گزارش بررسی پیکربندی نامناسب سرویس‌های NoSQL در اشل پروژه‌های بزرگ به تفکیک تکنولوژی‌های NoSQL

علیرضا سلطانی نشان

۹ آذر ۱۴۰۲

# فهرست مطالب

# ۱ تعریف مسئله

ذخیره‌سازی اطلاعات از مهم‌ترین نیازهای تحلیل کنندگان داده است. امروزه با توجه به پیشرفت صنعت IoT و یادگیری ماشین، تولید داده‌ها بسیار افزایش یافته است به گونه‌ای که بتوان این داده‌ها را به سریع‌ترین روش ممکن در محلی مناسب ذخیره‌سازی و نگهداری کرد. افراد برای ذخیره‌سازی این داده‌ها نیاز به نصب و راه‌اندازی یک سیستم DBM دارند که از طریق یک واسط با زبانی مناسب بتوانند به آن متصل شده و داده‌های دریافتی را بعد از تجزیه و تحلیل آنها در این محل ذخیره‌سازی و مدیریت کنند. امروزه محققان ترجیح می‌دهند به دلیل مقیاس پذیری بیشتر، سیستم‌های توزیع شده و قابلیت پایداری بالا از دیتابیس‌های رابطه‌ای به سمت دیتابیس‌های NoSQL مهاجرت کنند. این نوع دیتابیس‌ها امروزه توسط تمام اپلیکیشن‌های جدید پشتیبانی می‌شوند و برای استفاده آسان طراحی شده‌اند. حتی می‌توان متذکر شد که تعداد زیادی از سرویس‌های ذخیره‌سازی ابری امروزه از سرویس‌های دیتابیسی NoSQL پشتیبانی گسترده‌ای دارند. این ارائه دهندگان اغلب شرکت‌های معروفی مانند MS Azure CosmosDB Google Cloud Database Amazon DynamoDB می‌باشند. همچنین بیشتر این موتورهای دیتابیسی به صورت متن‌باز هستند و توسعه دهندگان زیادی از سرتاسر جهان روی آنها مشغول توسعه هستند.

در سال‌های اخیر، با پدید آمدن و رشد سریع سرویس‌های دیتابیسی NoSQL بین عموم توسعه‌دهندگان استفاده از این نوع سرویس‌ها افزایش یافته است. دلیل اصلی این محبوبیت نصب و راه‌اندازی و استقرار آسان آنها در هر محلی است. همچنین قابل اعتماد هستند، روش‌ها و مکانیزم‌های زیادی برای تهیه نسخه‌های پشتیبان‌گیر به صورت منظم از داده‌ها را ارائه می‌دهند. دلیل اصلی آسان بود این سیستم آن است که در هنگام راه‌اندازی آنها زمان زیادی را صرف نمی‌کنید، زیرا بعد از نصب اولیه و طی کردن فرایند نصب با زدن روی دکمه "بعدی" دیتابیس شما آمادست و می‌توانید از آن در برنامه خود استفاده کنید. بعد از این فرایند هیچ عملیاتی بر روی تعریف دسترسی‌ها، مدیریت کاربران در استفاده از دیتابیس مانند اختصاص سطح دسترسی، توسط راه‌انداز سیستم DBM صورت نمی‌گیرد. نتیجه این موارد پیکربندی غیر اصولی و اشتباه [۱] سیستم ذخیره‌سازی داده می‌شود که در نتیجه افشای اطلاعات حساس [۲] را به دنبال خواهد داشت.

سوالی که ممکن است در اینجا مطرح شود آن است که چه زمانی پیکربندی نادرست موجب افشای اطلاعات می‌شود؟

در ابتدا بعد از راه‌اندازی این نوع دیتابیس‌ها اولین هدف استفاده از آنها در محیط لوکال در یک شبکه است. اما افشای اطلاعات و پیکربندی اشتباه زمانی رخ می‌دهد که این دیتابیس‌ها در شبکه اینترنت مورد دسترسی قرار گیرند.

محققان با توجه به موارد گفته شده بالا توانسته‌اند یک ابزار خودکار جهت آنالیز و جست و جوی سیستم‌های دیتابیسی NoSQL را توسعه دهند که به وسیله آن می‌توانند پیکربندی نامناسب این سیستم‌های مستقر شده را متوجه شده، موارد آسیب‌پذیری را گزارش و سپس به صاحبان این دیتابیس‌ها هشداری در جهت در خطر بودن اطلاعاتشان ارسال کنند.

در این گزارش به طور خلاصه تمام موارد انجام شده را در پنج عنوان توضیح می‌دهیم. در ابتدا در مورد چالش‌ها و نحوه تحقیق روی این آسیب پذیری‌ها و عدم وجود پیکربندی مناسب می‌پردازیم. در بخش مدل پیشنهادی بیشتر ماهیت ابزار توسعه داده شده را مطرح می‌کنیم و سپس نتایج اجرای این ابزار را نمایش می‌دهیم و در نهایت به نوآوری و کارهای آینده می‌پردازیم.

---

[۱] Misconfigured

[۲] Data Leakage

# ۲ چالش‌ها

ابزاری توسعه داده شده است که در یک رنج گسترده‌ای از آدرس‌های IP می‌تواند اینگونه دیتابیس‌ها را اسکن کند و افشای سرویس آنها را تشخیص دهد. این تشخیص به شکل ایمن بدون هیچ نگهداری داده‌ها و یا افشای اطلاعات حساس آنها صورت می‌گیرد. بررسی ضعف پیکربندی‌های صورت گرفته بر روی ۶۷ میلیون ۷۲۶ هزار و ۶۴۱ آدرس IP بوده است که بین بازه زمانی اکتبر ۲۰۱۹ و مارچ ۲۰۲۰ تکمیل شده است. نکته جالب از آنجایی شروع می‌شود که این سرویس‌ها نه تنها به صورت شخصی راه‌اندازی شده‌اند بلکه تعداد ۱۲ هزار و ۲۷۶ نمونه از آنها در ارائه دهندگان سرویس‌های ابری معروف یافت شده است. با توجه به این موضوع در این تحقیق ۷۴۲ مورد آسیب پذیری پیدا شده است که به صورت مستقیم وب سایت این کاربران به دلیل ضعف در پیکربندی به دیتابیس‌های آنها ارجاع دارد این بدان معناست با وجود تنظیمات و پیکربندی پیش فرض و بدون هیچ گونه استراتژی امنیتی، هر کاربر ناشناس دیگری می‌تواند وارد این دیتابیس‌ها شده و آنها را با نظر و سلیقه خودش تغییر و حتی تخریب به قصد اخاذی کند.

## ۱.۲ بررسی نمونه‌ها در پیکربندی ضعیف راه‌اندازی

۱. در مارچ ۲۰۲۰، ۷ ترابایت از داده‌های سایت بزرگسالان به صورت صریح از یک نمونه دیتابیس Elastic Search با اطلاعاتی از قبیل، نام کاربران، جنسیت و گرایش‌ها، لاگ‌های مربوط به پرداخت‌هایشان، ایمیل، با ۱۰۸۸ میلیارد رکورد مورد افشا قرار گرفت.

۲. در نوامبر سال ۲۰۱۹ یک محقق توانست یک نمونه با پورت باز با بیشتر از ۱/۲ میلیارد رکورد از یک دیتابیس را پیدا کند که شامل اطلاعات حساس کاربران از قبیل آدرس ایمیل آنها بود.

۳. در ژانویه سال ۲۰۱۷، در یک حمله بیشتر از ۶۰۰ نمونه از دیتابیس Elastic search حذف شدند و برای بازیابی آنها از صاحبانشان اخاذی کردند [۴۰].

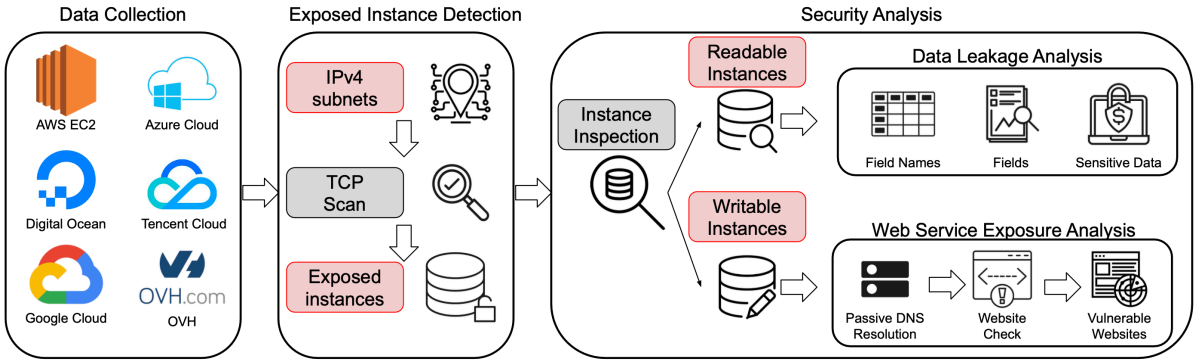
۴. براساس گزارشی در سال ۲۰۱۸ بیشتر از ده ها هزار نمونه از دیتابیس‌های Redis در دسترس کاربران مخرب، آسیب‌پذیر شناخته شدند که به دلیل دسترسی عموم افراد تعداد ۷۵۰۰ سرور یافت شد که در معرض خطر یک بدافزار به نام Botnet بودند که هدف اصلی آنها دزدیدن ارزهای دیجیتال [۳] آن پلتفرم ارائه دهنده بود.

براساس موارد مطرح شده در بندهای گفته شده بالا، اولین بررسی از ضعف پیکربندی دیتابیس‌های NoSQL انجام شده است به گونه‌ای که می‌توان از آن برای تشخیص و تعیین معیاری برای بررسی پیکربندی درست در این دیتابیس‌ها از آن استفاده کرد. محققان یک فریمورکی توسعه داده‌اند که به صورت کاملا خودکار می‌تواند سرویس‌های معرض دید عموم را تشخیص و عملیات بررسی امنیتی روی آنها انجام دهد بدون ذخیره‌سازی داده‌های کاربران یا باز کردن داده‌های دیتابیس پلتفرم‌ها و دریافت اطلاعات حساس آنها.

## ۲.۲ فرایند کلی عملکرد فریمورک

این فریمورک در ابتدا لیستی از آدرس‌های IP که توسط بیشتر ارائه دهندگان سرویس‌های ابری استفاده می‌شود را اسکن کرده و به دنبال ارتباطی باز بر روی پورت پیش فرض دیتابیس NoSQL می‌گردد که بتواند به آن به صورت مستقیم متصل شود. (در شکل ۱، می‌توانید عملکرد فریمورک را در تصویر مشاهده کنید.) سپس می‌تواند به یک نمونه از دیتابیس دسترسی داشته و عملیات بررسی امنیتی خود را شروع کند. به طور کلی این فریمورک به بررسی سطح دسترسی دیتابیس (همان دسترسی‌های

[۳] Cryptocurrencies




شکل ۱: بررسی عملکرد ابزار توسعه داده شده

خواندن و نوشتن روی یک سیستم مدیریت دیتابیس) متا دیتا از قبیل نسخه مورد استفاده از سرویس NoSQL، کاربران مجاز دسترسی به دیتابیس، سطوح دسترسی تعریف شده و جداول مرتبط به این دیتابیس‌ها، می‌پردازد.

### ۱.۲.۲ تشخیص عمل خواندن از دیتابیس‌های فاش شده

اگر این ابزار تشخیص دهد که دسترسی خواندن را از این دیتابیس‌ها دارد تضمین افشای اطلاعات این سیستم‌ها را به طور قطعی می‌دهد که می‌تواند خطری برای محتوای داخل دیتابیس باشد. ابزاری که توسعه داده شده است کاملا ایمن می‌باشد چرا که اصلا وارد محتوای این دیتابیس‌ها و داده‌های آنها نشده و تنها از توابعی مانند تابع Count برای شمارش رکوردهایی که مربوط به فیلدهایی مانند نام کاربران، شماره تلفن یا آدرس ایمیل آنها می‌شود، استفاده می‌کند. اغلب داده‌های جمع‌آوری شده از این دیتابیس‌ها به صورت نمایش تعداد رکوردهای آنها مربوط به فیلدی مشخص است که در جداول صفحات بعدی آنها را مشاهده خواهید کرد.

### ۲.۲.۲ تشخیص عمل نوشتن از دیتابیس‌های فاش شده

زمانی که این ابزار بتواند به این دیتابیس‌ها متصل شود و بعد از آن قادر به ساخت یک workspace یا یک رکوردی از داده NoSQL یعنی همان Document باشد، تشخیص می‌دهد که مجوز نوشتن را در این سیستم دارد به همین خاطر یک پیام جدی را برای صاحبان دیتابیس می‌نویسد تا در جریان ضعف پیکربندی و ایمن نبودن ارتباطات آنها و باز بودن دسترسی‌ها، قرار بگیرند. با این کار محققان از افشا و آسیب به نمونه از دیتابیس جلوگیری می‌کنند. داشتن دسترسی نوشتن یکی از خطرناک‌ترین دسترسی‌های این دیتابیس‌ها می‌باشد به طوری که این ابزار علاوه عملیات گفته شده بالا یک استراتژی دیگری را در پیش می‌گیرد و آن این است که به جست و جوی DNS های آن به صورت غیر فعال می‌پردازد تا متوجه آن شود که آیا روی این IP که دیتابیس مستقر شده است، منابع دیگری مانند برنامه‌های وب و وبسایت‌ها و دیگر سرویس‌ها مستقر شده‌اند یا خیر؟ چرا که اگر منابع وب را از این طریق پیدا کند به این معنی است که این سرورها پتانسیل حمله آسیب‌زننده‌ای که باعث دستکاری داده‌ها می‌شود را دارند. در تمام وضعیت گفته شده بالا با ارائه دهندگان سرویس‌های ابری ارتباط برقرار شده و به آنها در مورد آسیب‌پذیری‌های یافت شده گزارشی به عمل آمده است.

۶۷ میلیون آدرس IP اسکن شده در ارائه دهندگان سرویس‌های ابری مختلف بین اکتبر سال ۲۰۱۹ تا مارچ ۲۰۲۰، تعداد ۱۲،۲۷۶ سرویس دیتابیسی با دسترسی‌های مختلف یافت شدند که ٪۸۷ آنها با دسترسی آزاد خواندن و نوشتن و ٪۸/۶ آنها تنها قابلیت خواندن اطلاعات را داشتند. بین این بررسی محققان مواردی از قابل دسترس بودن اطلاعات فقط خواندنی این دیتابیس‌ها پیدا کردند که ۷۴۲ نمونه پتناسیل افشای اطلاعات حساس کاربران مانند آدرس ایمیل، نام‌ها، گذرواژه‌ها و تمام

منابعی که می‌تواند در اپلیکیشن‌های وب آنها استفاده شود، را داشتند. علاوه‌بر این ما دیتابیس‌های مختلفی را پیدا کردیم که توانایی افشای فایل‌های مهم و حساس مانند فایل‌های سرتیفیکیت سایت‌ها و لاگ‌های مربوط به آنها را داشتند. بین تمام سیستم‌های DBM سرویس MongoDB بیشترین مقدار ضعف پیکربندی را داشت به گونه‌ای که ۴،۸۵۹ نمونه از آن یافت شد و این سهم برای دیتابیس Elasticsearch به مقدار ۴،۷۲۵ نمونه بود.

### ۳.۲.۲ مرور سناریوهای تهدیدآمیز

**افشای اطلاعات (سطح دسترسی خواندن)**

زمانی که منابع دیتابیسی به صورت غیر عامدانه‌ای مورد دسترسی عموم قرار می‌گیرد که موجب مسائل شکسته شدن حریم خصوصی کاربران و افشای اطلاعات حساس و عدم محرمانگی می‌شود.

**آلوده شدن منابع وب (سطح دسترسی نوشتن)**

زمانی که دسترسی نوشتن روی یک میزبان فعال باشد به معنای آن است که تمام محتوای آن میزبان را می‌توان دستکاری کرد. اغلب وب سایت‌ها به این ترتیب تغییر چهره روی آنها اعمال می‌شود که مربوط به عملیات دستکاری Deface کردن این پایگاه‌های اطلاعاتی است. همچنین این عمل باعث تاثیر روی محتوای این وب‌سایت‌ها خواهد شد چرا که می‌توانند وارد دیتابیس شده و اطلاعات مروبطه را دستکاری کنند و به نفع خودشان ویرایشی انجام دهند. همچنین آسیب‌پذیری‌های دیگر نیز می‌تواند رخ دهد. برای مثال بعد از دسترسی نوشتن روی این میزبان‌ها می‌توانند از طریق وب‌سایت یک فایل مخرب و آلوده را قرار داده و کاربران آن را به عنوان فایل مورد نظر بارگیری کرده و باعث آلوده شدن دستگاه کاربران نهایی شود.

### ۴.۲.۲ اخاذی در ازای اطلاعات

مهاجمان می‌توانند با داشتن دسترسی نوشتن روی این دیتابیس‌ها حمله‌ای انجام دهند که موجب اخاذی از صاحبان اطلاعات شود. معمولا استراتژی مهاجمان در این خصوص از بین بردن اطلاعات یا رمزنگاری آنها می‌باشد که در ازای اخاذی از صاحبان دیتابیس یا داده می‌توانند داده‌ها را به آنها برگردانند یا آنها کلید رمزنگاری آن داده‌ها را تحویل دهند.

محققان چهار تا از محبوب‌ترین دیتابیس‌های NoSQL را مورد بررسی قرار دادند تا نشان دهند که تحقیقات آنها کافی بوده و تقریبا مهم‌ترین سرویس‌های NoSQL را پوشش داده است. محققان تحقیقاتی را نسبت به محبوب‌ترین سیستم‌های دیتابیسی براساس وب سایت db-engines.com به عمل آوردند. مهم‌ترین سوال آن است که چگونه یک سیستم به عنوان محبوب‌ترین سیستم دیتابیسی انتخاب می‌شود؟

انتخاب این دیتابیس‌ها براساس درصد استفاده آنها در وبسایت‌ها، بحث و گفت و گوهای گروه‌های فنی، پیشنهادات شغلی در رابطه با متخصص مربوط به این دیتابیس‌ها و ارتباطشان در شبکه‌های اجتماعی می‌باشد. براساس جدول ۱، رنک دیتابیس‌های مختلف براساس سایت db-engines آمده است. لازم به ذکر است که این جدول نسبت به جدول داخل مقاله به روز شده که طی ۳ سال گذشته دیتابیس‌های Redis و Elasticsearch به ترتیب مقال ۶ و ۷ را بدست آوردند. در حالی که بین سال ۲۰۱۹ تا ۲۰۲۰ مقام Redis و Elasticsearch به ترتیب ۷ و ۸ بود.

### ۵.۲.۲ اهداف تحقیق

اهداف این تحقیقات به شرح زیر می‌باشد:

۱. نتیجه عدم تدابیر امنیتی

۲. بررسی تاثیر ضعف پیکربندی

۳. افزایش آگاهی برای جلوگیری از فاش شدن و دستکاری اطلاعات

همچنین در بخش‌های بعدی در مورد آگاهی از نگرانی‌های اخلاقی مطرح شده است که در آن به جمع‌آوری داده‌های نتیجه این آزمایشات صرفا برای بررسی محاسبات محققان نسبت به آسیب‌پذیری داده‌ها در آینده است.

جدول ۱: رنکینگ موتورهای دیتابیس: ۱۰ دیتابیس محبوب از نظر سایت db-engines

| رنک | | | دیتابیس | مدل | امتیاز | | |
|---|---|---|---|---|---|---|---|
| ۲۰۲۳ نوامبر | ۲۰۲۳ اکتبر | ۲۰۲۲ نوامبر | | | ۲۰۲۳ نوامبر | ۲۰۲۳ اکتبر | ۲۰۲۲ نوامبر |
| ۱ | ۱ | ۱ | Oracle | R | 1277.03 | +15.61 | +35.34 |
| ۲ | ۲ | ۲ | MySQL | R | 1115.24 | −18.07 | −90.30 |
| ۳ | ۳ | ۳ | MSSQL Server | R | 911.42 | +14.54 | −1.09 |
| ۴ | ۴ | ۴ | PostgreSQL | R | 636.86 | −1.96 | +13.70 |
| ۵ | ۵ | ۵ | MongoDB | NS | 428.55 | −2.87 | −49.35 |
| ۶ | ۶ | ۶ | Redis | NS | 160.02 | −2.95 | −22.03 |
| ۷ | ۷ | ۷ | Elasticsearch | NS | 139.62 | +2.48 | −10.70 |
| ۸ | ۸ | ۸ | IBM Db۲ | R | 139.62 | +2.48 | −10.70 |
| ۹ | ۹ | ۱۰ | SQLite | R | 124.58 | +1.13 | −13.56 |
| ۱۰ | ۱۰ | ۹ | Microsoft Access | R | 124.49 | +0.18 | −10.53 |

# ۳ مدل پیشنهادی

## ۱.۳ جمع‌آوری داده با پیدا کردن آدرس‌های IP سرویس دهندگان

اولین مرحله از این رویکرد جمع‌آوری داده با استفاده از لیست آدرس‌های IP نسخه ۴ بوده است که می‌تواند امکان داشته باشد روی هر کدام از آدرس‌ها حداقل یک نمونه دیتابیس NoSQL وجود داشته باشد. همچنین محققان لیستی از ارائه دهندگان سرویس‌های ابری را مطرح کردند که در آنها امکان نصب و راه‌اندازی این نوع دیتابیس‌ها میسر بوده است. این ارائه دهندگان به کاربران اجازه می‌دهند تا سرویس‌های دیتابیسی خود را در شبکه اینترنت مستقر کنند و یک Connection String معتبر برای دسترسی آنها به اپلیکیشن خود راه‌اندازی نمایند.

این سرویس‌ها عبارت‌اند از:

- AmazonEC۲
- Microsoft Azure Cloud
- Goolge Cloud
- Tencent Cloud
- DigitalOcean
- OVH

هر کدام از نمونه ارائه‌دهندگان سرویسی که در بالا نام برده شد، قابلیت آن را دارند که کاربر بتواند در آن به صورت دستی دیتابیس NoSQL مورد نظر خود را نصب و پیکربندی کند. اما باید توجه داشت بعد از نصب اولیه این دیتابیس‌ها

روی این سرویس‌ها هیچ پیکربندی در رابطه با کنترل دسترسی و تغییر شماره پورت و غیره انجام نمی‌شود که این به خودی خود نشان دهنده پیکربندی ضعیف این دیتابیس‌ها می‌باشد. اکثر تقاضا برای راه‌اندازی اولیه این دیتابیس‌ها صرفا جهت داشتن فرایند آزمایشی توسط هر توسعه دهنده تازه کار است. دیگر به آن روند مهم پیکربندی توجه نمی‌کند و بعد از آن ممکن است داده‌های مهمی را بدون توجه به ضعیف بودن پیکربندی در دیتابیس خودش انتقال دهد.

بین شش سرویس ابری بالا، از سه مورد آنها (AmazonEC2، Microsoft Azure Cloud، Google Cloud) محققان توانستند به subnet آدرس پابلیک IP برسند. اما سه سرویس آخر یعنی (Tencent Cloud، DigitalOcean، OVH) آدرس IP پابلیک خود را ارائه نمی‌داند. برای دریافت اطلاعات مورد نظر محققان آدرس‌های IP را از ipinfo.io بدست آوردند و سپس بعد از توانستند به زیر آدرس‌های شبکه مورد نظر دسترسی پیدا کنند و برنامه خود را براساس لیست بدست آمده اجرا کرده و پیکربندی دیتابیس‌های مطرح شده را بررسی و بعد از آن گزارشی را تهیه کنند.

## ۲.۳ شناسایی نمونه‌های افشا شده

در مرحله دوم رویکرد، محققان موضوع پیدا کردن دیتابیس افشا شده در فضای عمومی اینترنت را بررسی کردند. با استفاده از لیست آدرس‌هایی که در مرحله اول بدست آمده است، محققان با به کارگیری Nmap آدرس‌های IP را برای یافت پورت‌های باز سرویس NoSQL بررسی می‌کنند. در نتیجه این مرحله دو حالت به وجود می‌آید:

۱. نتیجه Close را نمایش می‌دهد. این بدان معناست که سرور توسط آدرس IP در دسترس بوده است اما با استفاده از پورت پیش فرض دیتابی، برنامه محققان نتوانسته است به آن سرویس NoSQL متصل شود.

۲. نتیجه Filtered را نمایش می‌دهد. این بدان معناست که آدرس سرور و شماره پورت پیش فرض که این شماره پورت مربوط به یکی از سرویس NoSQL است بر روی آن هیچ سرویس دیتابیسی NoSQL مستقر نشده است و بجای آن اپلیکیشن‌های دیگری از این شماره پورت پیش فرض استفاده می‌کنند.

پس در این مرحله می‌توان نتیجه گرفت که برنامه محققان تمام آدرس‌هایی را بررسی کردند که بر روی شماره پورت پیش فرض آن‌ها یکی از دیتابیس‌های NoSQL مستقر شده است تا بتوانند ضعف پیکربندی آن‌ها را در این مقاله بیشتر مورد بررسی قرار دهند.

## ۳.۳ بررسی‌های امنیتی

فاز اصلی و عملی این تحقیق می‌باشد. به گونه‌ای که به سه وظیفه برای جست و جوی نمونه‌های باز دیتابیس‌های NoSQL انجام گرفته است:

۱. ساخت یک نمونه یا Instance از دیتابیس مربوطه برای اتصال به آن و بررسی‌های اولیه ورود بدون احراز هویت

۲. بررسی و ورود به داده‌ها برای دریافت اطلاعات بدون دسترسی به محتوای اصلی آنها برای اثبات انتشار داده‌های حساس کاربران

۳. در معرض دید عموم قرار گرفتن سرویس‌های وب

### بررسی و نمونه‌گیری از دیتابیس جهت اتصال و انجام عملیات

برای هر یک از آدرس‌های IP که در مرحله قبل بدست آورده شد، این فریمورک یک نمونه به ازای هر سرویس NoSQL ایجاد می‌کند تا ضعف پیکربندی را مورد بررسی قرار دهد.

در ابتدا این ابزار در دیتابیس‌های مذکور لاگین کرده و سطوح دسترسی [۴] آنها را مورد بررسی قرار می‌دهد. اگر این ابزار نتواند به صورت مستقیم مجوزهای خواندن و نوشتن کاربران تعریف شده در دیتابیس را دریافت کند وارد استراتژی دوم شده و یک workspace جدید در دیتابیس افشا شده ایجاد می‌کند. اگر نتیجه ایجاد این workspace موفقیت آمیز بود این بدین معناست که این دیتابیس دسترسی نوشتن را برای تمام کاربران چه داخلی و چه خارج از سیستم دارد. بعد از این ایجاد این workspace ابزار سعی می‌کند که یک پیام هشدار برای صاحبان دیتابیس ایجاد کند در این پیام در مورد پیکربندی اشتباه و ضعیف این نمونه توضیح خواهد داد و صاحبان دیتابیس را با این پروژه آشنا می‌کند که قصد هیچ تخریب اطلاعاتی ندارد بلکه می‌خواهد آنها را از بابت این نقص‌ها با خبر سازد که هر چه زودتر با سیاستی مناسب از این دسترسی‌ها به کاربران خارج از سیستم جلوگیری انجام شود.

### بررسی نشت داده

برای بررسی نشت اطلاعات، اگر ابزار بتواند بررسی کند که دسترسی خواندن را دارد که آن را اعلام می‌کند در غیر این صورت به صورت مستقیم به دیتابیس متصل شده و تمام متا دیتاها از قبیل ورژن کنونی دیتابیس، نقش‌های کاربران تعریف شده در دیتابیس و همچنین نام تمام تسون‌ها (ویژگی‌های داده) را دریافت می‌کنند. لازم به ذکر است که این ابزار این داده‌ها را با استفاده از توابع داخلی دیتابیس مانند استفاده از Regular Expresion توابع شمارشی و غیر بدست می‌آورد که هیچ داده‌ای از کاربران را در ابزار تجزیه و تحلیل نکند. این داده‌ها از قبیل آدرس ایمیل، حساب‌های شبکه‌های اجتماعی، شماره تلفن‌ها شماره حساب‌ها و گذرواژه هستند. این نتیجه مقادیر این داده‌ها را می‌توانید در جدول ۲ مشاهده کنید.

### بررسی در معرض دید قرار گرفتن سرویس‌های وب

در این مرحله پتانسیل احتمال آسیب پذیری سرور افشا شده در برابر حمله‌های مبتنی بر وب مورد بررسی قرار گرفته است. در عمل با استفاده از ابزاری به نام VirusTotal آدرس‌های IP با دسترسی عمومی را وارد این برنامه کرده و تمام وب سایت‌هایی که روی این IP مستقر شده اند را بر می‌گرداند. در نهایت محققان به این نتیجه رسیدند که ممکن است بر روی نمونه دیتابیس NoSQL مستقر شده روی یک IP با دسترسی عمومی، وب سایت مربوط به این نمونه‌ها نیز مستقر شده باشد. اگر وب سایت به صورت عمومی از این روش در دسترس همه افراد باشد (با ارسال درخواست به سمت این وب سایت‌ها

[۴] Access permission

وضعیت درخواست شما ۲۰۰ خواهد بود.) محققان به این نتیجه رسیدند که می‌توانند از طریق این اتصال بین وب سایت و دیتابیس به منابع دیگر آنها دسترسی داشته باشند.

جدول ۲: آنالیز نشت محتوای حساس

| Category | FileType | MongoDB | Elasticsearch | cassandra | Total No. |
|---|---|---|---|---|---|
| Text/Data | .log | ۳۵۰.۲۴۳ | ۶۹.۹۶۵.۶۱۳ | ۳.۳۲۲ | ۷۰.۳۱۹.۱۷۸ |
| | .zip | ۸.۷۹۰.۳۰۱ | ۱۸۱.۷۴۵ | ۳۳۵ | ۸.۹۷۲.۳۸۱ |
| | .json | ۷.۰۶۸ | ۵.۶۱۲.۰۷۲ | ۷.۴۱۳ | ۵.۶۸۹.۵۵۳ |
| | .xml | ۲.۰۴۰.۵۳۳ | ۹۸۱.۲۳۳ | ۴.۴۸۰ | ۳.۰۲۶.۲۴۶ |
| | .txt | ۱۵۰.۳۷۹ | ۱.۴۵۹.۴۹۵ | ۱.۷۳۳ | ۱.۶۱۱.۶۰۷ |
| | .pdf | ۸۱۱.۰۰۳ | ۷۵۶.۵۱۸ | ۱۰.۰۹۶ | ۱.۵۷۷.۶۱۷ |
| | .text | ۲۷.۲۱۹ | ۴۴۳.۹۹۰ | ۱۳۲ | ۴۷۱.۳۴۱ |
| | .gz | ۶.۴۱۶ | ۱۹۱.۳۸۷ | ۱۳۲ | ۱۴.۴۲۶ |
| | .docx | ۱۸.۹۱۰ | ۴۵.۳۷۵ | ۸۲۴ | ۶۵.۱۰۹ |
| Image/Video | .jpg | ۲۲.۷۷۲.۶۲۳ | ۲۵.۲۶۷.۹۳۴ | ۷۳۹.۵۰۸ | ۴۸.۷۸۰.۰۶۵ |
| | .png | ۴.۵۵۸.۳۷۲ | ۶.۰۷۲.۵۵۸ | ۱۱۲.۵۶۱ | ۱۰.۷۴۳.۴۹۱ |
| | .jpeg | ۶۱۳.۲۰۲ | ۷۰۷.۴۶۱ | ۳۴.۳۴۸ | ۱.۳۵۵.۱۲۱ |
| | .gif | ۵۹۴.۴۲۸ | ۵۸۱.۳۸۲ | ۱.۹۲۶ | ۱.۳۳۵.۱۲۱ |
| | .mp۴ | ۴۲۹.۰۳۸ | ۴۷۰.۲۳۱ | ۴.۳۳۳ | ۹۰۳.۶۰۲ |
| | .webp | ۳۹.۹۹۴ | ۷۵.۱۶۳ | ۲۴۹.۶۰۹ | ۳۶۴.۷۶۶ |
| Code | .html | ۱.۶۱۸.۰۱۰ | ۱۱.۶۸۴.۱۸۳ | ۲۷۲.۸۸۳ | ۱۳.۵۷۵.۰۷۶ |
| | .ts | ۸.۳۸۳ | ۳۳۵.۵۳۹ | ۱۱.۱۳۲ | ۳۵۵.۰۵۴ |
| | .js | ۵۸.۲۱۳ | ۱۳۵.۱۶۷ | ۱.۸۹۸ | ۱۹۵.۲۷۸ |
| | .hll | ۴۷ | ۵۸۵ | ۰ | ۶۳۲ |
| File Keys | .key | ۱۰.۴۴۴ | ۱۰.۸۳۲.۴۶۴ | ۲.۶۲۰ | ۱۰.۸۴۵.۵۲۸ |
| Crypto Files | .pem | ۲۴۷ | ۱۳۰.۳۷۰ | ۱۳ | ۱۳۰.۶۳۰ |
| | .pfx | ۸۹ | ۱.۲۵۸ | ۴ | ۱.۳۵۱ |
| | .p۱۲ | ۳۱ | ۴۵۷ | ۷ | ۴۹۵ |
| DB | .sql(Dumlps) | ۲.۲۷۰ | ۲۳۷.۳۰۶ | ۱۸۶ | ۲۳۹.۷۶۲ |
| Backup | .bak | ۱.۴۸۷ | ۵.۳۶۴ | ۳۰۰ | ۷.۱۵۱ |
| Password | .kbd(Keepass) | ۱۰ | ۴۲ | ۰ | ۵۲ |

# ۴ آزمایش‌ها و تحلیل نتایج

در این قسمت به نحوه عملکرد کلی فریمورک توسعه داده شده توسط محققان می‌پردازیم. همان طور که بالاتر بارها اشاره شد، این فریمورک طی تمام آزمایش‌ها، محققان اصلا ماهیت داده‌های دیتابیس و محتوای آنها را مورد بررسی قرار نداده بلکه تماما روی دسترسی در استفاده از داده‌ها تاکید زیاد داشتند.

در این بخش با نمایش یک نمونه کد، به طور کلی در مورد نحوه عملکرد این فریمورک صحبت خواهیم کرد.

```
۱        # Connect to the database
۲        mongo_client.connect(ip_address)
۳        # Get database names
۴        db_names = list_database_name()
۵        # Mongo-shell method used to get role mappings
۶        db.getRoles(showBuiltinRoles: false)
۷        # Get collection names
۸        collections = list_collection_names()
۹        # Map-reduce function used to get field-mapping for each table
۱۰       map = Code("function() {
۱۱           for (key in this) {
۱۲               emit(key, null);
۱۳           }
۱۴       }")
۱۵       reduce = Code("function(key, stuff) { return null; }")
۱۶       results = collection.map_reduce(map, reduce, "myresults")
۱۷       # Query methods for field and object detection
۱۸       count({ fieldname: {"$exists": True} })
۱۹       count({ fieldname: {r'.*\.(fieldvalue)'} })
۲۰       # Write the message in a new database
۲۱       writedb = mongo_client[MONGO_DB]
۲۲       writecollection = writedb[MONGO_COLLECTION]
۲۳       writecollection.insert_one(MONGO_DOC)
```

## بررسی قسمتی از فرایند مطرح شده

در خط ۲ با استفاده از آدرس IP حاضر در لیست آدرس‌ها، قصد اتصال به دیتابیس مانگو را به صورت ریموت داریم. سپس بعد از اتصال موفق می‌توانیم با استفاده از واسط Mongo Shell دستوراتی را برای انجام عملیاتی روی دیتابیس مورد نظر انجام دهیم. برای مثال در خط بعدی آن نام تمام دیتابیس‌هایی که در این آدرس IP وجود دارد را دریافت می‌کنیم و در متغیر dbnames قرار می‌دهیم. با استفاده از این عمل یعنی عاملی که توانسته به دیتابیس متصل شود قادر به خواندن محتوا بوده است که در این جا با اطمینان اعلام می‌کنیم که دسترسی خواندن در این دیتابیس برای یک کاربر عادی و خارج از سیستم وجود دارد. بعد از آن با استفاده از متد getRoles می‌توانیم تمام کاربرانی که در این دیتابیس تعریف شده‌اند براساس سطح دسترسی آنها، را دریافت کنیم. برای دسترسی به جداول (اصطلاحا در دیتابیس‌های مانگو به آن کالکشن می‌گویند.) از تابع که نوشته‌ایم استفاده کردیم تا لیست تمام کالکشن‌های مروبط به یک دیتابیس را دریافت کنیم. برای آن که بتوانیم فیلدهای (کلیدهای) مربوط به یک کالکشن (مجموعه‌ای از داکیومنت‌ها) را دریافت کنیم از یک تابع Map-Reduce استفاده کردیم که تنها نام کلیدهای استفاده شده در این کالکشن را به ما برگرداند که بدایم دیتابیس شامل چه فیلدهایی است. در نهایت با استفاده از Regex توانستیم تمام داده‌های مربوط به متادیتا هر کالکشن را توسط الگوی "exists" برای تشخیص تعداد فیلدها جست و جو کنیم. در سه خط کد آخر می‌توان متوجه شد که ما می‌خواستیم که از داشتن دسترسی نوشتن در دیتابیس

اطمینان حاصل کنیم که به همین خاطر سعی کردیم نتیجه تحقیقات خود را در یک کالکشن جدید در همان دیتابیس به عنوان یک داکیومنت جدید اضافه کنیم.

## یادآوری تابع Map-Reduce

تابع Map-Reduce یک تابع دلخواه و قابل سفارشی‌سازی توسط برنامه‌نویس در زبان جاوا اسکریپت است (اساس کلی دیتابیس‌های مانگو بر پایه موتور NodeJS و زبان Javascript می‌باشد.) که به واسطه آن عملیات مختلفی مانند بدست آوردن کلیدهای استفاده شده در یک داکیومنت، گروه‌بندی اجزا و یا دستکاری روی گروه‌بندی اجزا با استفاده از توابعی مانند sum() یا count() می‌باشد.

```
db.cardata.find()
{ "_id": ObjectId("9dd74a***2340***"), "name": "Peugeot", "Series": "2", "qty": 10 },
{ "_id": ObjectId("5f94********"), "name": "BMW", "Series": "X", "qty": 112 },
{ "_id": ObjectId("2e127a********"), "name": "Peugeot", "Series": "5", "qty": 105 },
{ "_id": ObjectId("8m113********"), "name": "BMW", "Series": "M", "qty": 25 },

var map = function() { emit(this.name, this.qty) }
var reduce = function(key, value) { return Array.sum(values) }
db.cardata.mapreduce(map, reduce, "myResults")
db.myResults.find()
{ "_id": "BMW", "value": 137 },
{ "_id": "Peugeot", "value": 115 }
```

## بررسی در انواع دیتابیس‌های اشاره شده

دقیقا همانند کد بالا در دو دیتابیس Cassandra و Elasticsearch به شکل مناسب برای انجام همان فرایندها، عملیات مورد نظر را اعمال کردیم اما در دیتابیس مموری Redis به شکل قبلی نتوانستیم عمل کنیم چرا که نمی‌توانستیم به طور مستقیم برخلاف سرویس‌های قبلی صحت دسترسی به لیست‌ها و کالکشن‌ها را بررسی کنیم. (این ویژگی در نسخه ۶ به بعد معرفی شد به گونه‌ای که در این مقاله امکانش میسر نبوده است). همچنین از آنجایی که در دیتابیس Redis نمی‌توان تعداد آبجکت‌ها و فیلدها را بدست آورد (به دلیل key:value بودن) از انجام این عملیات جلوگیری کرده‌ایم چرا که برای بدست آوردن هر فیلد ممکن بود که بتوانیم به مقدار آن فیلد دسترسی داشته باشیم که این قوانین ما را نقض می‌کند. در نهایت برای عدم آسیب رساندن به بقیه داده‌ها سعی کردیم یک کلید جدید ایجاد کنیم و پیام هشدار خود را در آن ایجاد کنیم. در زیر می‌توانید عملیات (کلی) انجام شده در دیتابیس Redis را بررسی کنید.

```
# Connect to the database
r = redis.Redis(ip_address)
# Get key names
keys = r.keys(pattern=u'*')
# Get eventual number of occurrences of key fieldnames
occurrences = scan_iter(match="fieldname")
for match in occurrences:
    counter += 1
# Write the warning message
r.append(key, value)
```

## ۵ نوآوری‌های تحقیق

## ۶ بخش‌های باقی مانده