. By combining Ebola-relevant reports submitted to LERN (downloaded through CrisisNET API) with CDC data, rapid visualization of the outbreak in Liberia is possible.

If you wish to help fight the Ebola outbreak in West Africa, consider donating to organizations like Médecins Sans Frontières (http://www.doctorswithoutborders.org) or Samaritan's Purse (http://www.samaritanspurse.org/).



Areas with confirmed or probable cases of Ebola (as of Aug 7, 2014)

Report from Liberia's Early Warning and Response Network

powered by CRISIS NET

Map credit: Wikipedia. Icon credit: UNOCHA. Report data: Liberia's Early-Warning and Response Network. Ebola case data: CDC

Problem:
difficult to stay connected
during an emergency

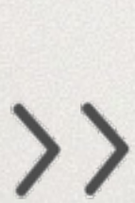
Ping

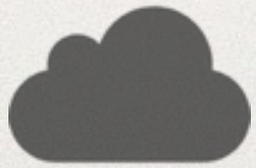
Ping

A group Check-in Tool for Emergencies

ping.Ushahidi.com

(accessible via mobile or desktop)

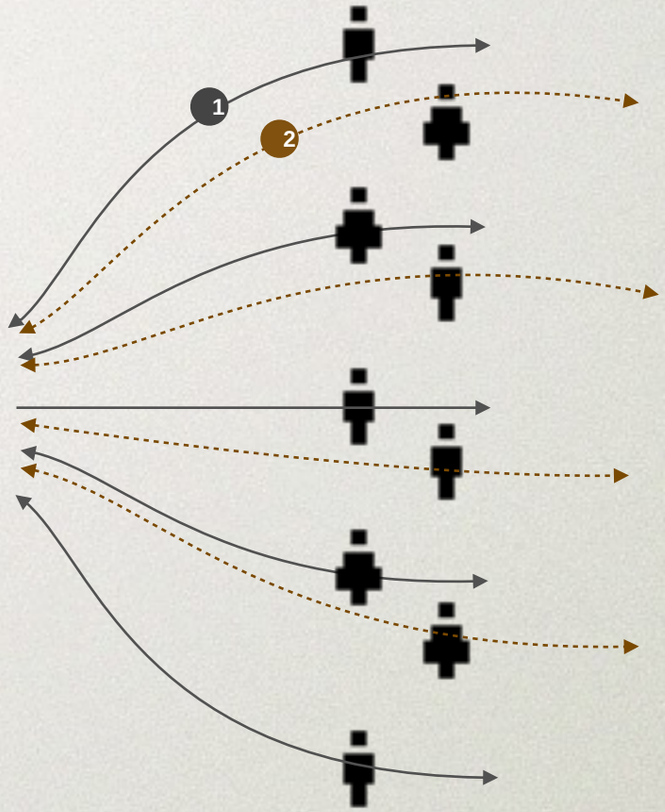




Send Message:

“are you ok?”

as a text message & email



1 family or team member

2 Next contact (spouse, roommate, friend)

Why do we use technology designed for **London** and **Los Angeles** when we live operate in **Nairobi** and **Delhi**?

http://www.flickr.com/photos/qilin/2262821682

Our reality

- ✖ Unreliable Power
- ✖ Unreliable Connectivity
- ✖ Challenging environments

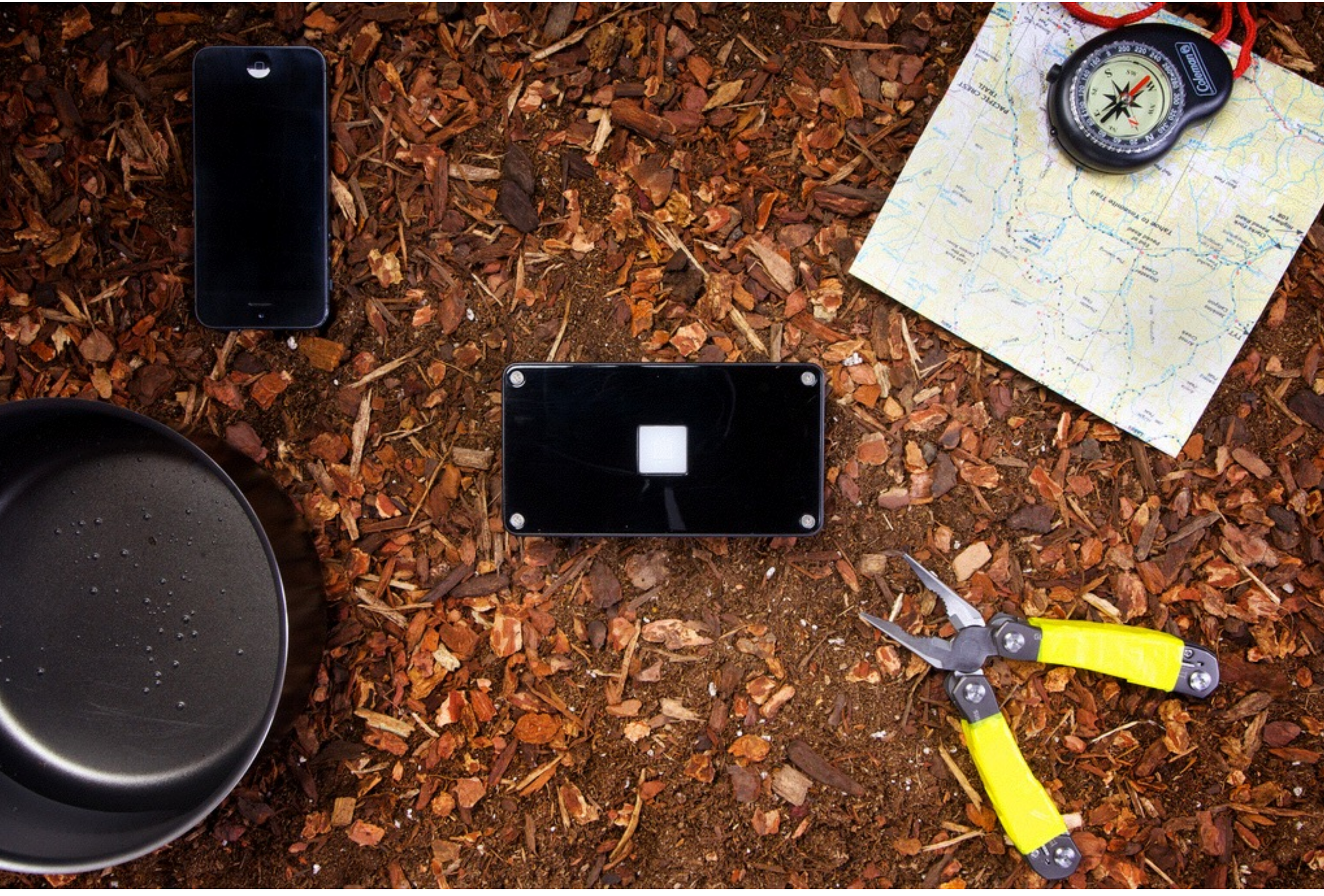
“Our problem is internet.”

- Headmaster, Kawangware Primary School

There were **411,265M** African children aged 14 years or below in 2010.

By 2050 there will be **838,717M** .

Source: UN


Columbia
PACIFIC CREST TRAIL

BRCK

the backup generator for the internet

two roles, One BRCK

Connecting people

to eduction, social, and communications

Connecting sensors

and information to responders, technicians, and analyst

**photo credit Charity Water*

The longterm objective...

Final Mile Connectivity

TOTAL Connectivity

needs of **375K** Rural

**photo credit Worldreader*
Africa Media Online
Getty Images Clinton

IHUB PHOTOGRAPHY
BRCK
A mobile router, with onboard power and storage.

IHUB

iHub Cluster
iHub UX
m:Lab

USHAHIDI

Ushahidi
Crowdmap
SwiftRiver
SMS Sync

CRISISMAPPERS

ICCM
Standby Task Force

HARDWARE & MAKER

BRCK
Maker Faire Africa
Gearbox
Afrigadget.com

OTHER

Making All Voices Count
Savannah Fund
Akirachix

You are 007.
We are Q.