GlobaLeaks: towards a more Open and Transparent Society

Project Plan

GlobaLeaks Project
Hermes Association, non-profit registered in italy
http://www.globaleaks.org

Contents

Goal

Goal of this document is to provide a detailed vision of the GlobaLeaks Project plan and deliverables.

Intro

GlobaLeaks is a project aimed at supporting the practice of whistleblowing by giving people the software tools necessary to start their own initiative. With GlobaLeaks even non-techies will be able to setup their own anonymous whistleblowing site.

We wish to promote transparency and the adoption of more open practices as fundamental values for spotting malpractice throughout society. Obviously this goes beyond just making a piece of software, that is why we try to keep an open eye to the world of whistleblowing as a whole.

Whistleblowing websites - running on globaleaks - require a number of corollary technologies in order to reach their full potential. These technologies will have their own independent communities which do not need to be directly tied to GlobaLeaks, but their growth will certainly influence it. Tor2web, LeakDirectory and the Tor PrivacyBadge are an example of this.

GlobaLeaks will provide high levels of security and anonymity by default, though these parameters can be flexibly changed. This allows the operator to tweak the system in order to reach the right security/usability tradeoff that he feels comfortable with.

Basically, GlobaLeaks aims at becoming the de-facto standard in technologically-powered whistleblowing.

Overview

The GlobaLeaks project is composed of several sub-projects, each required in order to properly implement the project goals. These are necessary to ensure a high degree of privacy, distribution of responsibilities, ease of use and operations of a GlobaLeaks site.
Below is a summary of all the sub-projects required to fulfill the end goals of the Project:

GlobaLeaks Software (GL)
A software which implements a whistleblowing site able to handle submissions from web, desktop and mobile. It is designed to provide the best user experience over Tor Hidden Services, ease of setup and operation.

Anonymous Web Application Framework (AWAF)
A framework providing a desktop application with a built-in python web server for anonymous inbound and outbound communications over Tor (shipped with the software).

Tor2web (T2W)
A software exposing a Tor hidden service as an internet-reachable website. Built to setup a network of distributed servers and multiple networks.

LeakDirectory (LD)
A site (and software) providing an index of Whistleblowing sites, with their capabilities, goals, references and information. A GlobaLeaks site may register itself to LD, and users will review and comment on its page evaluating its effectiveness and value. It's the AppStore of whistleblowing sites.

PrivacyBadge (PB)
A server software implementing a Web widget which indicates the users' anonymity status (if they are using Tor or not).

GlobaLeaks Mobile (GLMobile)
A mobile client software enabling devices to submit content (Audio, Video, Photo, Text); this requires a research about the best portable way to ensure compatibility between mobile platforms (to avoid different programming branches for Symbian, iPhone, BlackBerry devices).

Fax2social (F2S)
A software daemon which will handle incoming FAX documents, OCR them, anonymize them and submit them to a third party application (GlobaLeaks, Twitter, Dropbox).

(this page has been intentionally left blank)

Software Projects

These sections provide an architectural and technical overview of all GlobaLeaks software components, indicating relevant elements, technological choices and their current development status.

An updated version (ongoing) of the status of current projects and their description is available at http://wiki.globaleaks.org .

GlobaLeaks

GlobaLeaks is the core component and the main reason why the project was born.

GlobaLeaks software allows the node administrator to easily setup an anonymous whistleblowing site on any operating system. Since the site is running as a Tor Hidden Service there is no need for a static IP address, so the *node admin* can potentially run a site off of his home DSL connection on his desktop computer.

A *Whistleblower* can submit information through the submission system and is allowed to interact with the receivers of his submission by commenting or uploading new data.

The software architecture (Work in Progress) is technically described in General Architecture.

In 2011 a big set of ideas on globaleaks features and evolution came out during the 0.1 development phase. Those are represented here: https://github.com/globaleaks/globaleaks/issues
Howto's for GL 0.1 installation can be found here: https://github.com/globaleaks/globaleaks/wiki
CCC 28C3 workshop slides on Social Hacking with GlobaLeaks

The new general architecture of the software, focusing at first on modularity and scalability to accommodate all future software features, is composed of 2 main elements:
GLClient and GLBackend.

GlobaLeaks Client

The GlobaLeaks Web Client (GLClient) is a modern JavaScript application that allows whistleblowers, node administrators and receivers to interact with the GlobaLeaks Server.

Its main design goal is to provide a complete separation between the client-logic and the server-logic, this will lead to the following improvements on the previous model:
- Responsiveness over high latency networks (Tor Hidden Services[1])
- Ability to be verified and run locally on the clients' computer (this makes it possible to do secure JavaScript cryptography[2])
- Sharing of the client side code with Mobile version of the application (PhoneGap[3])

The GlobaLeaks server will run as a Tor Hidden Service for security and privacy reasons. Unfortunately one of the drawbacks of Tor Hidden Services is that they have a high latency. For this reason all of the application logic will come bundled into the GLClient that can be downloaded over a different channel, verified and then loaded into the browser (as a plugin, or opened locally). The client at this point will only do asynchronous requests to the backend via Tor and be able to display feedback to the user on the status of the requests.

This is a quite innovative approach and it completely avoids the so called "white page effect", where the user is presented with a blank page while he is waiting for the content to be loaded.

Another advantage of having the web application be locally run, as opposed to have it served from the server, is that it's not possible for an attacker to manipulate the DOM of the page. This means that it is possible to execute cryptographic functions in an environment that is "separated" from the hostile web.

The same code base can be reused to make an application for mobile devices. All that needs to be changed is the user interface, but none of the underlying logic need to be changed. The application can then be built using tools such as PhoneGap.

[1] https://www.torproject.org/docs/hidden-services.html.en

[2] http://hellais.wordpress.com/2011/12/27/how-to-improve-javascript-cryptography/

[3] http://phonegap.com/

GlobaLeaks Backend

The GlobaLeaks Backend Server (GLBackend) is the piece of software installed by the node administrator and responsible for handling the reception of submissions from whistleblowers and the notification to the targets.

The backend will only expose a REST interface. The client will be responsible for the UI rendering and will talk to the backend by exchanging json objects.
The core components of the backend are: Submission, Storage, Status Page, Notification and Delivery, Administration.

The Submission component deals with how the information is acquired from the whistleblower. The backend advertise a set of fields that it accept as input and the GLClient will render them in UI.
It is possible to highly customize the kind of submission fields and, to a certain degree, how they are presented to the user. To get an idea on how configurable the submission fields and the submission wizard should be it's possible to have a look at WhistleblowerSecurity and Alertline commercial Whistleblowing services.

The Storage mechanism is responsible for storing the material that is uploaded by the whistleblower.
It should be highly abstract and possibly replaceable with any other storage mechanism (for example: database, local filesystem, Tahoe-lafs, dropbox, remote ftp, encrypted container, etc.)

The Status page deals with how the user interacts with the receivers.
The user authenticates with such a page thanks to his submission receipt which is given to him after a successful Submission. From this page he is able read the comments the receivers write and send his own. He is also able to upload new material.
Every interaction with the status page is append-only (no content is ever modified, just added).
The Whistleblower is given real-time statistics on access/download of material done by the receivers. The status page have properties such as time of expiration, download limits and self-destruction to provide auto-clean-up of submissions and is going to reduce the attack surface.
A Preference page is provided to let the receivers make actions, like changing their access password, configure the encryption (PGP keys) or delete the whole submission if authorized.

The Notification system deals with how the availability of new submission data get notified to all members of the receiver list.
Multiple notification systems will be supported in the future, to give receivers the maximum flexibility in how they know about new submission data availability.
By default notifications will be delivered via email.

The Delivery system deals with how the submission data (form/checkbox data, files, comments) are delivered to members of the receivers list.
Multiple delivery systems will be supported in the future, to give receivers the maximum flexibility in how they receive the information and how it's reception integrates into their workflow. Integration with backoffice tools for Fact Checking (like DocumentCloud, Ahref's Timu) and Case Management (like OTRS, ZenDesk) gives the organization setting up a

Whistleblowing initiative a great flexibility in handling the submitted data and managing **fact-checking process**.
By default Delivery is done through the Status page, via a web interface trough randomly per target/submission, with a uniquely generated url.

The Administration component deals with how the node administrator sets up the node and selects the list of recipients, their notification/delivery methods and authentication/ authorization schema. It will also provide some basic statistics of the nodes usage and operations and a debugging/error reporting facility.

To facilitate the setup of a GlobaLeaks node, **GLBackend will be built and bundled using APAF**, another software component of the globaleaks project: The Anonymous Python Application Framework (**APAF**).

Anonymous Python Application Framework (APAF)

The goal of the Anonymous Python Application Framework (APAF) is to provide a container which will allow anybody to build their web application in such a way that it can automatically publish itself to the Tor network as a Tor Hidden Service.

The framework is designed to facilitate the creation of a Python Tornado-based Web Application and deliver it as a Desktop Application (**Program.exe / Program.app**) or as a *nix package. This drastically reduces the complexity of running a server anonymously even from a home PC.

Although we are creating it specifically for GlobaLeaks this could be applied to any web application. Think for example of a blog platform that can run directly from the user's home computer. The creator of the web application just needs to package it with APAF and the user can download the application which will bundle together all the required dependencies and start a web server on his home computer.

Since GlobaLeaks's aim is to reach even the non-technically proficient users and enable them too to run a whistleblowing initiative. Having a Desktop application that can be run by simply clicking on an executable drastically decreases the entrance barrier.
Although GlobaLeaks will be packaged as a Desktop Application for Windows and Mac OS X, we will still keep shipping, and recommending, the *nix package to be run on a proper server.

The dependencies that will be packaged thanks to APAF are GLClient, GLBackend, the Python interpreter, the required python modules and Tor. The output will be an executable that includes security features such as sandboxing. The goal is to guarantee the highest level of security and privacy possible, with the least effort on the user's part.

Hopefully APAF will also be used by other developers that are interested in making their own web application while leveraging the power of Tor Hidden Services (and also Tor2web), but they don't want to go through the hassle of designing up their own build environment.

APAF is going to be developed as part of the Google Summer of Code with the Tor Project. For more details on it check out: http://archives.seul.org/or/dev/Mar-2012/msg00088.html.

Tor2web

Tor Hidden Services allow people to publish content anonymously and without the necessity of having a static IP address assigned. Usually these services are only reachable from the Tor network, meaning that all visitors must install a Tor client.

Tor2web creates a tradeoff between the client's anonymity and usability, allowing regular internet users to visit all the Tor Hidden Service's websites.

A normal Hidden Service address would be like http://eqt5g4fuenphqinx.onion/ . To visit the same site from a regular browser, thanks to tor2web, the user just needs to visit https://eqt5g4fuenphqinx.tor2web.org.
This happens because a network of volunteers run a software that proxies all the requests via the Tor network.

The functionality of tor2web is fundamental for GlobaLeaks, since it allows a node administrator to immediately expose to the internet his site that is running as a Tor Hidden Service. This means that they can have the equivalent of a regular web site without having to register a domain name or pay for a static IP address, while being completely anonymous.

The current version of tor2web has faced and faces takedowns, every day. For this reason we are working on a new design of tor2web that aims at further distributing the responsibility of running a node.

It allows, for example, to run tor2web software on a domain name that is different than *.tor2web.org. This means that we do not need to trust that person with our private SSL certificate and it makes the infrastructure more resilient to DNS based censorship.

With the new version, every tor2web visitor is provided with a unique link that is valid only for him and for his session. This link may point to a different domain name (i.e. not tor2web.org). The benefit of this scheme is that there is a separation between the provider of content and the link to the content itself. It also means that illicit content may not be directly linked since the link expires after a certain amount of time.

Tor2web 3.0 will run as an AWAF application for easier distribution also among osx/windows users.
CCC 28C3 workshop slides https://github.com/globaleaks/advocacy/blob/master/Tor2web-Workshop-28C3.pdf?raw=true

LeakDirectory

A goal of GlobaLeaks is to reach a big number of whistleblowing sites that are dedicated to certain topics and certain geographical areas. As a whistleblower it might be difficult to choose which one is the best to use, or which node accepts the information that he wishes to speak out about.

LeakDirectory will fill this gap by creating an archive of all existing GlobaLeaks sites and whistleblowing related material. When a node administrator sets up a GlobaLeaks site he can choose to have it published to LeakDirectory.

From there the users will be able to comment on it and review it. The profiles of the sites will include the impact that the site has had (i.e. what submissions have led to something).

LeakDirectory will allow whistleblowers to make an informed decision as to what is the best recipient of the information they have. It will be the App Store of Whistleblowing sites.

LeakDirectory may also show statistics on whistleblowing site operations and submission form inquiry details from registered whistleblowing sites, given that they decided to publish those data. LeakDirectory will provide as OpenData all its database and information to stimulate further analysis on the whistleblowing environment.

LeakDirectory represent also a security asset in the whistleblowing environment because, in a distributed network, the amount of unreliable (or even rogue) sites can be a damaging effect in the sites' network. A user driven collaborative filter, like LD, can be the best way to select a trustable whistleblowing site.

PrivacyBadge (Torcheck)

To verify that the user has properly configured his browser and that he is in fact anonymous we need to develop a tool that verifies if the user is connecting to the server via Tor.

Since Tor Hidden Services receive the traffic directly from Tor they are not able to check if the client is coming from Tor or not, since he may be using tor2web. For this reason it is necessary to have a JavaScript-based badge that fetches a JSONP object from a web server that runs the TorCheck software. This object will contain True if the users IP address is amongst those of Tor Exit Nodes, False otherwise.

This is very useful for GlobaLeaks since it allows us to warn the user that he is connecting to the site in a non-anonymous way, but it is also of value to other websites.
The GLClient makes use of such service to let the user be aware of his anonymity status. The same service should be run by the organization running a whistleblowing initiative on their website, even before the submission happen.
Also anonymity advocates could place such a badge on their website and educate its visitors of the importance of using tools such as Tor to avoid network interception.

GlobaLeaks Mobile

To fully enable citizens to report on malpractice it is important that mobile tools are available. The GlobaLeaks client software is written in JavaScript/CSS. This code can be bundled in a way that the phone's browser loads it locally.

The Administrator of a GlobaLeaks site should be able to build such a mobile application directly from his administration panel. As output he should get the Android, iPhone and Symbian apps, which he can then upload to the market.

This means that a non-technically skilled node administrator can have a whole package of software to rally people into supporting his cause through whistleblowing.

The build process will be carried on by PhoneGap Build http://build.phonegap.com .

A set of native applications are also needed to include support for extended privacy features (Tor, obscuracam).

Fax2social

Fax2social started as an idea as an anti internet-kill-switch tool (drafted on http://fax2social.wordpress.com) to handle incoming FAX, OCR them, anonymize them and submit to a third party application (GlobaLeaks, Twitter, Dropbox).
We realized that the ability to easily setup a FAX dropbox for use as a Submission system for globaleaks is highly important, especially for countries with no wide internet availability. This is going be deployed as a separate daemon (also due to its requirement for Win32 Professional OCR software).

The project can can be found on https://github.com/globaleaks/fax2social .
With the help of a telephone provider, we are able to supply a wide range of phone numbers, covering the local numbers of almost every country in the world. This is one of the most effective "anti Internet-kill-switch" technologies because, even in the event of a Internet shutdown, the fax can still work. With a fax, you could reach a social network or a whistleblowing site directly.

Project Priority

This section provide an overview of the Project Status (at the date of writing 11.02.2012), Priority for the project and Effort required to complete various deliverables.

Below a brief table, followed by additional notes on the status.

Project	Status	Priority
GLClient 0.2	Early prototype	Critical
GLBackend 0.2	Early prototype	Critical
APAF	Early prototype	High
Tor2web 3.0	Early prototype	High
LeakDirectory	Require GL 0.2	Medium
PrivacyBadge	Working prototype	Medium
GlobaLeaks Mobile	Advanced Android prototype 0.1,Require GL 0.2	Low
Fax2social	Advanced prototype	Low
GlobaLeaks Software 0.1	Complete, feature-freeze	n/a
Tor2web 2.0	Complete, feature-freeze	n/a

Project Status

Below a brief description of each software project's status.

GlobaLeaks Software (GL)
GlobaLeaks 0.1 has been deployed. This version's features will be frozen in order to work on the new design.
GlobaLeaks 0.2 is proceeding by prototyping and specifying GLClient, GLBackend and APAF.

GLClient
Some initial experiments have been carried on to integrate Javascript client technologies required for a compact, single JavaScript client application in a modular way.
First experiments are on https://github.com/globaleaks/GLClient .

GLBackend
General Architecture specifications of the GLBackend and REST interfaces are available on https://github.com/globaleaks/GLBackend

Anonymous Python Application Framework (APAF)
Currently APAF handles Tor start/stop. Python tornadoweb tor facilities and prototyping work is going on by testing Pyinstaller to deliver win32/osx self-contained applications.
The working code is at https://github.com/globaleaks/apaf but it will migrated to a dedicated repository.

Tor2web (T2W)
The Tor2web 2.0 network is running on two servers and has some serious distribution scalability issues.Tor2web 3.0 is a basic working prototype, research has been done, a formal specification has to be written. More info http://wiki.tor2web.org .

LeakDirectory (LD)
Currently leakdirectory.org is a community-editable wiki that contains the list of all whistleblowing sites along with useful links and information on the topic of whistleblowing. The future plan is to convert leakdirectory into a fully-fledged web application to review and discuss whistleblowing sites.

PrivacyBadge (PB)
Torcheck working prototype is available as PHP code on https://github.com/globaleaks/TorCheck

GlobaLeaks Mobile (GLMobile)
Technological analysis for building on the base of PhoneGap has been done.
An Android GlobaLeaks Application (GLDroid) prototype with integration with Tor and Obscuracam has been made. It will be put in production with GlobaLeaks 0.2 thanks to the REST framework.

Fax2social (F2S)
An advanced prototype has been made, early testing is required. A very good formal paper on the project in a semi-academic way is on-going.

Implementation Plan

This is a very brief Project Plan that describes which are the main activities and goals to be carried on for each project.

Effort estimation

This is an effort estimation to build up the GlobaLeaks project up to its full realization.
The total effort to fully implement the overall project is forecasted to be 1292 days of activity (a couple of solar years at least).

Project management is estimated to account for 10% of the overall time.
Contingency is estimated at 30%, but at the alpha release a re-evaluation of estimates will be done, increasing or decreasing contingency on the basis of the experience acquired.

Project	Effort man working days
GlobaLeaks	152.6
GLClient 0.2	210
GLBackend 0.2	224
APAF	161
Tor2web 3.0	196
LeakDirectory	119
PrivacyBadge	27
GlobaLeaks Mobile	150
Fax2social	53

The effort estimation comes from more detailed evaluation based on the current status of the project and various intermediate deliverables. A detailed spreadsheet with estimates is also available.

Project Requirements

This section of the document defines all the project requirements that are not directly related to software coding, but that represent output and activities to be planned in Project Plan for each software project.
Being an Open Source software project, we need to make them alive within the opensource community and this requires more than just making a software work.
We'll consider this activities for each software project.

Release requirements

Each software will undergo a process of:
- Alpha

This is the first working prototype release of the software. It should be tested to work on a certain setup and it should be verifiable by another developer. The code does not need to be particularly efficient or clean, though it should reflect the design intentions.
A hacker should be able to get the software up and running with minor hacks.

- Beta

This is when the software is moderately tested, but not yet stable enough to be released to the general public. The code needs to be clean and readable by third party developers. A hacker should be able to get the software running without hacks.

- Stable

This is when the software has reached a fair level of maturity. It has been tested on different platforms and verified to work properly. The code should be clean and well documented. The software should come with a guide on how to set the software up even for non-hackers or non-developers.

- Release

This is when the software has been packaged and tested for quality on multiple different systems. This is the final release of the software that will reach the end user.

Software specification requirements

Each project must have the following specifications:
- General/Software Architecture
- User interface Mockup
- Interface API (API to use the software)
- Security Requirements

Documentation requirements

Each software project must have:
- Development documentation: How to build.
- Setup and Operations document: How to install, operational requirements, how to maintain, description of configuration and options.

Quality Assurance Requirement

Each software must be quality tested during and after the development.
This means that the software development must try to make extensive use of unit-testing and that, before the different intermediate releases, there must be a quality assurance testing activity.
For each software document we must:
- Document the unit testing strategy
- Provide a feature testing document to be executed and tested independently
- Execute the test with success, opening defect/improvements issues, at each run

Graphic/Website requirements

All the software project must have an appealing logo and graphics elements.
Each project must have a simple but effective website (or part of a website) to communicate its project goals to the different target users (users, volunteers).

Internationalization requirements

All the software must provide support for internationalization.
The localization files must be able to get imported/exported in a translation tool.
A collaboration support tool must be setup to handle collaborative contribution for the localization to various languages.
Documentation on how to handle the translation, with an invite for translation must be found.

Community Requirements

Each project must have:
- Open source code repository and collaborative tools (github)
- Always provide up-to-date information about the next steps and how to get involved
- A mailing list with a public archive
- A way to collect funds to be distributed to the community