GSOC
PROPOSAL

FOR

AMBER
(IPFS friendly distributed version of Amber)

Berkman Center for Internet and Society

Shivam Khandelwal

TABLE OF CONTENTS

Contact Information

Name	Shivam Khandelwal
Country	India
School & Degree	International Institute of Information Technology B.Tech. + MS (dual degree)
Email	skbly7@gmail.com shivam.khandelwal@research.iiit.ac.in
Phone	+91-945-77777-88
Preferred communication (higher to lower)	Chat, Email, Video Conference

Projects interested

I am interested in 2 projects by Berkman organization.

1. **Teem:** Reputation-based or gratitude-based immaterial rewards
2. **Amber:** IPFS friendly distributed version of Amber

This proposal is for Amber project.

About Me

Resume

http://shivamkhandelwal.in/resume.php

Brief Intro

This is a great pleasure to introduce myself to all of you. I am **Shivam Khandelwal** from International Institute of Information Technology, Hyderabad, India. I am dual degree student (B.Tech. + M.S. by Research), and working with Software Engineering Research Center (IIIT-Hyderabad) on Gamification domain.

I have been working for more than 3-4 years on web based development, but lack prior open source experience. I have a good understanding of software engineering related methodologies, tools and usages.

As a person, I am enthusiastic about promoting competitive programming culture in and outside college, and love new challenges. Due to same reason, I have been working with CodeChef since 1.5 years, and was even part of organizing team of ACM ICPC Kolkata and Chennai regionals (and ACM Amritapuri online support) this year.

Social profiles

IRC : skbly7
GitHub : https://github.com/skbly7
Twitter : https://twitter.com/skbly7
Blog : http://shivamkhandelwal.in/
LinkedIn : https://www.linkedin.com/in/skbly7

Experience

- ➢ PHP
  - Wordpress
    - Freelancer from May 2013 - Dec 2014 on various wordpress websites.
    - Good understanding of wordpress's source code & plugins.
    - Custom plugins written, none on production as per now.
  - Drupal
    - Working with CodeChef since Dec. 14 on it's Drupal codebase.
    - Custom modules created and deployed on production.
  - PHP based dynamic web crawler, capable to crawl websites with depth limit (& different priority queues) on basis of DOM path configured. (2014)
  - Took various workshops on PHP and crawling. (2013-14)
  - Experience with Slim3, CodeIgniter, etc frameworks.
- ➢ Python
  - Framework knowledge of Flask, Django, etc, with various applications developed using it.
    - Gamified code review platform
    - Worked with Codebase of Django based OSQA (~1.5 year) deployed here.
  - nltk, scikit-learn, etc for Machine learning purposes. (code on GitHub)
- ➢ Networking and Systems
  - Working with twemproxy (nutcracker), haproxy, squid3, polipo, etc as Software Engineer Intern with CodeChef.
  - Handles official courses portal (Moodle) as administrator for college.

The experience shared here are only which are related to Amber. Resume & LinkedIn profile have been added for complete list.

Why GSoC with Berkman?

I have been always passionate about projects which are related to web or networking, which is surely the main motivation for me to work with Berkman, and to write this proposal. Though there are many other organizations too in GSoC which also provide opportunity to work on similar fields, the possible research exposure and working experience with Berkman, is what which makes Berkman special to me. I am hoping to demonstrate potential by working as a remote fellow researcher, instead of just a traditional software engineer/developer during this GSoC timeline.

The reasons to select Amber project are as follows:

1. Majority of my works have been in wordpress and drupal, and I particularly loved the concept of Amber. (faced issue being solved by Amber myself in past)
2. Interest due to previous experience with Memcached, nutcracker (twemproxy) and other strategies, in Summer 2015 for handling traffic spikes.
   i.e. related to pipelining and sharding cache architecture to facilitate horizontal scaling.

Regarding GSoC Timeline and Timezone

These are clashes of GSoC timeline with my college's academic calendar.

Time Period	Phase	Reason	Time Commitment
22 April - 28 April	Community Bonding Period	College exams	2 hrs/day
~1 Aug - 23 Aug	Work Period	College reopens	No effect on time committed

AFAIK there isn't anything scheduled as per now. But, if there are any other future clashed with the timeline, it will be informed timely, and it will be made sure that committed working hours to organization are not disturbed.

Committed time (EST) : 09.30AM - 05.30PM (flexible as per organization need)

Along with that, I would be working on weekends on above time, or as per the need of the organization & mentors, and to cover up any project time lost during clashes described above.

Proposal

Title

IPFS friendly distributed version of Amber

Overview

Amber automatically preserves a snapshot of every page linked to on a website, giving visitors a fallback option if links become inaccessible. The main aim of this project is to come up with distributed and resilient version of Amber, which can help it in achieving the real vision of 404/broken-link free internet. The project can be divided into two different parts which involves IPFS integration with existing Amber codebase, and to come with distributed Amber design and implementation.

IPFS Integration

IPFS is the new P2P hypermedia protocol made to achieve "The permanent web" by Protocol Labs. It can be seen as a single BitTorrent swarm, exchanging objects within one Git repository. It will be added into Amber's backend options by using official PHP IPFS API implementation.

Distributed Amber

This section of project deals with design and implementation of a distributed version of Amber which enable Amber nodes to communicate and share cached contents with each other (P2P cache sharing). The proposed design (solution) can be further divided into:

- Custom memento timegate (Central Server)
  Allow users to query P2P Amber nodes and facilitate P2P cache sharing.
- Amber nodes
  Addition of APIs for sharing caches with peers.
- Security Measurements
  misuse of the system is tried to be prevented by using Public-key cryptography and Reputation based cache resolving.
- Dashboard on Central server
  To let amber nodes registered, and get their public and private key pairs, needed for further server and P2P communication.

Technologies involved:
Python (memento), PHP (amber), Networking concepts, Basic cryptography

Detailed solution

The project can be broken down into following tasks and subtasks, which are described further:

1. **Amber**
   a. Understanding existing codebase
2. **IPFS**
   a. IPFS understanding and implementation
   b. Integration with Amber
3. **Distributed Amber**
   a. Discover possibility if memento timegate code can be directly used with few/none modifications.
   b. Creating web server based with/without Memento based on Step(a). Other features of this web server would be:
      i. Registration APIs addition
      ii. Distributed Amber nodes communication
   c. Security
      i. Reputation based cache discovery
      ii. Communication Protocol/handshake
      iii. Further research and implementation

Amber

Reading and understanding the current implementation of the Amber's wordpress plugin.

- Installation and using Amber.
- Understanding the current implementation and functionality
  - Amber common.
  - Amber wordpres.

The amber wordpress module's code base has been completely read and understood. The study of Amber common is still being going on.

IPFS

The IPFS understanding and implementation will be treated as black box during the implementation in this project. Though still some basic knowledge was needed, which has

been acquired already through online reading material available. Also, have gone through some of basic FAQs and explanation of IPFS using this .git repo.
As the IPFS protocol is in development phase, and comparatively much more prone to security and other updates, the official library's implementation and code would not be changed (so it can be easily updated with new versions, will no/minimum efforts).

Library: https://github.com/cloutier/php-ipfs-api
All the basic commands needed to get/put files data on IPFS is provided through above. Also, the above code has been read and found to be extensible enough to add more API if needed which are available on https://ipfs.io/docs/commands/.

The currently identified changes are as follows:

1. Wrapper over php-ipfs-api, to make it compatible to interact with iAmberStorage and iAmberFetcher interface.
2. get_storage_instance function
   New condition for AMBER_BACKEND_IPFS to be added, which will initialize above PHP IPFS library.
3. Amber-dashboard.php and amber-settings.php for adding new options related to IPFS.
    a. Addition into multiple select dropdown.
    b. Text box to add desirable IPFS endpoint.
4. Test cases

Everything would be kept same, and amber_cache table will be used only, with new provider value for IPFS and provider_id equal to IPFS hash.

Memento

The codebase for memento timegate which will be initially looked into is from this GitHub repo. The main step would be checking how existing implementation can be used directly, or as a wrapper in further step.
This step would need some more code reading and understanding Memento for detailed steps for conversion.

Distributed Amber

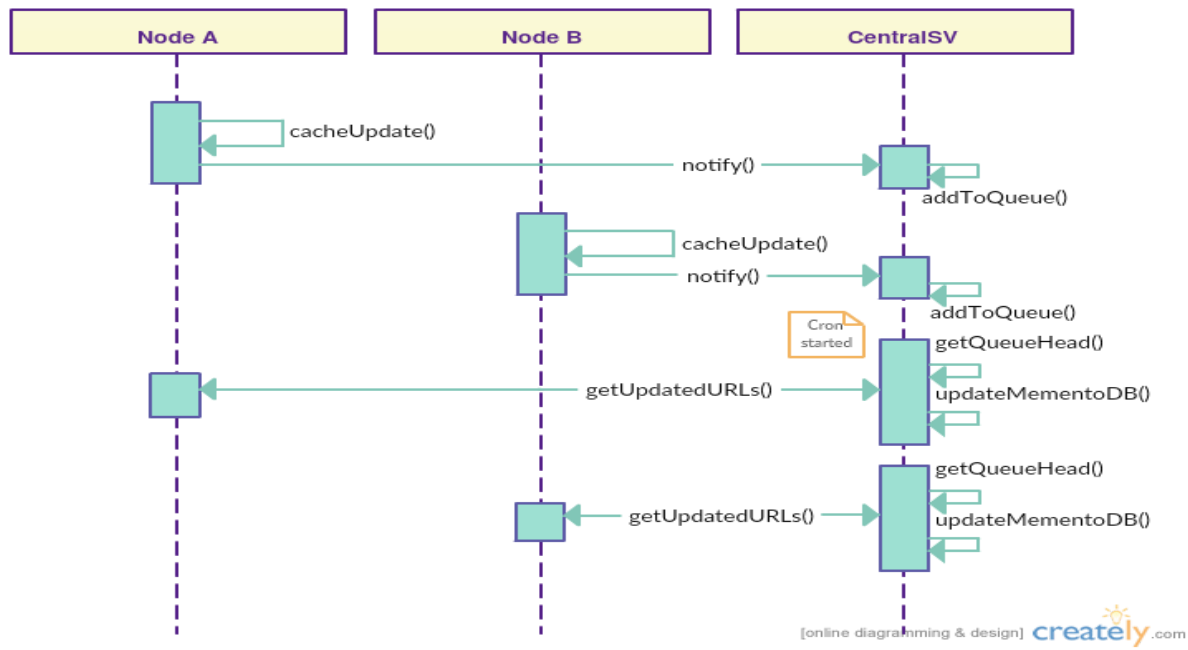
Memento Server communication
The meaning of terms used in this section, are as follows:

Amber nodes	All the amber users.
Central server	Memento based central server hosted by Berkman

1. Amber nodes
    a. The ambers nodes would be in active/inactive as per user's opt-in, and option for the same would be in Amber Dashboard.
    b. Whenever the caches data i.e. amber_cache is changed, the amber nodes will notify about it to the central server.

    c. One public API, with “date” payload would be exposed on nodes.
2. Central server
    a. Listens to all the notifications, and add sites making notification into “to-be-updated” queue if it already isn’t in the queue.
    Reasons:
        i. Keeping scalability in mind. Because with millions of nodes probably in future, direct update would not be a good option, and can possibly throttle the central server.
        ii. The node owner might be updating one by one, and instead of those x requests with cache data, we can possibly combine all x requests, and visit the site once afterward to get all.
    b. Each “to-be-updated” node is taken out once from the queue on FIFO basis, and public API is called.
    c. The API call is made, with date payload. (date = last_updated time of node), and amber_cache table is fetched with amber_cache.date > date(payload).
    d. Only the provider and provider_id will be kept with central server.

The above process can also be explained through state diagram added below.

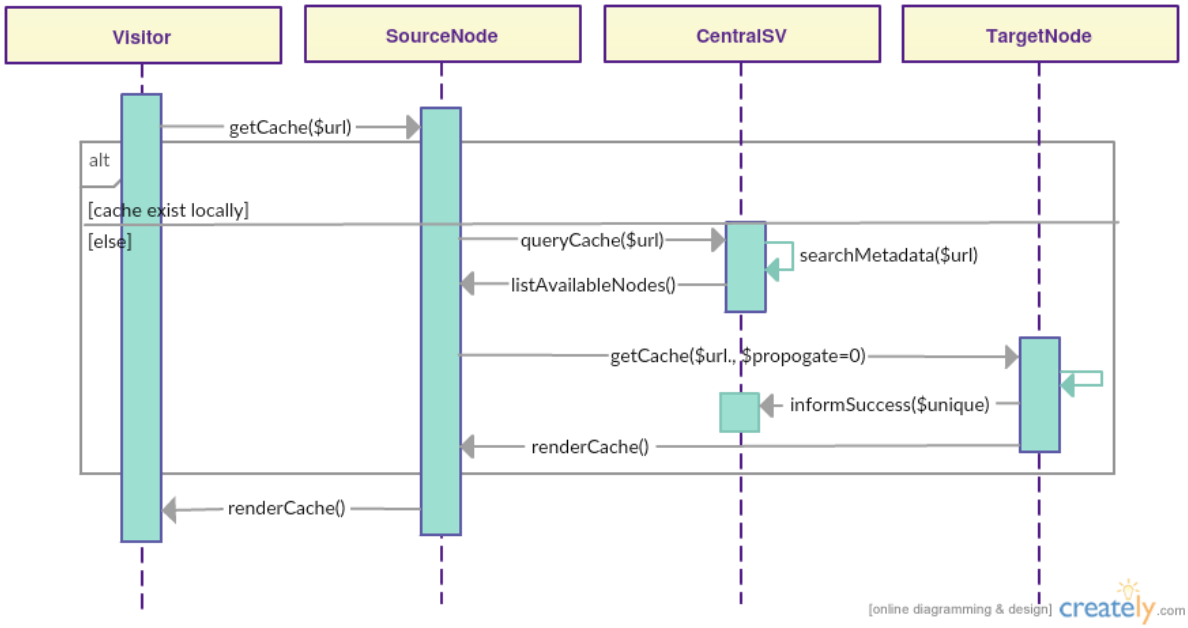


Cache Discovery

The metadata information containing information of available cached pages, time updated, and node information would be saved on central server, based on above implementation. The cache discovery period is when the user hovers over the link, or the cache page is opened. The behaviour would be as follows:

1. Look into self’s (node) cache, and try getting cached content. I.e. display_cached_content.
2. Query the central server (memento) for location of node containing cache.
3. Get the list of nodes and store as queue.
4. Try fetching from nodes one by one.

5. On successful fetch, store it on self (based on node's configuration)
6. Render cache to the visitor.



Security Protocol/Handshake

100s/1000s of Amber nodes would be left exposed to be used by external nodes, thus, doing it without security layer, isn't a good idea.
Similarly, the central server, would be exposed to any device, without any accountability. For the same this section proposes security measures, which can bring some accountability for fair usage, etc.

Workflow

1. Amber node's owner need to register on central server, which will generate a public & private key pair for them. (and store with itself also)
2. All the communication made with central server would need this public/private key pair compulsory, and would be over HTTPS.
3. When central server would return listAvailableNodes(), it will give unique hash for each node also.
   Hash = Encrypt(public_key_target_node, <$url, $unique_random_hash>)
4. The target node will decrypt the hash using private key and check if it is legitimate request, i.e. sent after handshake with central server. It will honor every $unique_random_hash only once. Thus, source_node can't throttle target_node unnecessary and need to authenticate new request with central server.

The unique_random_hash enables writing of rate limiters on central server only, which would automatically be honored by all nodes.
Also, central server protect the nodes, by not sharing same node repeatedly (alternative nodes whenever possible).

This table describes the keys available and not-available to every actor on system.

Key/Hash	Source Node	Central Server	Target Node	Visitor
S.N. Public Key	✔	✔	✘	✘
S.N. Private Key	✔	✔	✘	✘
T.N. Public Key	✘	✔	✔	✘
T.N. Private Key	✘	✔	✔	✘
Hash	✔	✔	✔	✘
Unique Random Hash	✘	✔	✔	✘

Secure  **Vulnerable**

Note:
As the unique hash can only be decrypted by T.N. Private Key, and T.N. don't know the unique hash until successful connection has been made between S.N. and T.N., i.e. hash has been transmitted to T.N.
Thus, whenever T.N. registers informSuccess() to Central Server, it implies T.N. was alive and accessible to S.N. This saves us from polling for active nodes time to time, plus, all successful P2P nodes communication are registered in central server for reputation increment (next section).

Reputation of nodes

This section This section described the feedback based mechanism to automatically reduce potential spammy nodes.
At the beginning every registered node would be given Reputation x.
The following things will affect the reputation, in positive and negative manner. (values would be left configurable in central server)

- Vote up to cache served from remote Node by visitor.
- Vote down to cache served from remote Node by visitor.
- Every successful cache sharing with peers.
- Verified email ID and phone number of node.
- (more as per discussion with mentors)

If the time permits (or as future work), the reputation would be tried to made compatible/follow PeerTrust as suggested by:
Li Xiong and Ling Liu, "PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843-857, July 2004.

Project Plan

This is the project plan, in which I have tried to schedule work as per the timeline of GSoC.

Days	Work to be done
April 22	Student Projects Announced
April 22 - May 22	Community Bonding
April 22 - April 28 (1 week) (clashing period)	● Initial mentor discussion and finalization for: ○ Weekly meeting time and method ● Discussion on weekly status update method. ○ Blog/Wiki? ○ Email? ● Setup of wiki/blog/mailing list as per above. ● Setting up IPFS.
April 29 - May 15 (~ 2.5 weeks)	● Revisit codebase and understand changes made after commit #cfaa5cb. ● Finalize design of distributed Amber with mentors. ○ Discussion on pros/cons of existing design. ○ More reading and research for alternative designs. ● Finalizing IPFS implementation & retd. discussion.
May 16 - May 22 (1 week)	● Integration of IPFS with Amber as backend option. ● Libraries and tools selection for finalized design of distributed Amber.
May 23	First Coding Period Starts
May 23 - May 29 (1 week)	● IPFS integration (ongoing) ○ Refactoring. ○ Writing test cases and documentation. ○ Changes according to the review comments.
May 30 - June 05 (1 week)	● Getting IPFS branch merged on master. ● Release of IPFS feature. ● Immediate bug fixes (if any) after release.
June 06 - June 12 (1 week)	● API endpoints writing in Amber codebase ○ To share newly created cache objects with date payload. ○ Identify events and code writing to call notify() function (left as endpoint). ○ (any other, as per new finalized design)
June 13 - June 19 (1 week)	● Setup central server ○ Based on memento. ○ Make it able to update based on notify() calls. ● Dashboard for application registration on central server.
June 20 - June 26	● Midterm Evaluation

(1 week)	● Buffer period
June 21 - June 28	Midterm Evaluations
June 28	Second Coding Period Starts
June 27 - July 06 (1.5 week)	● Implementation of Cache discovery from Amber nodes. ○ Supporting all cache types. ○ Query memento server for metadata, and actual page cache from nodes. ● Addition of email verification and OTP to Dashboard. ○ OTP (if needed by mentors)
July 07 - July 10 (0.5 week)	● Integration of cache discovery logic with plugin.
July 11 - July 17 (1 week)	● Add more policies to cache discovery ○ Reputation based cache discovery logic. ○ Reputation +/- rules finalization with mentors.
July 18 - July 24 (1 week)	● Security layer addition for communication between nodes
July 25 - July 31 (1 week)	● Performance testing and fixes. ● JMeter test plans and load testing. ○ Central server ○ Profiling the APIs, and server communication
Aug. 01 - Aug. 07 (1 week)	● Review period - Amber (can be extended and buffer period can be utilized for the same, depending on Mentor's availability during this period) ● Test cases writing for distributed Amber.
Aug. 08 - Aug. 14 (1 week)	● Finalize Documentation for developers and users. ● Getting work merged on master.
Aug. 15 - Aug. 16 (2 days)	● Unofficial/Personal pen down target. ● Buffer period (this and further)
Aug. 16 - Aug. 24	Submit Code and Evaluations
Aug. 24 - Aug. 30	Mentors Submit Final Evaluations

Note:

- Open to change the timeline according to mentor.
- Holidays and other stuff are also considered while writing project plan.
- GSoC timeline events have been marked in grey, and merged, for better representation.
- Buffer time period have been kept multiple times, to cope up with any unexpected changes.

Primary and Stretch goals

The project can be divided into various sub-projects, which are highly related to each other. The various aspects in term of priority and other classification is shared below:

Name	Priority	Primary?	Comments

IPFS	Low	✗	• Can be possibly different project, and not be part of this proposal
			• Still included because
			  ○ Implementation would be done even before start of coding phase of GSoC. Thus wouldn't affect GSoC proposal as such.
			  ○ Would be great hands on, before writing code for distributed Amber. Similarly working of the review process, etc in Amber.
			  ○ Points of effect, etc have already been identified.

Distributed Amber + Memento	High	✓	• Main deliverable at the end of project.
Cache discovery	High	✓	• Main deliverable at the end of project.
Security protocol	Normal	✓	• Main deliverable at the end of project. • Extension to work done in previous 2 sub-project. • The end users might hesitate to use distributed version without any security feature. Due to which it is must, to be included, at the time of releasing Distributed Amber.
Reputation	Low	✗	• Amber end users would not ideally wish to *leave it on chance* of serving any spam content/caches to their users. • Thus, they would look forward for such mechanism to be taken care of by Amber. • Due to which, this sub-project can be part of GSoC timeline. (as an improvement of above implementation). • In case the timeline is not met by developer due to any circumstances or unpredictable issues, it can be possibly considered as Future work. Thus, not included as Primary Goal.

Infrastructure requirements

Permanent container/server for:

- Central server.

Following infrastructure would be required from Berkman during the development and testing phase of project.

- 2 containers
    - Source and Target server.
    - JMeter tests to Central server.
- Possibly in same network to central server, which might be helpful during JMeter testings. (so none of intermediate network get throttled by requests.)

Thanking Note to organisation

I would like to thank all the members of Berkman organization, to give me this opportunity to write proposal and of providing possibility to work with you.

I will look forward to every feedback from organisation members reviewing this document, and would be glad to discuss/change accordingly.