

**دانشگاه صنعتي شريف**

**دانشکده مهندسي کامپيوتر**

**سمينار کارشناسي ارشد گرايش شبکه‌های کامپیوتری**

**عنوان:**

**طراحی بستر مشترک ذخیره‌سازی محلی داده در اینترنت اشیاء**

**Designing a shared local data storage in IoT**

**نگارش:**

**عمران باتمان‌غلیچ**

**۹۸۲۰۹۴۴۵**

**استاد راهنما:**

**دکتر علی‌محمدافشین همت‌یار**

**استاد ممتحن داخلي:**

**دکتر جعفر حبیبی**

بهمن ۹۹

**فهرست مطالب**

[1 مقدمه 3](#_Toc63417166)

[2 متن گزارش 4](#_Toc63417167)

[2.1 بررسی تکنولوژی‌های مرتبط 4](#_Toc63417168)

[2.2 تست عملی کار با تکنولوژی‌های مرتبط 4](#_Toc63417169)

[2.3 طرح اولیة بستر ذخیره‌سازی مشترک داده 5](#_Toc63417170)

[2.4 نقاط ضعف و ادامة کار 6](#_Toc63417171)

[2.4.1 طراحی لایة انتزاعی بر روی بستر ذخیره و بازیابی داده 6](#_Toc63417172)

[2.4.2 طراحی دقیق پروتکل کشف سرویس 7](#_Toc63417173)

[2.4.3 در نظر گرفتن جنبة امنیت در طراحی 7](#_Toc63417174)

[2.4.4 بررسی معیارهای عملکرد در طراحی پیشنهادی 7](#_Toc63417175)

[3 جمع‌بندی 7](#_Toc63417176)

**چکيده:** پروژة پیش رو قصد دارد در ناحیة محلی، با ایجاد یک بستر شفاف ، منابع ذخیره داده را به نحوی به اشتراک بگذارد که امکان استفادة اشتراکی از فضاهای ذخیره‌سازی دستگاه‌ها فراهم شود و میزان تاثیر آن بر کارایی کل سیستم را محاسبه کند. این امر باعث می‌شود فضای نهایی بیشتری در اختیار اجزا قرار داده شده و هر جزء در عین اینکه به ظرفیت حافظة خود محدود نیست، مجبور به ارتباط با اجزای ذخیره‌سازی دورتر نیز نباشد. این رویکرد در موارد مشابهی برای بهینه‌سازی ارتباط استفاده شده است؛ از جمله روش‌های پویای مسیریابی از ناحیه محلی به سرورهای ابری. [4] علاوه بر این، با بهینه‌سازی انتقال داده بین اجزای نزدیک به هم، با گذر زمان می‌توان کارایی کل سیستم را افزایش داد. در این حالت، در واقع اجزا از یک حافظة سریعتر، بزرگتر و نزدیک‌تر بهره خواهند برد.

**واژه هاي كليدي:** اینترنت اشیاء، Mist Computing، Fog Computing، Shared Storage

# مقدمه

رشد روزافزون اینترنت اشیاء نه تنها در تعداد و تنوع اجزا خود را نشان داده، بلکه حجم داده‌های مورد مبادله در اینترنت اشیاء را نیز تحت تأثیر قرار داده است. در مقابلِ این حجم زیاد داده‌ها راه حل‌های مناسبی نظیر بهره‌گیری از رایانش ابری ارائه شده است؛ در این مدل، داده‌های حجیم با قرارگیری بر روی سرورهای ابری و مبادله به هنگام نیاز، بسیاری از مشکلات مرتبط با خود را پوشش داده اند. در مقابل، نگرانی‌هایی از جنس امنیت داده‌ها، دسترس‌پذیری و کارایی باعث شده است تا راه حل‌های جدیدتری نظیر محاسبات مهی در صدد رفع چنین مشکلاتی برآیند. [1]

اخیراً معماری‌هایی پیشنهاد شده اند که با روشی ترکیبی، از مزایای رایانش ابری استفاده کرده و معایب فوق الذکر را مرتفع یا کم‌اثر کرده اند. [2] اما همچنان مسألة توزیع نامتوازن منابع یکی از دلایل مهم و تأثیرگذار در کارایی اجزای اینترنت اشیاء است. [3] هرچند محاسبات مهی در همین زمینه بهبودهایی حاصل کرده است؛ از جمله ایجاد مکانیزم حافظه نهان بین لبه و منابع ابری که از طرفی فضای ذخیره‌سازی و محاسباتی بیشتری برای لبه فراهم می‌کند، و از سوی دیگر، امکان استفاده از منابع ابری را به شیوه‌ای بهینه‌تر فراهم می‌کند. [2]

به طور کلی معماری ترکیبی ابر و مه را می‌توان متشکل از دو قسمت اصلی دانست: ۱. قسمت محلی (شامل دستگاه‌ها و اجزای لبه و اصطلاحاً Mist Node ها و ارتباطشان با سرورهای مه) ۲. قسمت عمومی (ارتباط سرورهای مه با سرورهای ابری)

پروژة پیش رو قصد دارد در ناحیة محلی، با ایجاد یک بستر شفاف ، منابع ذخیره داده را به نحوی به اشتراک بگذارد که امکان استفادة اشتراکی از فضاهای ذخیره‌سازی دستگاه‌ها فراهم شود و میزان تاثیر آن بر کارایی کل سیستم را محاسبه کند. این امر باعث می‌شود فضای نهایی بیشتری در اختیار اجزا قرار داده شده و هر جزء در عین اینکه به ظرفیت حافظة خود محدود نیست، مجبور به ارتباط با اجزای ذخیره‌سازی دورتر نیز نباشد. این رویکرد در موارد مشابهی برای بهینه‌سازی ارتباط استفاده شده است؛ از جمله روش‌های پویای مسیریابی از ناحیه محلی به سرورهای ابری. [4] علاوه بر این، با بهینه‌سازی انتقال داده بین اجزای نزدیک به هم، با گذر زمان می‌توان کارایی کل سیستم را افزایش داد. در این حالت، در واقع اجزا از یک حافظة سریعتر، بزرگتر و نزدیک‌تر بهره خواهند برد.

نکتة کلیدی در این طراحی توازن سرعت و ظرفیت فضاست؛ به این صورت که هرچه به درخواست‌کننده نزدیک‌تر می‌شویم سرعت بیشتر و در عوض (احتمالاً) فضای ذخیره‌سازی کمتری خواهیم داشت. [5] و برعکس هرچه از درخواست‌کننده دورتر می‌شویم فضای ذخیرة بزرگتر ولی سرعت کمتری خواهیم داشت، تا جایی که ـ در بدترین شرایط ـ مجبور به استفاده از دورترین، کندترین و البته بزرگترین منبع، یعنی منبع ابری می‌شویم. در این میان تنظیم این توازن می‌تواند در کارایی بهینة کل سیستم تاثیرگذار باشد.

به منظور اندازه‌گیری میزان تأثیر طرح فوق بر کارایی سیستم، معیارهای مختلفی می‌توان ارائه داد؛ از جمله سرعت خواندن و نوشتن اطلاعات (b/s)، میانگین تأخیر شبکه، میزان بهره‌وری از مجموع ظرفیت ذخیره‌سازی دستگاه‌های محلی، دسترس‌پذیری و... . برای سنجش این معیارها سعی می‌شود از مزایای هر دو محیط واقعی و شبیه‌سازی بهره گرفته شود؛ اما به فراخور نوع و پیچیدگی محاسبة هر معیار ممکن است برای برخی محدود به استفاده از یک محیط خاص شویم و یا به محاسبات نظری بسنده کنیم.

# متن گزارش

در ادامه به گزارش مختصری از کارها و تحقیقات انجام‌گرفته پرداخته خواهد شد. این گزارش شامل سه بخش کلی در رابطه با بررسی تکنولوژی‌ها، تست عملی کار با آن‌ها و ارائة طرح اولیه خواهد بود.

## بررسی تکنولوژی‌های مرتبط

در گام اول، قطعات پایه و اولیه مورد کاربرد در اینرنت اشیاء برای آشنایی با تکنولوژی‌های مرتبط، مورد بررسی قرار گرفتند. برای این منظور، یک تراشه اتمگا۳۲[[1]](#endnote-1)، یک ماژول وای‌فای[[2]](#endnote-2)، یک ماژول رطوبت‌سنج و همچنین یک بورد آردویینو[[3]](#endnote-3) برای انجام آزمایشات اولیه استفاده شدند. البته بدیهتاً برای جلوگیری از اتلاف وقت، این ابزارها در راستای ایجاد یک محصول نهایی و کامل به کار گرفته نشدند؛ بلکه صرفاً برای آشنایی با کارکرد آن‌ها برنامه‌ریزی و تست شدند.

در این دوره تلاش شد تا بستر اولیه‌ای از اینترنت اشیاء فراهم شود تا بتوان در آینده ایدة ذخیره‌سازی محلی داده را بر روی آن اجرا و تست کرد؛ اما پس از بررسی‌های به عمل آمده، نویسنده به این نتیجه رسید که بجای تمرکز بر روی ابزارهای سطح پایین[[4]](#footnote-1) [6]- که راه را برای رسیدن به یک بستر حداقلی از اینترنت اشیاء، طولانی می‌کند – از اشیاء مناسب‌تری نظیر ساعت هوشمند، تلفن همراه، تلویزیون هوشمند و ... استفاده شود.

این کار اگرچه به لحاظ اقتصادی برای تشکیل بستر اینترنت اشیاء به‌صرفه نیست، اما در زمان کمتری می‌تواند ایدة بستر مشترک ذخیره‌سازی داده را مورد ارزیابی قرار دهد.

در ادامه جدول مربوط به مقایسه هرکدام از دو روش فوق برای فراهم‌سازی بستر اولیة اینترنت اشیاء مشاهده می‌شود.

جدول 1 مقایسه تجهیزات مبتنی بر اتمگا و تجهیزات هوشمند

|  |  |  |
| --- | --- | --- |
| **عنوان** | **بورد آردویینو (اتمگا، ماژول وای‌فای و...)** | **تجهیزات هوشمند (ساعت، تلویزیون و ... هوشمند)** |
| مصرف انرژی | کمتر | بیشتر |
| قدرت پردازش | کمتر | بیشتر |
| سیستم عامل | ندارد | دارد |
| کتابخانه‌های نرم‌افزاری تسهیل‌گر | دارد | دارد |
| هزینه اقتصادی | کمتر | بیشتر |

## تست عملی کار با تکنولوژی‌های مرتبط

پس از بررسی‌های اولیه بر روی کارکرد ساعت هوشمند، نحوه ارتباط آن با سایر اجزای موجود در شبکه مورد ارزیابی قرار گرفت. به منظور راحتی توسعه و ارزیابی، از امکان شبیه‌سازی سیستم‌عامل تایزن[[5]](#endnote-4) بهره برده شد. [7] در ذیل یک نمونه از شبیه‌سازی این ارتباط قابل مشاهده است.



شکل 1 شبیه‌سازی دریافت داده تحت شبکه توسط ساعت هوشمند

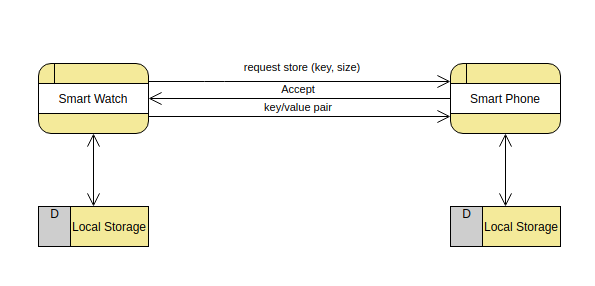
سمت راست: قبل از درخواست، سمت چپ: مدت زمان دریافت داده

همانطور که مشاهده می‌شود، این شبیه‌سازی صرفاً یک امکان‌سنجی[[6]](#endnote-5) ساده برای انتقال داده توسط ساعت هوشمند (به عنوان جزئی از اینترنت اشیاء) تحت شبکه وایرلس است. در بخش بعدی، جزئیات بیشتری از طراحی اولیة صورت‌گرفته ارائه خواهد شد.

## طرح اولیة بستر ذخیره‌سازی مشترک داده

در ادامه، طراحی اولیه به عنوان کوچکترین محصول قابل رشد[[7]](#endnote-6) ارائه می‌شود. پس از آن نقاط ضعف و قوت آن بررسی و پیشنهادات توسعه ارائه خواهد شد.

ایده اولیه، ارتباط تک‌به‌تک[[8]](#endnote-7) اجزای اینترنت اشیاء برای ذخیره‌سازی و بازیابی داده است. به این صورت که یک درگاه[[9]](#endnote-8) مشخص به همراه یک پروتکل مورد توافق واقع شده که هر یک از اجزا بتوانند از طریق آن، اقدام به ذخیره و بازیابی داده نمایند. در ذیل نمونه‌ای از این ارتباط را تحت نمودار جریان داده[[10]](#endnote-9) مشاهده می‌کنید.



شکل 2 نمودار جریان داده

همانطور که مشاهده می‌شود، ساعت هوشمند برای ذخیرة داده، ابتدا حجم دادة مورد نظر و کلید متناظر آن را به تلفن هوشمند ارسال می‌کند[[11]](#footnote-2). چنانچه تلفن هوشمند فضای کافی برای ذخیره داده را داشته باشد، با یک پیام متناظر، منتظر دریافت محتوا می‌شود. در این مرحله، ساعت هوشمند اقدام به ارسال دادة مورد نظر می‌کند.

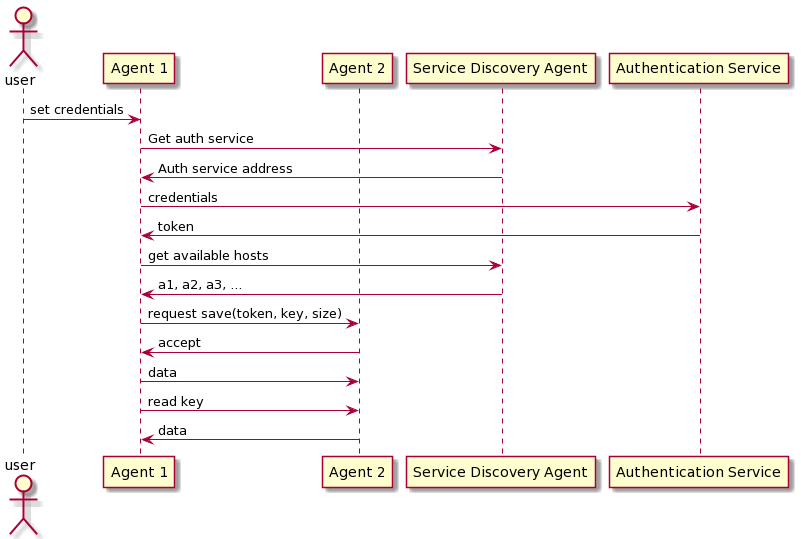
توضیح فوق، ساده‌ترین سناریوی ممکن برای ذخیره داده است؛ حال در ادامه به توضیح جزئیات مرتبط می‌پردازیم.

مورد اول، نحوة یافتن آدرس میزبان برای درخواست ذخیره‌سازی است. این کار توسط فرایند کشف سرویس[[12]](#endnote-10) انجام می‌پذیرد. [8]

مورد دوم، نحوه تضمین یکتایی کلید ذخیره‌سازی است. همانطور که اشاره شد، ذخیره سازی به صورت زوج کلید و مقدار[[13]](#endnote-11) انجام می‌شود. اما مهم است که کلیدها در سطح کل شبکه یکتا باشند؛ در غیر این صورت، امکان برخورد[[14]](#endnote-12) و از دست رفتن داده وجود دارد. برای رفع این مشکل، شناسة ذخیره‌کننده به کلید افزوده می‌شود.

سومین مورد، بحث امنیت ذخیره‌سازی است. ذخیره‌سازی قابلیتی است که فقط در اختیار اجزاء مجاز قرار می‌گیرد. به عنوان راه حل اولیه، این وظیفه به عهده یک سیستم واسط گذاشته شده است که وظیفة اعطای گواهی به اجزاء اینترنت اشیاء را دارد؛ به این صورت که هر جزء، کلید خود را – که در مورد دوم ذکر شد – از این سیستم واسط دریافت می‌کند. کلید هر شیء در شبکه، رشته‌ای است که نام آن شیء را به همراه مدت اعتبار آن امضا[[15]](#endnote-13) کرده و در خود جای داده است. [9] البته که این روش کامل نیست و برای تکمیل آن نیاز است ارتباط اولیة امن بین اجزاء و سیستم واسط نیز طراحی شود. در اینجا فرض شده است که این ارتباط امن است و برای پیشبرد مراحل دیگر به آن اکتفا شده است.

شمای کلی ارتباطات ذکر شده بین اجزاء مختلف را در نمودار زیر می‌توان مشاهده کرد.



شکل 3 نمودار شمای کلی ارتباطات بین اجزای مختلف برای ذخیره‌سازی داده

## نقاط ضعف و ادامة کار

در این گزارش به بیان طرح اولیه از ارتباط اجزاء پرداخته شد. در ادامه قسمت‌هایی که در آینده باید تکمیل شوند به اختصار ذکر خواهند شد.

### طراحی لایة انتزاعی بر روی بستر ذخیره و بازیابی داده

همانطور که مشاهده می‌شود، در طراحی کنونی، ذخیره‌سازی و بازیابی داده به صورت کاملا تک‌به‌تک انجام می‌شود؛ به این صورت که هر نود مسئولیت به خاطر سپردن نود میزبان دادة خود را داراست. در حالی که نیاز است یک لایة انتزاعی، مدیریت این امر را به عهده بگیرد و حتی در صورت لزوم اقدام به جابجایی یا تصمیم‌گیری در مورد محل جدید داده‌ها بکند.

مواردی که در این طراحی باید در نظر قرار بگیرند به طور عمده عبارتند از:

* تعیین حد آستانه برای حجم ذخیره‌سازی محلی؛ هر نود باید قبل از اشغال شدن تمام ظرفیت خود، اقدام به توزیع داده‌ها در نودهای دیگر کند.
* ذخیرة همزمان در فضای محلی و فضای یک نود دیگر برای تسریع در فرایند نوشتن و خواندن؛ این ایده که همزمانی نوشتن یا خواندن توسط چند دستگاه می‌تواند سرعت را در کل افزایش دهد باید مورد آزمایش قرار گیرد.
* مدیریت محل‌های ذخیره‌سازی؛ بدین معنا که محل ذخیره‌سازی هر جزء از داده باید به صورت محلی و یا غیر محلی (در قالب یک دادة جدید) ذخیره و مدیریت شود تا در آینده قابل بازیابی باشد.
* ساخت کلید یکتا با استفاده از شناسة نود مبدأ. همانطور که گفته شد، کلیدهایی که برای ذخیره سازی استفاده می‌شوند باید در سطح کل سیستم یکتا باشند.
* و ...

### طراحی دقیق پروتکل کشف سرویس

پروتکل کشف سرویس باید به گونه‌ای باشد که در کنار استفاده از راه‌حل‌های کنونی مثل تعریف دامنه، تایموت[[16]](#endnote-14) و ... که در مرجع پیوست ذکر شده است [8]، موارد خاص‌منظوره جهت ذخیره و بازیابی داده را نیز شامل شود؛ به عنوان مثال، حداقل فضای مورد نیاز به عنوان آرگومان به سیستم کشف سرویس داده شود و در مقابل، اجزائی که فاقد حداقل فضای مورد نیاز هستند در لیست نتایج بازگردانده نشوند.

با توسعة این روش، مواردی نظیر ایجاد تعادل بار[[17]](#endnote-15) نیز به طریق مشابه قابل پیاده‌سازی است.

### در نظر گرفتن جنبة امنیت در طراحی

با شکل‌گیری طراحی، دغدغه‌های مربوط به امنیت نیز از زاویه نگاه خود باید مورد بررسی قرار گیرد و در صورت لزوم اقدام به تکمیل طراحی از این بعد شود.

### بررسی معیارهای عملکرد در طراحی پیشنهادی

مواردی نظیر سرعت خواندن و نوشتن، مصرف انرژی، بهره‌وری منابع و ... باید مورد ارزیابی قرار گیرد و با نمونه‌های مشابه مقایسه شود. این ارزیابی می‌تواند شامل شبیه‌سازی، محاسبة نظری و یا اجرای عملی بخش‌های مختلف طرح پیشنهادی باشد.

# جمع‌بندی

در طی ۳ ماه گذشته تلاش شد تا با مطالعة کارهای مرتبط و فراگیری ادبیات حوزة اینترنت اشیاء، پیشنیازهای لازم برای طراحی بستر مشترک ذخیره‌سازی محلی داده آماده شده و حتی الامکان مواردی که ممکن است کار را در ادامه دچار چالش کند زودتر کشف شود. به همین دلیل سعی شد فرایند مطالعه، فراگیری تکنولوژی و تست عملی ابزارها با رویکردی موازی پیش گرفته شود، تا زوایای پنهان کار در سطوح مختلف کشف و پیش‌بینی شود.

همچنین در این گزارش تلاش شد تا با ارائة خلاصه‌ای از طراحی اولیه، مسیر طی شده تا کنون شفاف شده و همچنین نقشة راهی برای آینده شکل گیرد. مشکلات و نقاط ضعفی که نویسنده در طی طراحی با آن‌ها مواجه گردید نیز بیان شد. برخی از آن‌ها توسط مقالاتی که مورد اشاره قرار گرفتند پاسخ داده شده‌اند و لذا در طراحی اولیه از ایدة آن‌ها الهام گرفته شده است؛ اما برخی نیز نیاز به کار بیشتر و طراحی دقیق‌تری دارند که در بخش ۲ قسمت ۴ مورد اشاره قرار گرفتند.

**مراجع**

|  |  |
| --- | --- |
| [1] | B. Donassolo, I. Fajjari, A. Legrand and P. Mertikopoulos, "Fog Based Framework for IoT Service Provisioning," *16th IEEE Annual Consumer Communications & Networking Conference (CCNC),* pp. 1-6, 2019. |
| [2] | T. Wang, J. Zhou, A. Liu, M. Z. A. Bhuiyan, G. Wang and W. Jia, "Fog-Based Computing and Storage Offloading for Data Synchronization in IoT," *IEEE Internet of Things Journal,* pp. 4272-4282, 2018. |
| [3] | F. Karatas and I. Korpeoglu, "Fog-Based Data Distribution Service (F-DAD) for Internet of Things (IoT) applications," *Future Generation Computer Systems,* vol. 93, pp. 156-169, 2019. |
| [4] | F. Y. Okay and S. Ozdemir, "Routing in Fog-Enabled IoT Platforms: A Survey and an SDN-Based Solution," *IEEE Intenet of Things Journal,* vol. 5, pp. 4871-4889, 2018. |
| [5] | R. Mahmud, F. L. Koch and R. Buyya, "Cloud-Fog Interoperability in IoT-enabled Healthcare Solutions," *19th International Conference on Distributed Computing and Networking,* pp. 1-10, 2018. |
| [6] | P. Patel and D. Cassou, "Enabling high-level application development for the Internet of Things," *Journal of Systems and Software,* vol. 103, pp. 62-84, 2015. |
| [7] | W. Jackson, SmartWatch Design Fundamentals, Berkeley: Apress, 2019. |
| [8] | S. Mastorakis and A. Mtibaa, "Towards Service Discovery and Invocation in Data-Centric Edge Networks," *IEEE 27th International Conference on Network Protocols (ICNP),* pp. 1-6, 2019. |
| [9] | I. I., R. A. P.M. and V. Bhaskar, "Encrypted token based authentication with adapted SAML technology for cloud web services," *Journal of Network and Computer Applications,* vol. 99, pp. 131-145, 2017. |

**واژه نامه**

1. ATMEGA32 [↑](#endnote-ref-1)
2. ESP8266 V1.0 WiFi [↑](#endnote-ref-2)
3. ARDUINO UNO [↑](#endnote-ref-3)
4. منظور از سطح پایین، ارتباط نزدیک با سخت‌افزار است که به دلیل عدم وجود abstraction، موارد مختلف توسعه نرم‌افزار را با پیچیدگی زیاد و نامرتبط با پروژه روبرو می‌کند. [↑](#footnote-ref-1)
5. Tizen [↑](#endnote-ref-4)
6. Feasibility Study [↑](#endnote-ref-5)
7. Minimum Viable Product (MVP) [↑](#endnote-ref-6)
8. Peer to Peer [↑](#endnote-ref-7)
9. Interface [↑](#endnote-ref-8)
10. Data Flow Diagram [↑](#endnote-ref-9)
11. این ارسال می‌تواند تحت شبکه وای‌فای، بلوتوث و یا هر شبکه دیگر انجام گیرد. در مثال کنونی، مورد اول مد نظر قرار داده شده است. هرچند تاثیری در طراحی ندارد. [↑](#footnote-ref-2)
12. Service Discovery [↑](#endnote-ref-10)
13. Key-Value Pair [↑](#endnote-ref-11)
14. Collision [↑](#endnote-ref-12)
15. Sign [↑](#endnote-ref-13)
16. timeout [↑](#endnote-ref-14)
17. Load Balance [↑](#endnote-ref-15)