

ШИНЖЛЭХ УХААН, ТЕХНОЛОГИЙН ИХ СУРГУУЛЬ
Мэдээлэл, Холбооны Технологийн Сургууль



Амгаланбаатарын Мягмарцэрэн

Прокси дахин шифрлэх схемийн
туршилтын системийг хөгжүүлэх нь

БАКАЛАВРЫН ТӨГСӨЛТИЙН АЖИЛ

Улаанбаатар хот

ШИНЖЛЭХ УХААН, ТЕХНОЛОГИЙН ИХ СУРГУУЛЬ
МЭДЭЭЛЭЛ, ХОЛБООНЫ ТЕХНОЛОГИЙН СУРГУУЛЬ

Мэдээллийн сүлжээ, аюулгүй байдлын салбар

Прокси дахин шифрлэх схемийн
туршилтын системийг хөгжүүлэх нь

Мэргэжлийн индекс: D061940

Мэргэжил: Мэдээллийн системийн аюулгүй байдал

Удирдагч: доктор (Ph.D) В.Нямсүрэн

Зөвлөгч: доктор (Ph.D), В.Нямсүрэн
магистр Ц.Манлайбаатар

Гүйцэтгэгч: А.Мягмарцэрэн

Улаанбаатар хот

2023 он 6 сар

Батлав. Мэдээллийн сүлжээ, аюулгүй байдлын салбарын эрхлэгч:

..... /доктор (Ph.D) Б.Мөнхбаяр/

Удирдагч: /доктор (Ph.D) В.Нямсүрэн/

ДИПЛОМЫН ТӨСӨЛ ГҮЙЦЭТГЭХ ТӨЛӨВЛӨГӨӨ

Дипломын төслийн сэдэв:

Монгол: ” Прокси дахин шифрлэх схемийн туршилтын системийг хөгжүүлэх нь”

Англи: ” Developing Prototype System of Proxy Re-Encryption Scheme”

Төслийн зорилго: Proxy Re-Encryption схемийн хэрэглээнүүдийг судалж, нэгэн хэрэглээг хэрэгжүүлэх туршилтын систем хөгжүүлэх

Гүйцэтгэх оюутны овог нэр:

А.Мягмарцэрэн/В190970106/

Холбоо барих утас:

99754252

№	Ажлын бүлэг, хэсгийн нэр	эзлэх хувь	дуусах хугацаа
Бүлэг №1. Өгөгдөл хуваалцах үйлчилгээний тухай			
1	1.1 Өгөгдөл хуваалцах үйлчилгээний тухай 1.2 Өгөгдлийн аюулгүй байдал 1.3 Шифрлэх схемүүд 1.4 Файл шифрлэх аргууд	20%	
Бүлэг №2. Прокси дахин шифрлэлтэд суурилсан файл хуваалцах систем			
2	2.1 Прокси дахин шифрлэлт 2.2 Хөгжүүлэх технологи, хэл сонгох 2.3 Хөгжүүлэлтийн орчин бэлдэх	40%	
Бүлэг №3. Прокси дахин шифрлэлтэд суурилсан файл хуваалцах систем хөгжүүлэх			
3	3.1 Системийн шаардлага 3.2 Системийн загвар 3.3 Системийн хөгжүүлэх 3.4 Файл хуваалцах системийг турших	40%	
Бүлэг №4. Ерөнхий дүгнэлт			

Төлөвлөгөөг боловсруулсан оюутан: /А.Мягмарцэрэн/

ТӨГСӨЛТИЙН АЖЛЫН ҮЗЛЭГИЙН ХУУДАС

Оюутны код: B190970106

Оюутны нэр: А.Мягмарцэрэн

Сэдвийн монгол нэр: ” Прокси дахин шифрлэх схемийн туршилтын системийг хөгжүүлэх нь”

Сэдвийн англи нэр: ” Developing Prototype System of Proxy Re-Encryption Scheme”

Удирдагч багш: доктор (Ph.D) В.Нямсүрэн

Зөвлөгч багш: доктор (Ph.D), В.Нямсүрэн, магистр Ц.Манлайбаатар

№	Үзлэгийн гүйцэтгэл	Гүйцэтгэлийн 30% -с багагүй байна.	Огноо	Удирдагч доктор (Ph.D) В.Нямсүрэн багшийн гарын үсэг
1	Үзлэг-1		IV/03-IV/07	

Багшийн товч зөвлөгөө, тайлбар:

.....

.....

.....

.....

.....

.....

.....

Үзлэг-1 хийсэн багш: /доктор (Ph.D) В.Нямсүрэн/

№	Үзлэгийн гүйцэтгэл	Авсан оноо (10 оноо)	Гүйцэтгэлийн 50% -с багагүй байна.	Огноо	доктор (Ph.D), В.Нямсүрэн багшийн гарын үсэг
1	Үзлэг-2			IV/17-IV/21	

Багшийн товч зөвлөгөө, тайлбар:

.....

.....

.....

.....

.....

.....

.....

Үзлэг-2 хийсэн багш: /доктор (Ph.D), В.Нямсүрэн/

ТӨГСӨЛТИЙН АЖЛЫН ҮЗЛЭГИЙН ХУУДАС

Оюутны код: B190970106

Оюутны нэр: А.Мягмарцэрэн

Сэдвийн монгол нэр: ” Прокси дахин шифрлэх схемийн туршилтын системийг хөгжүүлэх нь”

Сэдвийн англи нэр: ” Developing Prototype System of Proxy Re-Encryption Scheme”

Удирдагч багш: доктор (Ph.D) В.Нямсүрэн

Зөвлөгч багш: доктор (Ph.D), В.Нямсүрэн, магистр Ц.Манлайбаатар

№	Үзлэгийн гүйцэтгэл	Авсан оноо (10 оноо)	Гүйцэтгэлийн 70% -с багагүй байна.	Огноо	магистр Ц.Манлайбаатар багшийн гарын үсэг
1	Үзлэг-3			V/08-V/12	

Багшийн товч зөвлөгөө, тайлбар:

.....

.....

.....

.....

.....

.....

Үзлэг-3 хийсэн багш: /магистр Ц.Манлайбаатар/

№	Үзлэгийн гүйцэтгэл	Гүйцэтгэлийн 90% -с багагүй байна.	Огноо	Удирдагч доктор (Ph.D) В.Нямсүрэн багшийн гарын үсэг
1	Үзлэг-4		V/15-V/19	

№	Удирдагч доктор (Ph.D) В.Нямсүрэн багшийн үнэлгээ (30 оноо)	Огноо	Удирдагч багшийн гарын үсэг
1		V/17	

Удирдагч багш: /доктор (Ph.D) В.Нямсүрэн/

Жич: Удирдагч багш өөрийн үнэлгээгээ 30 хүртэл оноогоор өгөх ба үнэлгээ тавьсан хуудсыг оюутанд буцааж өгөлгүй төгсөлтийн нарийн бичгийн даргад хураалгана уу.

ТӨГСӨЛТИЙН АЖЛЫН ЯВЦ

№	Хийж гүйцэтгэсэн ажил	Биелсэн хугацаа	Удирдагчийн гарын үсэг
1	Бүлэг №1. Прокси дахин шифрлэх схемийн онолын хэсэг	2023-5-28	
2	Бүлэг №2. Серверт шифрлэгдсэн файл хуваалцах судалгаа	2023-5-28	
3	Бүлэг №3. Прокси дахин шифрлэх систем хөгжүүлэх	2023-5-28	
4	Бүлэг №4. Ерөнхий дүгнэлт	2023-5-28	

Ажлын товч дүгнэлт

.....
.....
.....
.....
.....
.....
.....

Удирдагч: /доктор (Ph.D) В.Нямсүрэн/

ЗӨВШӨӨРӨЛ

Оюутан А.Мягмарцэрэн–н бичсэн төгсөлтийн ажлыг УШК-д хамгаалуулахаар тодорхойлов.

Салбарын эрхлэгч: /доктор (Ph.D) Б.Мөнхбаяр/

ШИНЖЛЭХ УХААН, ТЕХНОЛОГИЙН ИХ СУРГУУЛЬ
Мэдээлэл, Холбооны Технологийн Сургууль

ШҮҮМЖИЙН ХУУДАС

Мэдээллийн сүлжээ, аюулгүй байдлын салбар–н салбарын төгсөх курсийн оюу-
тан А.Мягмарцэрэн-н "Прокси дахин шифрлэх схемийн туршилтын системийг хөг-
жүүлэх нь" сэдэвт төгсөлтийн ажлын шүүмж.

1. Төслөөр дэвшүүлсэн асуудал, үүнтэй холбоотой онолын материал уншиж су-
далсан байдал. Энэ талаар хүмүүсийн хийсэн судалгаа, түүний үр дүнг ун-
шиж тусгасан эсэх.

.....
.....
.....
.....
.....
.....
.....

2. Төслийн ерөнхий агуулга. Шийдсэн зүйлүүд, хүрсэн үр дүн. Өөрийн санааг
гарган, харьцуулалт хийн, дүгнэж байгаа чадвар.

.....
.....
.....
.....
.....
.....
.....

3. Эмх цэгцтэй, стандарт хангасан өөрөөр хэлбэл диплом бичих шаардлагуудыг
биелүүлсэн эсэх. Төсөлд анзаарагдсан алдаанууд, зөв бичгийн болон өгүүл-
бэр зүйн гэх мэт /Хуудас дугаарлагдаагүй, зураг хүснэгтийн дугаар болон
тайлбар байхгүй, шрифт хольсон, хувилсан зүйл ихээр оруулсан/.

.....
.....
.....
.....
.....

4. Төслөөр орхигдуулсан болон дутуу болсон зүйлүүд. Цаашид анхаарах хэрэгтэй зүйлүүд.

.....

.....

.....

.....

.....

.....

.....

5. Төслийн талаар онцолж тэмдэглэх зүйлүүд.

.....

.....

.....

.....

.....

.....

.....

6. Ерөнхий оноо. (30 оноо)

.....

Шүүмж бичсэн: /магистр Ц.Манлайбаатар/

Ажлын газар:

Хаяг (Утас)

Зохиогчийн эрх хамгаалал

Миний бие А.Мягмарцэрэн, "Прокси дахин шифрлэх схемийн туршилтын системийг хөгжүүлэх нь" сэдэвт энэ ажил нь минийх бөгөөд дараахыг нотолж байна. Үүнд:

- Горилогч энэ ажлыг тус сургуулиас боловсролын зэрэг авахаар бүхэлд нь буюу голлон хийсэн болно.
- Энэ ажлын аль нэг хэсгийг тус сургуульд эсвэл өөр байгууллагад боловсролын зэрэг, мэргэшил авахаар өмнө нь илгээсэн бол түүнийгээ тодорхой заасан болно.
- Бусад хүмүүсийн хэвлүүлсэн ажлаас зөвлөгөө авсан бол түүнийгээ үндэслэсэн болно.
- Бусад хүмүүсийн ажлаас ишлэл хийхдээ эх үүсвэрийг нь заасан болно.
- Миний ажилд тусалсан голлох бүх эх үүсвэрт талархаж байна.
- Ажлыг бусадтай хамтарсан бол алийг нь бусад хүмүүс хийсэн болохыг тодорхой заасан болно.

Гарын үсэг: _____

Огноо: _____

Зорилго

Прокси дахин шифрлэлэх схем судалж, аюулгүй файл хуваалцах систем туршилтын загвар хөгжүүлэх.

Зорилт

Дээрх зорилгыг хэрэгжүүлэхийн тулд дараах зорилтуудыг дэвшүүлж байна.

- Шифрлэлтийн схемүүдийг судлах
- Файл шифрлэх аргуудыг судлах
- Хөгжүүлэхэд шаардлагатай хэрэглэгдэхүүнийг судлах
- Аюулгүй файл хуваалцах үйлчилгээний загвар гарах
- Хөгжүүлсэн системийг туршиж, ажиллуулах

ШИНЖЛЭХ УХААН, ТЕХНОЛОГИЙН ИХ СУРГУУЛЬ
Мэдээлэл, Холбооны Технологийн Сургууль

Хураангуй

Прокси дахин шифрлэх схемийн туршилтын системийг
хөгжүүлэх нь

А.Мягмарцэрэн
b190970106@must.edu.com

Түлхүүр үгс: мэдээллийн аюулгүй байдал, прокси дахин шифрлэлт

Прокси дахин шифрлэх схем нь өгөгдөл эзэмшигч өөрийн нийтийн түлхүүрээр шифрлэгдсэн итгэмжлэгдсэн хүнд шифрийг тайлахгүйгээр гурав дахь ч хагас итгэмжлэгдсэн тал шифрийг дахин шифрлэж итгэмжлэгдсэн хүн тайлах боломжийг олдог.

Прокси дахин шифрлэх схемийг ашиглан файл хуваалцах туршилтын систем хөгжүүлэлтийг хийж гүйцэтгэв.

Талархал

Энэхүү дипломын ажлыг бичихэд туслалцаа үзүүлсэн удирдагч багш Доктор(Ph.D) В.Нямсүрэн болон зөвлөх багш Доктор(Ph.D) Ц.Энхтөр, Магистр Манлайбаатар багш нартаа болон Мэдээлэл сүлжээ, аюулгүй байдлын салбарын багш нарт талархсанаа илэрхийлье.

Товчилсон үгс

PRE	P roxy R e- E ncryption
BBS	B laze B leumer S trauss
IBE	I ntity B ased E ncryption
ABE	A tttribute B ased S ncryption
HE	H omomorphic E ncryption
API	A pplication P rogramming I nterface
PKG	P rivate K ey G enerator
GUI	G raphical U ser I nterface

Гарчиг

Зохиогчийн эрх хамгаалал	i
Хураангуй	iii
Талархал	iv
Товчилсон үгс	v
1 Онолын хэсэг	1
1.1 Өгөгдөл хуваалцах үйлчилгээ	1
1.2 Өгөгдлийн аюулгүй байдал	2
1.3 Шифрлэх схемүүд	4
1.4 Файл шифрлэх хадгалах	7
1.5 Бүлгийн Дүгнэлт	9
2 Судалгааны хэсэг	10
2.1 Прокси дахин шифрлэлт	10
2.2 Хөгжүүлэх технологи, хэл сонгох	12
2.3 Хөгжүүлэлтийн орчин бэлдэх	14
2.4 Бүлгийн дүгнэлт	18
3 Хэрэгжүүлэлтын хэсэг	19
3.1 Системийн шаардлага	19
3.2 Системийн загвар	22
3.3 Системийн хөгжүүлэх	25
3.4 Файл хуваалцах системийг турших	28
3.5 Дүгнэлт	31
4 Ерөнхий дүгнэлт	32
Дүгнэлт	32

Зургийн жагсаалт

1.1	Танилтад суурилсан шифрлэлт	4
1.2	Танилтад суурилсан шифрлэлт КР-АВЕ	5
1.3	Танилтад суурилсан шифрлэлт СР-АВЕ	6
2.1	Прокси дахин шифрлэх схем	11
2.2	pyUmbra1	13
2.3	Модулийн файлын бүтэц	15
3.1	Usecase diagram	20
3.2	Ерөнхий загвар	22
3.3	Файл эзэмшигчийн үйл ажилгааны диаграмм	23
3.4	Файл хэрэглэгчийн үйл ажилгааны диаграмм	23
3.5	Өгөгдлийн сангийн бүтэц	24
3.6	API сервисийн файлын бүтэц	25
3.7	Десктоп файлын бүтэц	26
3.8	Figma загвар	26
3.9	Нэвтрэх цонх	27
3.10	Бүртгүүлэх цонх	27
3.11	Үндсэн цонх	28
3.12	Бүртгүүлэх цонх жишээ	28
3.13	Жишээ өгөгдөл үүсгэнэ	29
3.14	Байршуулсан файл	29
3.15	Хуваалцах цонх	30
3.16	Хуваалцсан файл	30
3.17	Хуваалцсан файл шифрлэлтийг тайлсаны дараа	31
3.18	Wireshark барисан пакет	31

Хүснэгтийн жагсаалт

1.1	Текст файл шифрлэх гүйцэтгэл	7
1.2	Текст файлын шифрийг тайлах гүйцэтгэл	8
1.3	Бинари файл шифрлэх гүйцэтгэл	8
1.4	Бинари файлын шифрийг тайлах гүйцэтгэл	8
3.1	Бүртгүүлэх юзкейсийн тодорхойлолт	20
3.2	Нэвтрэх юзкейсийн тодорхойлолт	20
3.3	Файл шифрлэх юзкейсийн тодорхойлолт	21
3.4	Шифрлэсэн файл серверт байршуулах юзкейсийн тодорхойлолт . . .	21
3.5	Файл хуваалцах юзкейсийн тодорхойлолт	21
3.6	Файл татах юзкейсийн тодорхойлолт	21
3.7	Шифрлэсэн файлын тайлах юзкейсийн тодорхойлолт	21

БҮЛЭГ 1

Онолын хэсэг

1.1 Өгөгдөл хуваалцах үйлчилгээ

Өгөгдөл хуваалцах гэдэг нь ижил өгөгдлийн нөөцийг олон программ, хэрэглэгч эсвэл байгууллагад ашиглах боломжтой болгохыг хэлнэ. Үүнд технологи, практик, хууль эрхзүйн орчин, соёлын элементүүд багтах ба өгөгдлийн бүрэн бүтэн байдлыг алдагдуулахгүйгээр аж ахуйн нэгжүүд хялбар, аюулгүй өгөгдөлд хандах боломжийг олгодог.

Байгууллагын үр ашгийг дээшлүүлж, борлуулагчид болон түншүүдтэй хамтын ажиллагааг дэмжинэ. Хуваалцсан өгөгдлийн эрсдэл, боломжуудын талаар мэдлэгтэй байх нь үйл явцын салшгүй хэсэг юм. [1].

Өгөгдөл хуваалцах технологиуд

Өгөгдөл хуваалцах олон технологи байдгаас зарим технологиудаас дурдвал.

- **Өгөгдлийн агуулах (Data warehousing)** нь нэг буюу хэд хэдэн ялгаатай эх сурвалжийг нэгтгэсэн төвлөрсөн агуулах юм. Архитектур нь шатлалаас бүрддэг. Дээд давхарга нь тайлагнах, дүн шинжилгээ хийх, үр дүнг харуулдаг front-end клиент юм. Дунд шат нь өгөгдөлд хандах, дүн шинжилгээ хийхэд ашигладаг аналитик механизмаас бүрдэнэ. Доод шат нь өгөгдлийг ачаалах, хадгалах өгөгдлийн сангийн сервер юм. Дээд болон дунд түвшний програмууд нь доод давхаргад хадгалагдсан нийтлэг өгөгдлийн багцыг хуваалцах боломжтой.
 - Олборлох, хувиргах, ачаалах суурилсан (ETL based data warehouse)
 - Олборлох, ачаалах, хувиргах суурилсан (ELT based data warehouse)
- **Хэрэглээний программчлалын интерфейс (API)** нь программ хангамжийн бүрэлдэхүүн хэсгүүд тодорхой протоколуудыг ашиглан хоорондоо харилцах боломжийг олгодог механизм юм. Интерфейс нь хоёр программын хоорондох үйлчилгээний тохиролцоо гэж үзэж болно. Энэхүү тохиролцоо нь хэрхэн харилцах хүсэлт болон хариултыг тодорхойлдог. Хандалтыг нарийн тодорхойлж болдог ба хэрэглэгчид яг ямар өгөгдөл хүсэж болохыг зааж өгдөг.
 - SOAP API нь XML ашигладаг ба уян хатан бус.
 - RPC API нь алсаас өөр сервер болон сервисийн функцийг дуудаж ажиллуулдаг.
 - Websocket API нь холболт үүсгэж сервер болон хэрлэгч аль ч чиглэлд нэг холболтоор дамжуулах боломжтой.
 - REST API нь уян хатах PUT DELETE гэх мэт хүсэлт илгэдэг.
- **Холбооны сургалт (Federated learning)** нь тархсан өгөгдлийг багц дээр хиймэл оюун ухааныг сургах боломжийг олгодог. Бүх өгөгдлийг нэг дор цуглуулж нэгтгэхийн оронд тус тусдаа төхөөрөмж дээр хадгалж зөвхөн загварын шинэчлэлтүүдийг төв сервер рүү илгээдэг.
- **Блокчейн технологи** нь сүлжээн дотор ил тод мэдээлэл солилцох боломжийг олгодог өгөгдлийн сангийн дэвшилтэт механизм юм. Өгөгдлийг гинжин хэлхээнд холбосон блокуудад хадгалдаг. Сүлжээнээс зөвшилцөлгүйгээр гинжийг устгах эсвэл өөрчлөх боломжгүй.

- **Өгөгдөл солилцох платформууд**

Нээлттэй өгөгдлийн платформууд нь өөр өөр өгөгдлийн багцыг нийтийн хэрэгцээнд ашиглах боломжийг олгодог. Ихэвчлэн өгөгдлийн менежмент, өгөгдлийн аюулгүй байдал, өгөгдөл нэгтгэх, өгөгдөл хуваалцах, хамтран ажиллах зэрэг олон төрлийн функцуудыг санал болгодог.

Файл хуваалцах

Олон хэрэглэгч эсвэл төхөөрөмж нэг файл эсвэл багц файлд хандах боломжийг олгох практикийг хэлнэ. Файл хуваалцахыг уламжлалт болон орчин үеийн янз бүрийн арга технологи ашиглан хийж болно.[2]

Уламжлалт

- Физик зөөвөрлөгч: Файлуудыг CD, DVD эсвэл USB гэх мэт физик медиа ашиглан хуваалцаж болно. Энэ арга нь интернэтийн хандалт хязгаарлагдмал эсвэл боломжгүй үед файл хуваалцахад тустай.
- Сүлжээгээр файл хуваалцах: Файлуудыг Server Message Block (SMB) эсвэл Network File System (NFS) зэрэг технологийг ашиглан дотоод сүлжээгээр хуваалцаж болно. Энэ арга нь байгууллага дотор эсвэл гэрийн сүлжээн дэх төхөөрөмжүүдийн хооронд файл хуваалцах боломжтой.

Орчин үеийн

- Клоуд сан: Dropbox, Google Drive эсвэл OneDrive зэрэг үүлэн хадгалах үйлчилгээ нь хэрэглэгчдэд үүлэн доторх файлуудыг хадгалах, бусадтай хуваалцах боломжийг олгодог. Энэ арга нь өөр өөр төхөөрөмжид байршил хамаарахгүй файл хуваалцахад тустай бөгөөд интернэт холболттой хаанаас ч хандах боломжтой.
- Файл дамжуулах үйлчилгээ: WeTransfer, Hightail эсвэл Filemail зэрэг файл дамжуулах үйлчилгээ нь хэрэглэгчдэд том хэмжээний файлуудыг бусдад хурдан бөгөөд хялбар илгээх боломжийг олгодог. Энэ арга нь байгууллагаас гадуурх хүмүүстэй файл хуваалцах эсвэл имэйл хавсралтын хязгаарт хүрсэн үед хэрэгтэй.
- Peer-to-peer файл хуваалцах: Peer-to-peer (P2P) файл хуваалцах нь хэрэглэгчдэд төвлөрсөн сервер ашиглахгүйгээр шууд бие биетэйгээ файл хуваалцах боломжийг олгодог. P2P файл хуваалцах нь ихэвчлэн кино, программ хангамж гэх мэт том файлуудыг хуваалцахад ашиглагддаг боловч бусад төрлийн файлуудыг хуваалцахад ашиглаж болно.

Файл болон өгөгдөл хуваалцах нь олон давуу талтай ч өгөгдлийг эрсдэлд оруулдаг.

1.2 Өгөгдлийн аюулгүй байдал

Өгөгдлийн аюулгүй байдал гэдэг нь дижитал мэдээллийг зөвшөөрөлгүй хандах, өөрчлөх, хулгайлахаас хамгаалах үйл ажиллагаа юм. Физик төхөөрөмжийн хамгаалалтаас эхлээд хандалтын удирдлага, программ хангамжийн логик аюулгүй байдал мэдээллийн аюулгүй байдлын бүх талыг хамарсан ойлголт юм. [3]

Нууц мэдээлэл, санхүүгийн чадамж бичиг баримт зэргийг буруу зорилгоор ашиглах боломжтой. Байгууллагын хувьд хэрэглэгчдийнхээ мэдээллийг алдаж буруу гарт орохоос сэргийлж хамгаалах ёстой. Халдлагад өртөх, мэдээлэлээ алдах нь тухайн байгууллагын нэр хүнд нь халтай ба хэрэглэгчдийн итгэлийг алдах аюултай.

Өгөгдөл хуваалцах эрсдэлүүд

- **Нууцлалыг задруулах**

Хувийн нууцыг алдагдуулахгүйгээр өгөгдлийг хуваалцахын тулд шифрлэлт, засварлах зэрэг нууцлалыг хамгаалах технологи нь өгөгдлийг аюулгүй хуваалцах боломжийг олгодог.

- **Өгөгдлийн буруу тайлбар**

Өгөгдөл бэлтгэгч болон хэрэглэгчдийн хоорондын харилцаа холбоо дутмагаас буруу тайлбар гарч болзошгүй. Шинжээчид тайлан, үр дүнг тайлбарлахдаа буруу таамаглал дэвшүүлж болно. Жишээлбэл, тухайн сард үйлчлүүлэгчдийн захиалга багассан нь маркетингийн төсөв багатай холбоотой байж болох ч бодит шалтгаан нь бүтээгдэхүүний бэлэн байдлын саатал байж болох юм.

- **Өгөгдлийн чанар муудах**

Давхардсан эсвэл дутмаг чанар муутай өгөгдөл авах эрсдэлтэй.

Өгөгдлийг аюулгүй хуваалцах төрлүүд

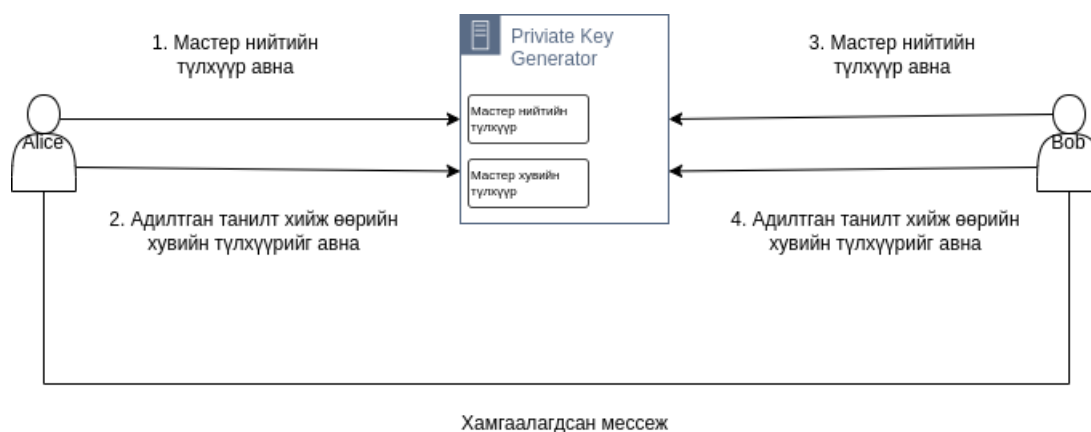
- **Шифрлэлт** нь түлхүүр нууц үггүйгээр өгөгдлийг унших боломжгүй бологдог ба криптографын алгоритмуудыг ашиглан энгийн текстийг шифрлэх үйл явц юм. Энэ нь халдагчид өгөгдөлд нэвтэрсэн байсан ч зохих итгэмжлэлгүйгээр үүнийг уншиж чадахгүй гэдгийг баталгаажуулахад тусалдаг.
- **Хандалтын удирдлага** нь нууц өгөгдөлд хэн хандах эрхтэй болохыг тэдний үүрэг, зөвшөөрлийн түвшинд үндэслэн хязгаарладаг. Үүнд нууц үг, биометрийн баталгаажуулалт, хамгаалалтын токен зэрэг арга хэмжээ багтана.
- **Нөөцлөх, сэргээх** үйл явц нь аюулгүй байдлын зөрчил эсвэл өгөгдөл алдагдсан тохиолдолд сэргээх боломжтой байхын тулд мэдээллийн хуулбарыг үүсгэх, хадгалах явдал юм.
- **Физик аюулгүй байдал** нь өгөгдөл хадгалах төхөөрөмж болон физик хандалтыг хамгаалахын тулд түгжээтэй хаалга, хамгаалалтын камер зэрэг физик хамгаалалтын арга хэмжээг ашигладаг.
- **Өгөгдөл устгах** Өгөгдлийг устгах нь хамгийн аюулгүй хэдий дахин ашиглах боломжгүй. Ихэвчлэн дахин ашиглахгүй өгөгдлийн дарж бичих зэргээр устгадаг.
- **Өгөгдлийн далдлах** нь нууц мэдээллийг анхны өгөгдлийн бүтцийг хадгалан зөвшөөрөлгүй хэрэглэгчдэд ашиглах боломжгүй болгож буй хуурамч мэдээллээр солих явдал юм.

1.3 Шифрлэх схемүүд

Танилтад суурилсан шифрлэлт (IBE)

Тэмдэгт мөр зэрэг мэдэгдэж буй утгаас нийтийн түлхүүр үүсгэх боломжийг олгодог. Итгэмжлэгдсэн гуравдагч тал түлхүүрүүдийг үүсгэж өгдөг(PKG). Хувийн түлхүүр үүсгэгч (PKG) итгэмжлэгдсэн гуравдагч тал холбогдох хувийн түлхүүрүүдийг үүсгэдэг. PKG эхлээд мастер нийтийн түлхүүрийг нийлтэй тавьж, мастер хувийн түлхүүрийг хадгална. Аль ч тал мастер нийтийн түлхүүр, таних утгыг ашиглан тохирох нийтийн түлхүүрийг гаргаж авах боломжтой. Харгалзах хувийн түлхүүрийг авахын тулд мастер түлхүүрээр гаргаж авсан таних түлхүүрийг ашиглана. [4]

1. Бэлтгэл үе: PKG нь өөрийн мастер түлхүүрүүдийг үүсгэнэ.
2. Алис нийтийн мастер түлхүүрийг авна. Өөрийн хувийн түлхүүрийг авна.
3. Боб-ын имэйл гэх мэт өвөрмөц мэдээллээр Боб-ын нийтийн түлхүүрийг авч шифрлэлт хийн явуулна.
4. Боб PKG-ээс өөрийн хувийн түлхүүрийг авч шифрийг тайлж авна.



ЗУРАГ 1.1: Танилтад суурилсан шифрлэлт

IBE-дээр суурилсан алгоритмууд

- Boneh–Franklin (BF-IBE)
- Sakai–Kasahara (SK-IBE)
- Boneh–Boyen (BB-IBE)

Давуу талууд

- Тодорхойгүй олон хэрлэгч үед давуу талтай
- Мөн нийтийн түлхүүрийг хэрэглээг халж орлох боломжтой
- Мөн цаг минут зэрэг ашиглан хандах эрэхийг нарийн тодорхойлж болно.

Шинж чанарт суурилсан шифрлэлт (ABE)

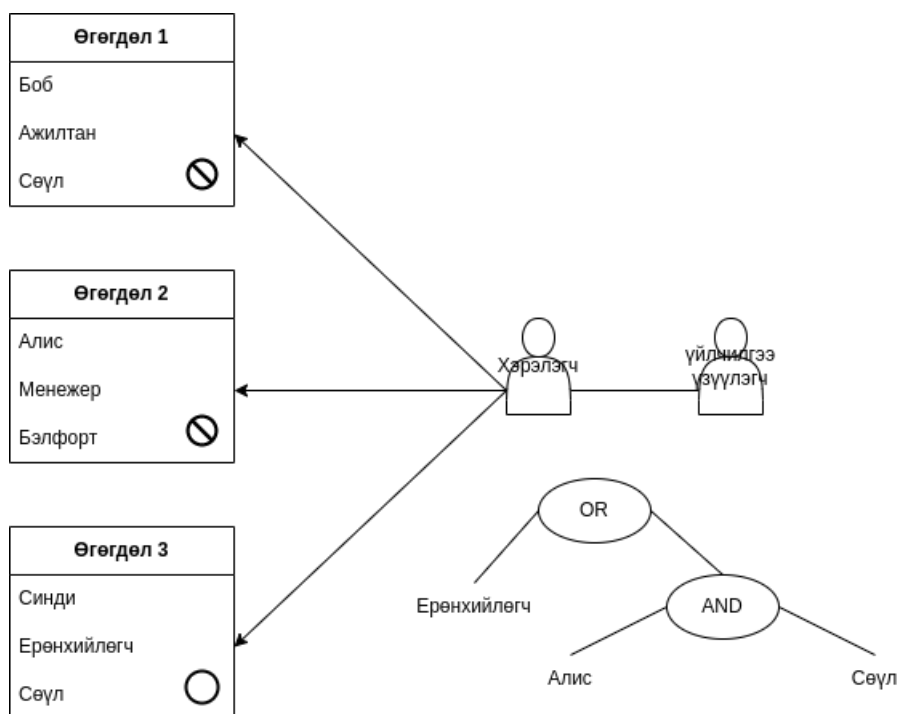
ABE-тэй ерөнхийдөө төстэй. Шинж чанаруудаар бүлэглэж зөвхөн нэг хэрлэгчийн түлхүүр ашиглахгүй олон хүн тайлах боломжтой. Үндсэн хоёр төрөлтэй. Түлхүүр-Дүрэмийн шинж чанарт суурилсан шифрлэлт (KP-ABE) болон Шифртескт-Дүрэмийн шинж чанарт суурилсан шифрлэлт (CP-ABE).[5]

ABE ерөнхий санаа нь сайн боловч сул талуудтай.

- Түлхүүр зохицуулалт
- Түлхүүр хадгалах
- Түлхүүр хүчингүй болгох

KP-ABE

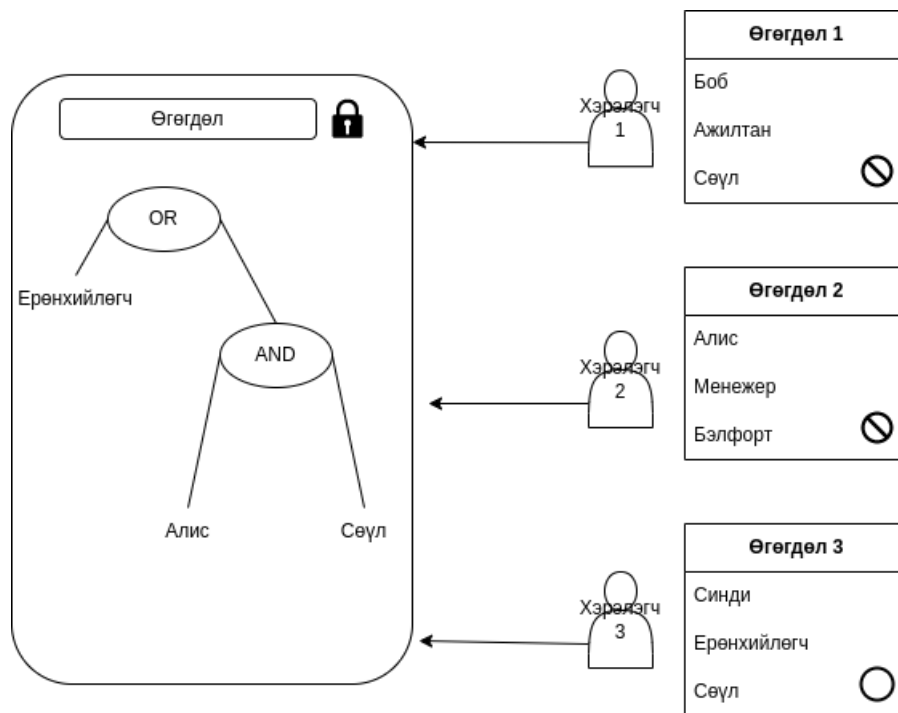
Хэрэглэгч бүр шинж чанаруудтай. Хэрлэгч нь хандалтын модыг өөртөө агуулдаг тул бүх хэрэглэгч өгөгдөл рүү хандах эрхтэй байдаг.



ЗУРАГ 1.2: Танилтад суурилсан шифрлэлт KP-ABE

CP-ABE

KP-ABE-ээс ялгаатай зүйл нь шифр нь өрөө хандалтын модыг агуулдаг. KP-ABE-ээс ялгаатай нь шифрлэгдсэн текст нь хандалтын модыг агуулдаг. Хэрэв хэрэглэгчийн шинж чанар нь шифрлэгдсэн текст дэх хандалтын модны нөхцөлийг хангаж байвал шифрлэгдсэн өгөгдлийг тайлах боломжтой. Хандалтын мод нь шифрлэгдсэн өгөгдөлд байгаа тул өгөгдлийн хандалтыг хянах боломжтой.



ЗУРАГ 1.3: Танилтад суурилсан шифрлэлт CP-ABE

Гомоморф шифрлэлт (HE)

Энэ нь шифрлэгдсэн өгөгдлийг тайлахгүйгээр тооцоолол хийх боломжийг олгодог шифрлэлтийн төрөл юм. Аюулгүй үүлэн тооцоолол, цахим санал хураалт, эмнэлгийн бүртгэл гэх мэт төрөл бүрийн хэрэглээнд хэрэглэж болдог.[6]

- Хэсэгчилсэн гомоморф шифрлэлт (PHE). PHE-д "хэсэгчилсэн" гэдэг нь зөвхөн сонгосон математик функцийг шифрлэгдсэн утгууд дээр боловсруулах боломжтой. Зөвхөн нэг үйлдлийг нэмэх эсвэл үржүүлэхийг шифрлэгдсэн текст дээр хязгааргүй олон удаа хийхийг зөвшөөрдөг.
- Зарим ижил төстэй шифрлэлт (SHE). "Somewhat" нь PHE-ээс илүү ерөнхий шинж чанартай бөгөөд энэ нь нэмэх, үржүүлэх бүхий гомоморф үйлдлийг дэмждэг. Гэхдээ энд байгаа гол сул тал нь зөвхөн цөөн тооны үйлдлийг гүйцэтгэх боломжтой юм.
- Бүрэн гомоморфик шифрлэлт (FHE). PHE болон SHE нь хязгаарлагдмал ажиллагаатай бол бүрэн гомоморф шифрлэлт нь шифрлэгдсэн текст дээр хязгаарлалтгүйгээр нэмэх болон үржүүлэх үйлдлийг хоёуланд нь ашиглах чадвартай.

Хямд үүлэн тооцоолол, үүлэн хадгалах систем нь аж ахуйн нэгжүүд болон хувь хүмүүсийн өгөгдлөө ашиглах, удирдах арга хэлбэрийг үндсээр нь өөрчилсөн. AES гэх мэт уламжлалт шифрлэлтийн аргууд нь маш хурдан бөгөөд өгөгдлийг шифрлэгдсэн хэлбэрээр хялбархан хадгалах боломжийг олгодог. Гэсэн хэдий ч, шифрлэгдсэн өгөгдөл дээр энгийн дүн шинжилгээ хийхийн тулд үүлэн сервер нууц түлхүүрт хандах шаардлагатай бөгөөд энэ нь аюулгүй байдлын асуудалд хүргэдэг, эсвэл өгөгдөл эзэмшигч нь өгөгдлийг татаж авах, тайлах, локал байдлаар ажиллуулах шаардлагатай. зардал ихтэй, логистикийн сорилтыг бий болгоно. Клоуд

нь шифрлэгдсэн өгөгдөл дээр шууд ажиллаж, зөвхөн шифрлэгдсэн үр дүнг өгөгдлийн эзэнд буцааж өгөх боломжтой тул гомоморф шифрлэлтийг энэ хувилбарыг ихээхэн хялбарчлахад ашиглаж болно. Хэрэглээний илүү төвөгтэй хувилбарууд нь гуравдагч этгээдийн ажиллах боломжтой хувийн өгөгдөл бүхий олон талуудыг оролцуулж, үр дүнг нэг буюу хэд хэдэн оролцогчид шифрлэхээр буцааж өгч болно.

Давуу талууд

- Найдваргүй гуравдагч орчинд өгөгдөл найдвартай, нууцлалтай хэвээр байна. Өгөгдөл нь үргэлж шифрлэгдсэн хэвээр байх бөгөөд энэ нь нууц мэдээлэл алдагдуулах магадлалыг бууруулдаг.
- Бүрэн гомоморф шифрлэлтийн схемүүд нь квант халдлагын эсрэг тэсвэртэй.

Сул талууд

- Тооцооллын хурд багатай
- Нарийвчлал багатай

1.4 Файл шифрлэх хадгалах

Файлыг тэгш хэмт болон тэгш бус шифрлэлтээр аль алинаар нь шифрлэдэг. Тэгш хэмт шифрлэлт нь тэгш бус хэмт шифрлэлтээс хувьд хурдан боловч түлхүүр дамжуулах зэрэгт асуудал үүсдэг. Ихэвчлэн тэгш бус шифрлэлтээр файлыг шифрлэдэг ба тэгш хэмт шифрлэлтээс илүү хурдан байдаг.[7]

Нийтлэг ашигладаг алгоритмууд

- Нарийвчилсан шифрлэлтийн стандарт (AES): AES нь тэгш хэмт шифрлэлтийн алгоритм бөгөөд хамгийн найдвартай гэж тооцогддог.
- Гурвалсан өгөгдөл шифрлэлтийн стандарт (3DES): 3DES нь аюулгүй гэж тооцогддог хуучин тэгш хэмтэй шифрлэлтийн алгоритм юм. Энэ нь ихэвчлэн өндөр хамгаалалттай байх шаардлагагүй өгөгдлийг шифрлэхэд ашиглагддаг.
- Ривест-Шамир-Адлеман (RSA): RSA нь бусадтай хуваалцах шаардлагатай өгөгдлийг шифрлэхэд ихэвчлэн ашиглагддаг тэгш бус шифрлэлтийн алгоритм юм. Үүнийг мөн тоон гарын үсэг үүсгэхэд ашигладаг бөгөөд үүнийг баримт бичиг эсвэл мессежийг жинхэнэ эсэхийг шалгахад ашиглаж болно.
- 64 битийн блокийн хэмжээтэй, 32 битээс 448 бит хүртэлх хувьсах түлхүүрийн урттай. Энэ нь 16 шаттай Feistel шифр бөгөөд түлхүүрээс хамааралтай S-box ашигладаг.

Шифрлэх алгоритмуудыг симуляцийг Windows 64 бит, i3 процессор, 4 GB RAM бүхий 1.90 GHz CPU бүхий зөөврийн компьютер туршиж хийсэн. Энгийн текст файл болон бинари (binary) файл дээр шифрлэх туршилт хийсэн. Энгийн текст файл шифрлэх хүснэгтийг харахад AES болон BLOWFISH алгоритмууд хурдан байна. 1.1.

Хүснэгт 1.1: Текст файл шифрлэх гүйцэтгэл

Файл Size	AES (мсек-ээр)	DES (мсек-ээр)	RSA (мсек-ээр)	BLOWFISH (мсек-ээр)
1 MB	80	136.2	425.6	133.2
2 MB	154.7	269.6	710.9	192.6
5 MB	376.1	665.4	1710.9	373.6
10 MB	683.7	1236.2	3017.1	702.5
20 MB	1350.5	2356.5	6641	1355.2

Энгийн текст файлын шифрийг тайлах хүснэгтийг харахад BLOWFISH алгоритм хурдан байна. 1.2

Хүснэгт 1.2: Текст файлын шифрийг тайлах гүйцэтгэл

Файл	AES (мсек-ээр)	DES (мсек-ээр)	RSA (мсек-ээр)	BLOWFISH (мсек-ээр)
1 MB	118.9	144.6	3.8	50.2
2 MB	197.6	269.6	3.5	126.3
5 MB	457.7	690.9	3.7	210.7
10 MB	897.5	1294.6	3.7	575.4
20 MB	1844.5	2744.2	4.0	1025.5

Бинари файл шифрлэх хүснэгтийг харахад AES алгоритм хурдан байна. 1.3.

Хүснэгт 1.3: Бинари файл шифрлэх гүйцэтгэл

Файл	AES (мсек-ээр)	DES (мсек-ээр)	RSA (мсек-ээр)	BLOWFISH (мсек-ээр)
1 MB	74.6	120.8	393.3	150.5
2 MB	136.2	256.5	734.8	220.9
5 MB	393.4	687.4	1773.7	439.1
10 MB	711.2	1264.9	3446.1	731.3
20 MB	1362.6	2316.6	6868.5	1376.4

Бинари файлын шифрийг тайлах хүснэгтийг харахад BLOWFISH алгоритм хурдан байна. 1.4.

Хүснэгт 1.4: Бинари файлын шифрийг тайлах гүйцэтгэл

Файл	AES (мсек-ээр)	DES (мсек-ээр)	RSA (мсек-ээр)	BLOWFISH (мсек-ээр)
1 MB	111.1	156.5	3.6	44.2
2 MB	196.8	281.3	3.3	84.7
5 MB	460.4	673.6	3.6	203.8
10 MB	876.8	1451.7	3.6	408.6
20 MB	1791.9	2556.9	3.8	923.7

BLOWFISH нь AES алгоритмаас хурдан боловч аюулгүй байдлын хувьд AES нь BLOWFISH-ээс илүү.

Файлын системийг шифрлэх

Файл системийн түвшинд шифрлэлтийг файлд суурилсан шифрлэлт (FBE) эсвэл файл/хавтас шифрлэлт гэдэг нэрлэдэг дискний шифрлэлтийн нэг төрөл юм.[8]

Файл системийн шифрлэлтийн төрөл

- Криптограф файлын систем
- Ерөнхий зориулалт бүхий файлын шифрлэлт системүүд

Давуу талууд

- Файлд суурилсан уян хатан түлхүүрийн удирдлага гэдэг нь файл бүрийг тусдаа түлхүүрээр шифрлэх боломжтой.
- Шифрлэгдсэн файлын тус тусад нь удирдах нь шифрлэгдсэн файлыг бүхэлд нь засаж өөрчлөхийн оронд зөвхөн өөрчлөгдсөн хэсгийг л өөрчлөх боломжтой.
- Хандалтыг нийтийн түлхүүр ашиглах хянах боломжтой.

Криптограф файлын систем

Криптограф файлын системүүд нь шифрлэлт, аюулгүй байдлын үүднээс тусгайлан бүтээгдсэн (ерөнхий зориулалтын бус) файлын системүүд юм. Тэд ихэвчлэн мета өгөгдлийг оруулаад агуулсан бүх өгөгдлийг шифрлэдэг. Диск дээрх формат болон өөрийн блокийн хуваарилалт хийдэггүй одоо байгаа файлын системүүдийн дээр байрладаг.

Ерөнхий зориулалт бүхий файлын шифрлэлт системүүд

Криптограф файлын систем болон бүрэн дискний шифрлэлтээс ялгаатай нь файлын системийн түвшинд шифрлэлт хийдэг ерөнхий зориулалтын файлын системүүд юм. Файлын нэр, хэмжээ, өөрчлөлтийн цагийн тэмдэг гэх мэт файлын системийн мета өгөгдлийг ихэвчлэн шифрлэдэггүй.

1.5 Бүлгийн Дүгнэлт

Өгөгдөл хуваалцах болон өгөгдлийн аюулгүй байдлын ерөнхийд нь судалсан. Орчин үеийн шифрлэлтийн схемүүд болох танилтад суурилсан шифрлэлт (IBE), шинж чанарт суурилсан шифрлэлт (ABE), гомоморф шифрлэлт (HE) схемүүдийг судалсан. Файл шифрлэх аргуудыг судалсан.

БҮЛЭГ 2

Судалгааны хэсэг

2.1 Прокси дахин шифрлэлт

Прокси дахин шифрлэлт нь нийтийн түлхүүрээр шифрлэсэн өгөгдлийг дахин шифрлэж өөр хувийн түлхүүрээр тайлах боломжийг олгодог.

Мамбо болон Окамото шифрийг тайлан дараа нь шифрлэх уламжлалт аргыг сайжруулах зорилгоор анх гаргаж ирсэн.

1998 онд Blaze, Bleumer, Strauss (BBS) нар "atomic proxy cryptography" гэсэн ойлголтыг санал болгосон бөгөөд үүнд хагас итгэмжлэгдсэн прокси нь үндсэн энгийн текстийг харалгүйгээр Алисын шифрийг Бобын шифр текст болгон хувиргадаг. El Gamel дээр суурилсан схем ба прокси буюу гуравдагч талийн тусламжтайгаар шифрийг дамжуулах зорилготой. [9]

Прокси дахин шифрлэлт үйл явц нь гурван хэсгээс тогтоно.

- **Төлөөлөгч:** Прокси дахин шифрлэлт ашиглан шифрийг тайлах эрхээ өөр хүнд шилжүүлэх өгөгдлийн эзэн. Дахин шифрлэлтийн түлхүүр үүсгэж, прокси руу илгээдэг. Төлөөлөгчийг ихэвчлэн "Алис" гэж нэрлэдэг.
- **Орлогч:** Төлөөлөгчийн шифрийг тайлах эрхтэй хүн. Ихэвчлэн "Боб" гэдэг нэртэй байдаг.
- **Прокси:** Дахин шифрлэх шифрийг дамжуулах гуравдагч тал.

Прокси дахин шифрлэлт үндсэн хоёр төрөлтэй.

- Нэг чиглэлт (Unidirectional PRE)
- Хоёр чиглэлт (Bidirectional PRE)

Мөн олон дахин шифрлэх боломжтой болон нэг удаагийн шифрлэх боломжтой гэж хуваадаг.

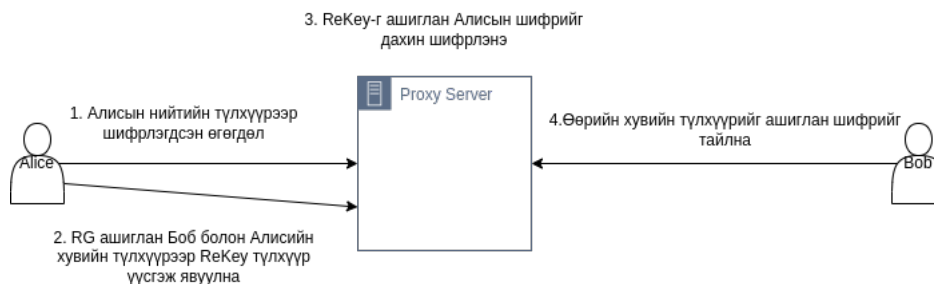
Прокси дахин шифрлэх схем дараах хэсгүүдээс тогтоно.

- **Нийтийн параметрууд:** Нийтлэг нийтэд байдаг параметрууд. Түлхүүрийн урт зэрэг.
- **Нийтийн болон нууц түлхүүрийн хослол:** Хэрэглэгч бүр нийтийн/нууц түлхүүрийг үүсгэдэг. Нийтийн түлхүүрийг нийтэд тавьж, харин хувийн түлхүүр нь зөвхөн хэрэглэгч өөртөө үлдээнэ. Боб Алис руу мессеж илгээхдээ Алисын нийтийн түлхүүрийг ашиглан шифрлэж, Алис түүний хувийн түлхүүрийг ашиглан тайлна.
- **Төлөөлөгчийн түлхүүрүүд:** Хэрэглэгч бүр энд $rk_{Alice} \rightarrow Bob$ гэж тэмдэглэсэн дурын төлөөлөгчид (дахин шифрлэх) түлхүүр үүсгэж болно. $rk_{Alice} \rightarrow Bob$ тулд Алис прокси дахин шифрлэлтийн алгоритмыг ашиглан хувийн түлхүүрээ Бобын нийтийн түлхүүртэй хослуулан түлхүүр гаргаж авна. Гаргаж авсан түлхүүрээр дахин шифрлэж төлөөлөгчийн хувийн түлхүүрээр тайлах боломжтой болно.
- **Шифрлэгдсэн текст:** Хэрэглэгч ямар нэгэн нийтийн түлхүүрээр мессежийг (энгийн текст) шифрлэх үед шифр текст үүснэ.

Нэг чиглэлт $PRE (KE, RG, E, R, D)$ хэсгүүдээс тогтоно.

1. Алис, Боб болон Чарли хувийн болон нийтийн түлхүүрийг үүсгэнэ. (KE)
2. Алис өгөгдлөө шифрлэж серверт байршуулна.

3. Алис-ын өгөгдлийг Боб-той хуваацлахын тулд $RE(pkB, skB, pkC, skC)$ шифрлэж серверт байршуулна. Бобын хувийн заавал шаардахгүй үүсгэж болно.
4. Боб RE-г ашиглаж үүсгэсэн түлхүүрийг серверт явуулж Алисын файлыг дахин шифрлэж Чарли тайлах боломжтой болно.



ЗУРАГ 2.1: Прокси дахин шифрлэх схем

Давуу талууд:

- Нууцлалыг сайжруулна: PRE нь оролцогч талуудын хувийн мэдээллийг задруулахгүйгээр өгөгдлийг хуваалцахыг зөвшөөрснөөр нууцлалыг сайжруулдаг. Талууд нууцаар эсвэл хувийн нууц мэдээллийг задруулахгүйгээр мэдээллээ хуваалцахыг хүссэн тохиолдолд ашиглах боломжтой.
- Илүү уян хатан шифрлэсэн өгөгдлийг өргөн хүрээтэй тараах боломжтой

Сул талууд:

- Проксид итгэх: PRE нь дахин шифрлэлтийг гүйцэтгэхэд гуравдагч талын прокси дээр тулгуурладаг ба схемийн аюулгүй байдал нь прокси талаас хамаарна.
- Хязгаарлагдмал өргөтгөх чадвар: PRE нь өргөтгөх чадварын хувьд хязгаарлагдмал байж болно. Учир нь хэрэглэгчдийн тоо нэмэгдэхийн хэрээр олон талыг дэмжихэд шаардлагатай дахин шифрлэлтийн түлхүүрүүдийн тоо хурдацтай өсөх болно. Энэ нь гол менежментийг төвөгтэй болгож, удирдахад хэцүү болгодог.
- Potential for replay attacks: PRE нь халдагч хариуг зогсоож хандах эрхийг өөрт ашигтай солих боломжтой.
- Хүчингүй болгоход хүндрэлтэй: PRE дахь өгөгдөлд хандах эрхийг цуцлах нь ялангуяа олон тал оролцсон тохиолдолд төвөгтэй. Хэрэв аль нэг талын дахин шифрлэлтийн түлхүүр алдагдсан бол бусад талуудын мэдээлэлд хандах эрхэд нөлөөлөхгүйгээр өгөгдөлд хандах эрхийг цуцлах нь хэцүү юм.
- Хязгаарлагдмал хэрэглээ: PRE нь харьцангуй шинэ бөгөөд шинээр гарч ирж буй технологи хэвээр байгаа бөгөөд илүү уламжлалт шифрлэлтийн схемүүдтэй харьцуулахад хэрэглээ нь хязгаарлагдмал байдаг. Энэ нь технологийг хэрэгжүүлэх, удирдах туршлагатай мэргэжилтнүүд бага байдаг.

Прокси дахин шифрлэх схемийн ерөнхий хэрэглээ

- **Аюулгүй өгөгдөл хуваалцах:** Өгөгдлийн олон хэрлэгчид найдвартай хуваалцахад ашигладаг. Өгөгдлийн эзэмшигч нь өгөгдлийг өөрийн түлхүүрээр шифрлэж, дахин шифрлэж прокси руу шилжүүлдэг. Дараа нь прокси нь хүлээн авагчийн түлхүүрээр шифрлэгдсэн өгөгдлийг хувиргаж, өгөгдөл эзэмшигчийн оролцоогүйгээр шифрийг тайлж, өгөгдөлд хандах боломжийг олгодог.
- **Cloud Storage:** Үүл хадгалах системд хадгалагдсан мэдээллийн аюулгүй байдлыг сайжруулахын тулд прокси дахин шифрлэлтийг ашиглаж болно. Нууц мэдээллийг үүлэн үйлчилгээ үзүүлэгч рүү шууд оруулахын оронд өгөгдлийг эзэмшигчийн түлхүүрээр шифрлэж, прокси түлхүүрээр дахин шифрлэх боломжтой. Дараа нь прокси нь дахин шифрлэгдсэн өгөгдлийг үүлэн дотор хадгалдаг. Энэ нь үүлэн үйлчилгээ үзүүлэгч нь зөвхөн дахин шифрлэгдсэн өгөгдөлд хандах эрхтэй бөгөөд анхны контентыг уншиж чадахгүй байх боломжийг олгодог.
- **Түлхүүр зохицуулалт: (Key escrow)** Түлхүүр эскроу гэдэг нь криптограф түлхүүрийг итгэмжлэгдсэн гуравдагч этгээдэд хадгалах практикийг хэлнэ. Энэ тохиолдолд анхны өгөгдөл нь эзэмшигчийн түлхүүрээр шифрлэгдсэн бөгөөд дахин шифрлэлтийн түлхүүрийг проксид хадгалагдана. Хэрэв эзэмшигч нь түлхүүрээ алдсан эсвэл өөр төхөөрөмжөөс хандах шаардлагатай бол прокси нь өгөгдлийг шинэ түлхүүрээр дахин шифрлэх боломжтой бөгөөд ингэснээр эзэмшигчид дахин хандах эрх олгоно.
- **Аюулгүй имэйл:** Илгээгч нь хувийн түлхүүрээ шуудангийн сервер эсвэл бусад зуучлагчтай хуваалцахын оронд өөрийн түлхүүрээр имэйлээ шифрлэж, дахин шифрлэх ажиллагааг прокси руу шилжүүлэх боломжтой. Дараа нь прокси нь хүлээн авагчийн түлхүүрээр имэйлийг дахин шифрлэх боломжтой бөгөөд зөвхөн хүлээн авагч нь мессежний кодыг тайлж унших боломжтой.

2.2 Хөгжүүлэх технологи, хэл сонгох

Прокси дахин шифрлэлт схемийг ашиглан файл хуваалцах систем хөгжүүлэхээр болсон. Хөгжүүлэлтэд хэрэгтэй хэл болон технологи сонгосон.

Систем хоёр хэсгээс тогтох ба. API сервер болон хэрлэгч талын программ. Пайтон маш олон нэмэлт сангууд байдаг маш том нийгэмлэгтэй (community) хэл. Бичиглэл хялбар олон давуу талтай тул пайтон хэлийг сонгосон. Үүнд:

Flask framework

Пайтон хэл дээр бичсэн веб хөгжүүлэхэд зориулагдсан фреймворк юм. Хөнгөхөн олон нэмэлт сан шаардахгүй хэрэгтэй сангуудаа суулгаад хөгжүүлэх боломжтой. Сурахад хялбар өөрийн хүссэн загвараар загварчилж хийх боломжтой. RESTful API гаргахад хялбар. Хэрэгтэй гэвэл нэмэлт сан ORM зэрэг өөр сангууд суулгаж хамт ашиглах боломжтой.

```
1 from flask import Flask
2
3 app = Flask(__name__)
4
5 @app.route("/")
```

```

6     def hello_world():
7         return "<p>Hello, World!</p>"
8 @app.get("/")
9     def hello_world():
10        return "<p>Hello, get request!</p>"

```

Эх код 2.1: Энгийн API үүсгэж буй код

app.route endpoint бичих боломжтой маш хялбар мөн хурдан бичих боломжтой.

Tkinter

Хэрэглэгчийн интерфэйс (GUI) үүсгэхэд ашигладаг Python номын сан юм. Tcl/TK GUI дээр суурилсан. Линукс виндовс зэрэг олон үйлдлийн системийг дэмждэг. Нэмэлт сангуудтай ажиллах боломжтой.

```

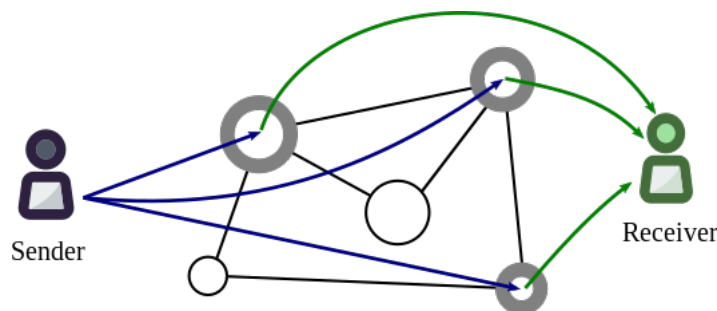
1 import tkinter as tk
2
3 root = tk.Tk()
4 # Create a label widget.
5 label = tk.Label(root, text="Hello, world!")
6 # Create a button widget.
7 button = tk.Button(root, text="Click me!")
8 # Add the label and button widgets to the root window.
9 label.pack()
10 button.pack()
11 # Start the main loop.
12 root.mainloop()

```

Эх код 2.2: Энгийн цонх үүсгэж буй код

pyUmbral

Прокси дахин шифрлэлтийг файтон дээр хэрэгжүүлсэн пайтоны сан юм. OpenSSL болон Cryptography.io ашигласан.



ЗУРАГ 2.2: pyUmbral

ECIES-KEM болон BBS98 санаа авсан нэг чиглэлтэй дахин шифрлэлтийн сан. pyUmbral функцуудийг ерөнхийд нь гурав ангилсан.

1. Түлхүүр үүсгэх алгоритмууд.

- KeyGen()
- ReKeyGen(skA, pkB, N, t)

2. Encapsulation болон Decapsulation

- Encapsulate(pkA)
- Decapsulate(skA, capsule)

3. Re-Encapsulation болон Fragments Decapsulation.

- ReEncapsulation(kF rag, capsule)
- DecapsulateFrag(skB, cF ragiti=1, capsule):

Фернет (Fernet)

Тэгш хэмт шифрлэлт болон баталгаажуулалт хийдэг. 128 битийн түлхүүр бүхий CBC горим дахь AES; PKCS7 ашигладаг. HMAC болон SHA256 баталгаажуулалтад ашигладаг.

Google Cloud Platform

Гуравдагч тал болох сервер проксиг (API) deploy хийж байршуулах шаардлагатай. Google Cloud нь маш олон давуу талтай ба үнэгүй туршиж үзэх 300 долларын эрхтэй тул сонгосон.

2.3 Хөгжүүлэлтийн орчин бэлдэх

Пайтон хэл нь орчин бүрдүүлэхэд хялбар ба виртуал орчин үүсгэж хэрэгтэй сангуудыг татаж суулгах боломжтой. Бүх линукс тархцад пайтон хэл нь суулгалцан байдаг тул хэрэгтэй сангуудыг татаж суулгахад л хангалтай.

Мөн криптографт ашиглхад сайн хэлнүүдийн нэг ба сангууд нь аюулгүй ашиглахад хялбар.[10]

Пайтон хэл дээр хөгжүүлэлт хийхийн тулд прокси дахин шифрлэлтийн сан хэрэгтэй болсон. JHU-MIT Proxy Re-cryptography Library (PRL) нь хэд хэдэн прокси дахин шифрлэлтийн схемийг хэрэгжүүлсэн нээлттэй эхтэй сан юм. Алгоритмуудыг Жонс Хопкинсийн Их Сургууль болон Массачусетсийн Технологийн Их Сургуулийн судлаачид боловсруулсан бөгөөд Жузеппе Атениезе, Кевин Фу, Мэттью Грин, Сюзан Хохенбергер нарын анхны судалгаанд үндэслэсэн. JHU-MIT Прокси дахин шифрлэлтийн сан нь C/C++ хэл дээр бичигдсэн байсан. Пайтон хэлний setup tools болон C хэлний "python.h" санг ашиглан пайтоны сан бичих гэж оролдож үзсэн. JHU-MIT сан нь MIRCL санг ашигладаг.

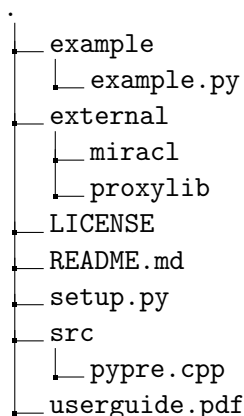
PRL сан нь хоёр схемийг хэрэгжүүлдэг.

- **PRE1:** алгоритм нь Decisional Bilinear Diffie-Hellman Assumption (DBDH) дагуу аюулгүй.
- **PRE2:** алгоритм нь Decisional Bilinear Inversion Assumption (DBDHI)-ийн дагуу аюулгүй.

PRE1 схемийг пайтон сан руу хөрвүүлэх гэж оролдсон.

Ерөнхий бүтэц

- **example** хэсэг тестийн код байрлана.
- **external** хэсэгт PRL сан болон MIRCL сан байна.



ЗУРАГ 2.3: Модулийн файлын бүтэц

- Сангуудыг compile хийж **lib*.a** файл гаргаж авна.
- **setup.py** файл нь тухайн package-ын мэдээлэл байна.
- **pypre.cpp** C++ функцийг нэмэлт сан *.a-аас дуудаж пайтон руу хөрвүүлэх хэсэг.

```

1 import glob
2 import os
3
4 from setuptools import Extension, setup
5
6 os.environ["CC"] = "g++"
7 os.environ["CXX"] = "g++"
8
9 module = Extension(
10     "pypre",
11     language="c++",
12     sources=[f for f in glob.glob('src/*.cpp')] +
13             ['external/proxylib/proxylib_pre1.cpp'],
14     include_dirs=["external/miracl", "external/proxylib"],
15     library_dirs=["external/miracl", "external/proxylib"],
16     libraries=["miracl", "proxylib"],
17     extra_objects=['external/miracl/libmiracl.a', 'external/proxylib/
18                     libproxylib.a']
19 )
20
21 setup(name='pypre',
22       license='Apache',
23       author="Myagmartseren",
24       author_email='myagmartseren7@gmail.com',
25       description='Proxy_Re-encryption_Python_library',
26       url='https://github.com/myagmartseren/pre_python',
27       ext_modules=[module],
28       include_dirs=['external/proxylib', 'external/miracl'],

```

28)

Эх код 2.3: setup.py

pypre.py файл дотор үндсэн C++ кодны санг дуудаж пайтонруу хөрвүүлэх хэсэг байна. Хэрэгтэй толгой файлыг include хийнэ. Proxylib.h болон miracle гэх мэт сангууд нь setup.py хавсарсан сангууд юм.

```

1 #define PY_SSIZE_T_CLEAN
2 #include <zsn.h>
3 #include <miracl.h>
4
5 #include <Python.h>
6 #include <proxylib.h>
7 #include <proxylib_pre1.h>

```

Эх код 2.4: pypre.cpp

Класс бүрийг C-ын struct-ыг ашиглаж зарлаж өгнө.

```

1 typedef struct
2 {
3     PyObject_HEAD
4     ProxyCiphertext_PRE1 *proxyCiphertext_PRE1;
5 } PyProxyCiphertext_PRE1;

```

Эх код 2.5: pypre.cpp

Классын method-уудыг зарлаж өгнө. Буцах утга нь хөгжүүлэх функц ашиглана.

```

1 static PyObject* PyPRE1_generate_params(PyObject *self, PyObject *args)
2 {
3     if (PyTuple_Size(args) !=1){
4         PyErr_SetString(PyExc_RuntimeError, "Should have CurveParams");
5         return Py_None;
6     }
7     PyObject *params = PyTuple_GetItem(args, 0);
8
9     if (!PyCurveParams_Check(params)) {
10         PyErr_SetString(PyExc_RuntimeError, "Failed to cast CurveParams");
11         return Py_None;
12     }
13
14     PyCurveParams *pyParams = (PyCurveParams *)params;
15
16     int result = PRE1_generate_params(*pyParams->params);
17     if (!result) {
18         PyErr_SetString(PyExc_RuntimeError, "Failed to generate params");
19         return PyBool_FromLong(result);
20     }
21
22     Py_INCREF(Py_None);
23     return PyBool_FromLong(result);

```

24 }

Эх код 2.6: generate_params method

Бичсэн method-уудыг класст хамааруулна классдаа нэмж өгөн.

```

1 static PyMethodDef module_methods[] = {
2     {"PRE1_generate_params", PyPRE1_generate_params, METH_VARARGS, "
      Generate_curve_parameters_for_the_PRE1_scheme."},
3     {NULL, NULL, 0, NULL}
4 };

```

Эх код 2.7: generate_paramsmethod – .

Модулийг зарлаж өгнө.

```

1 static struct PyModuleDef module_def = {
2     PyModuleDef_HEAD_INIT,
3     "pypre",
4     0, // m_doc
5     -1, // m_size
6     module_methods, // m_methods
7     /*
8     NULL, // m_reload
9     NULL, // m_traverse
10    NULL, // m_clear
11    NULL // m_free
12    */
13 };

```

Эх код 2.8: Модуль зарлалт

Модулийг import хийх үед классууд зарлаж өгнө.

```

1 PyMODINIT_FUNC PyInit_pypre(void);
2
3 PyObject *PyInit_pypre(void)
4 {
5     extern PyTypeObject PyProxyPK_PRE1Type;
6     if (PyType_Ready(&PyProxyPK_PRE1Type) < 0)
7         return NULL;
8
9     Py_INCREF(&PyProxyPK_PRE1Type);
10    if (PyModule_AddObject(module, "PK_PRE1", (PyObject *)&
      PyProxyPK_PRE1Type) < 0)
11    {
12        Py_DECREF(&PyProxyPK_PRE1Type);
13        Py_DECREF(module);
14        return NULL;
15    }
16
17    extern PyTypeObject PyProxySK_PRE1Type;
18    if (PyType_Ready(&PyProxySK_PRE1Type) < 0)
19        return NULL;
20
21    Py_INCREF(&PyProxySK_PRE1Type);

```

```
22     if (PyModule_AddObject(module, "SK_PRE1", (PyObject *)&  
23         PyProxySK_PRE1Type) < 0)  
24     {  
25         Py_DECREF(&PyProxySK_PRE1Type);  
26         Py_DECREF(module);  
27         return NULL;  
28     }  
29     // PyModule_AddObject(module, "__version__", PyUnicode_FromString(  
30         VERSION));  
31  
32     if (PyErr_Occurred())  
33         PyErr_SetString(PyExc_ImportError, "pypre: init failed");  
34     return module;  
35 }
```

Эх код 2.9: Модуль нь init хийхэд зарлах классууд

PRE1 хөрвүүлж байсан боловч цаг хугацааны хувьд амжихгүй болсон тул pyUmbral санг ашиглахаар шийдсэн.

2.4 Бүлгийн дүгнэлт

Прокси дахин шифрлэх схемийн төрөл, хэрэглээ, давуу тал зэргийг судаллаа. Файл шифрлэх аргуудыг судалж файлыг тэгш хэмт шифрлэлтийн алгоритмаар шифрлэвэл хамгийн хурдан. Тэгш хэмт шифрлэх алгоритмуудаас хамгийн аюулгүй хурлан алгоритм нь AES байсан тул цаашдын хөгжүүлэлт ашиглахаар шийдэв.

БҮЛЭГ 3

Хэрэгжүүлэлтын хэсэг

3.1 Системийн шаардлага

Систем нь түлхүүр үүсгэх файл хадгалах сервер, өгөгдөл эзэмшигч, өгөгдөл хэрэглэгч гэсэн үндсэн гурван хэсгээс тогтоно. Өгөгдөл эзэмшигч бүртгэл үүсгэж өөрийн нийтийн түлхүүр болон хувийн түлхүүрийг үүсгэж авна. Хувийн түлхүүрийг өөрийн төхөөрөмж дээр авч явах ёстой учир десктоп программ бичих хэрэгтэй болсон. Файл шифрлэх болон тайлах үйлдлийг өөрийн төхөөрөмж дээр үйлдэнэ. Мөн ямар нэг ReKey зэрэг хувийн түлхүүртэй холбоотой үйлдлүүдийг хэрлэгч өөрийн төхөөрөмж дээр хийх ёстой.

Системийн оролцогч

- Хэрэглэгч

Системийн тоглогч

- Файл эзэмшигч
- Файл хэрэглэгч

Функцийн шаардлага

Файл эзэмшигчийн функционал шаардлага:

- Системд өөрийн бүртгэлийг үүсгэх
- Файл оруулах, шифрлэх
- Файлыг тайлах хэрлэгч сонгох
- Шифрлэсэн файлыг хуваалцсан хэрлэгчдийн жагсаалт

Файл хэрэглэгчийн функционал шаардлага:

- Системд өөрийн бүртгэлийг үүсгэх
- Өөрт хуваалцсан файлын жагсаалт
- Өөрийн хуваалцсан файлын жагсаалт
- Файлыг хэн хэнтэй хуваалцсан жагсаалт
- Файлыг татаж авах
- Шифрлэсэн файлыг тайлах

Функцийн бус шаардлага

1. Файлыг шифрлэх, шифрийг тайлах хурдан гүйцэтгэдэг байх
2. Хэрэглэгчийн интерфэйс ойлгомжтой энгийн байх.
3. Прокси серверт файлыг шифрлэсэн байдлаар хадгалах, хуваалцах
4. Өөрийн бүртгэлийг ашиглаж нэвтрэх
5. Хувийн түлхүүрийг хэрлэгчийн төхөөрөмж дээр авч явах

Юзкейс диаграмм

Хэрэглэгчид болох файл эзэмшигч болон файл хэрэглэгч нь ямар үйлдлүүдийг системд хийж болохыг харуулсан хэрэглээний диаграмм (Зураг 3.1)-т харуулаа.



ЗУРАГ 3.1: Юзкейс диаграмм

Юзкейс тодорхойлолт

Бүртгүүлэх	
ID	1
Үндсэн тоглогч	Файл эзэмшигч болон файл хэрэглэгч
Тодорхойлолт	Хэрэглэгч шинээр бүртгэл үүсгэх
Өмнөх нөхцөл	Шинэ хэрлэгч байх
Үндсэн урсгал	Хэрэглэгчийн мэдээллийг авч өгөгдлийн санд шинэ хэрэглэгч үүсгэх
Дараах нөхцөл	Хэрэглэгч өөрийн бүртгэлтэй болох нэвтрэх боломжтой болох

Хүснэгт 3.1: Бүртгүүлэх юзкейсний тодорхойлолт

Нэвтрэх	
ID	2
Үндсэн тоглогч	Файл эзэмшигч болон файл хэрэглэгч
Тодорхойлолт	Системд нэвтрэх
Өмнөх нөхцөл	Бүртгэлтэй хэрлэгч байх
Үндсэн урсгал	Хэрлэгчийн нууц үг бүртгэл таарч байгааг шалган нэвтрүүлэх
Дараах нөхцөл	Бусад хэрэглээ рүү хандах боломжтой болох

Хүснэгт 3.2: Нэвтрэх юзкейсний тодорхойлолт

Файл шифрлэх	
ID	3
Үндсэн тоглогч	Файл эзэмшигч
Тодорхойлолт	Файл эзэмшигч файлыг шифрлэх
Өмнөх нөхцөл	Систем нэвтэрсэн байх
Үндсэн урсгал	<ul style="list-style-type: none"> • Файлыг сонгох • Санамсаргүй байдлаар түлхүүр үүсгэх • Файлыг тухайн түлхүүрээр шифрлэх
Дараах нөхцөл	Тухайн төхөөрөмж дээр шифрлэгдсэн байх

Хүснэгт 3.3: Файл шифрлэх юзкейсийн тодорхойлолт

Шифрлэсэн файл серверт байршуулах	
ID	4
Үндсэн тоглогч	Файл эзэмшигч
Тодорхойлолт	Шифрлэсэн файлыг серверт хуулна
Өмнөх нөхцөл	Файлыг шифрлэсэн байх
Үндсэн урсгал	<ul style="list-style-type: none"> • Шифрлэсэн файлыг серверт илгээж • Файлын мэдээллийг өгөгдлийн санд нэмэх
Дараах нөхцөл	Файлын талаар мэдээллийг авах боломжтой болох

Хүснэгт 3.4: Шифрлэсэн файл серверт байршуулах юзкейсийн тодорхойлолт

Файл хуваалцах	
ID	5
Үндсэн тоглогч	Файл эзэмшигч
Тодорхойлолт	Файл эзэмшигч файл хуваалцах хүнийг сонгох
Өмнөх нөхцөл	Файлыг серверт байршуулсан байх
Үндсэн урсгал	<ul style="list-style-type: none"> • Файл хуваалцах хүний нийтийн түлхүүрийг авах • Дахин шифрлэх түлхүүр үүсгэх • Файлыг дахин шифрлэх
Дараах нөхцөл	Шифрлэсэн файлыг төхөөрөмж дээр тайлхад бэлэн болох

Хүснэгт 3.5: Файл хуваалцах юзкейсийн тодорхойлолт

Файл татах	
ID	6
Үндсэн тоглогч	Файл хэрэглэгч
Тодорхойлолт	Дахин шифрлэсэн файлыг татаж авах
Өмнөх нөхцөл	Файлыг хуваалцсан байх
Үндсэн урсгал	Файлыг татаж авах
Дараах нөхцөл	Файлыг тайлхад бэлэн болох

Хүснэгт 3.6: Файл татах юзкейсийн тодорхойлолт

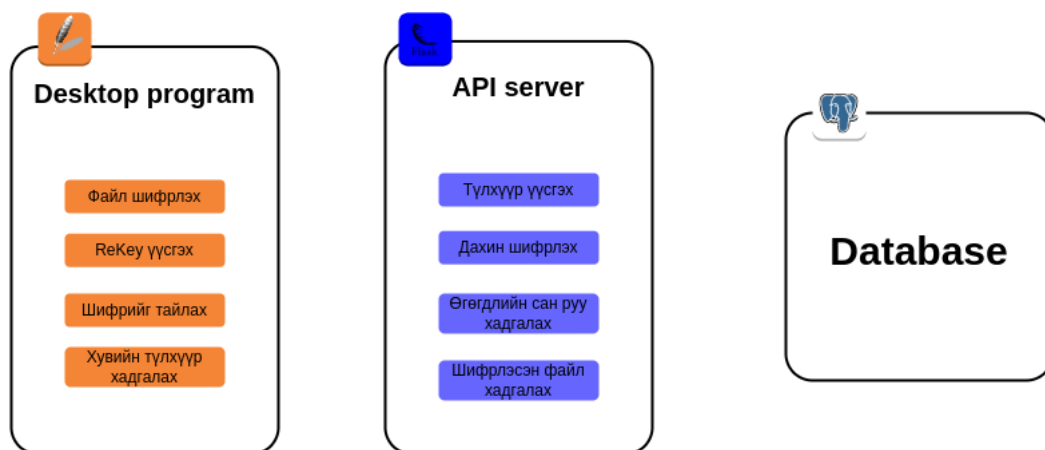
Шифрлэсэн файлын тайлах	
ID	7
Үндсэн тоглогч	Файл хэрэглэгч
Тодорхойлолт	Татаж авсан шифрлэсэн файлыг тайлах
Өмнөх нөхцөл	Файлыг хуваалцсан байх
Үндсэн урсгал	<ul style="list-style-type: none"> • Файлын түлхүүрийг өөрийн хувийн түлхүүрээр тайлах • Файлын түлхүүрээр файлыг тайлах
Дараах нөхцөл	Файлыг унших боломжтой болгох

Хүснэгт 3.7: Шифрлэсэн файлын тайлах юзкейсийн тодорхойлолт

3.2 Системийн загвар

Файл хуваалцах үйлчилгээ нь хоёр хэсгээс тогтоно. Хэрэглэгчийн интерфэйс нь десктоп программ ба тухайн төхөөрөмж дээр татаж суулгана. Десктоп программ өөр дээр шифрлэх болон шифрийг тайлах үйлдлийг хийнэ. Сервер хэрэгтэй мэдээллийг хангаж хоёр хэрэглэгчийг холбож өгнө. Файлыг шифрлэхдээ тэгш хэмт шифрлэх алгоритм ашигласнаар илүү хурдан шифрлэх боломжтой. Тэгш хэмт шифрлэх алгоритмуудаас хамгийн сайн нь AES алгоритм.

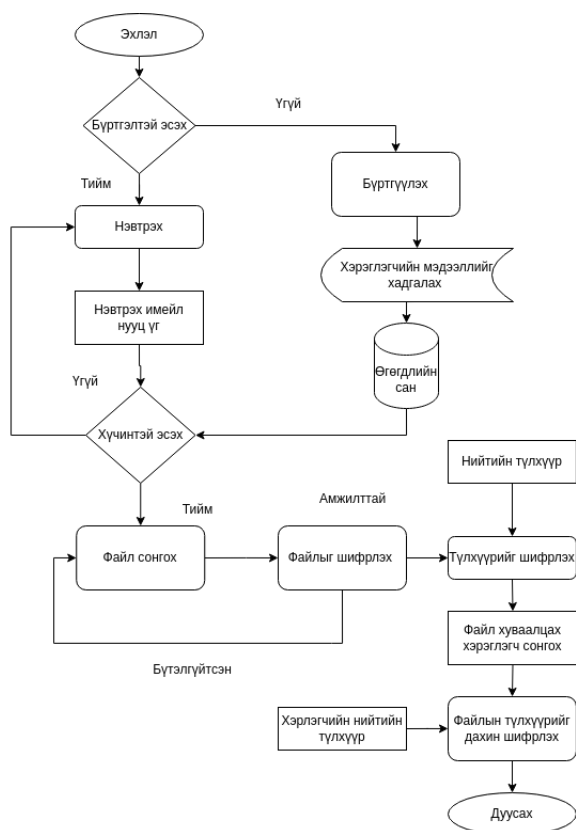
Прокси дахин шифрлэх схемийг зөвхөн тэгш хэмт шифрийн түлхүүр дээр ашиглана.



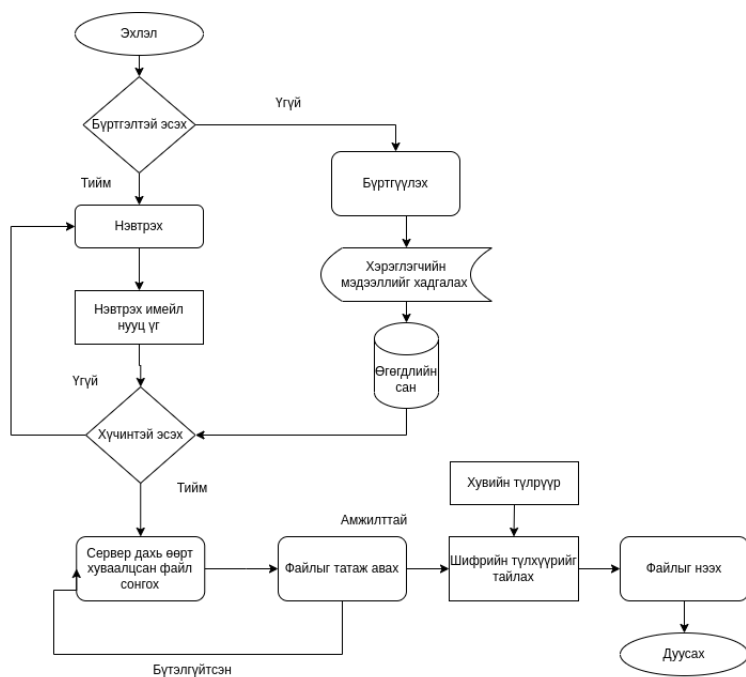
ЗУРАГ 3.2: Ерөнхий загвар

Үйл ажилгааны диаграмм

Файл эзэмшигч болон файл хэрэглэгчийн системд нэвтрэхээс файл хуваалцах хүртэл ерөнхий үйл ажиллагааны диаграммыг Зураг 3.4 3.3-т харуулав.



ЗУРАГ 3.3: Файл эзэмшигчийн үйл ажилгааны диаграмм

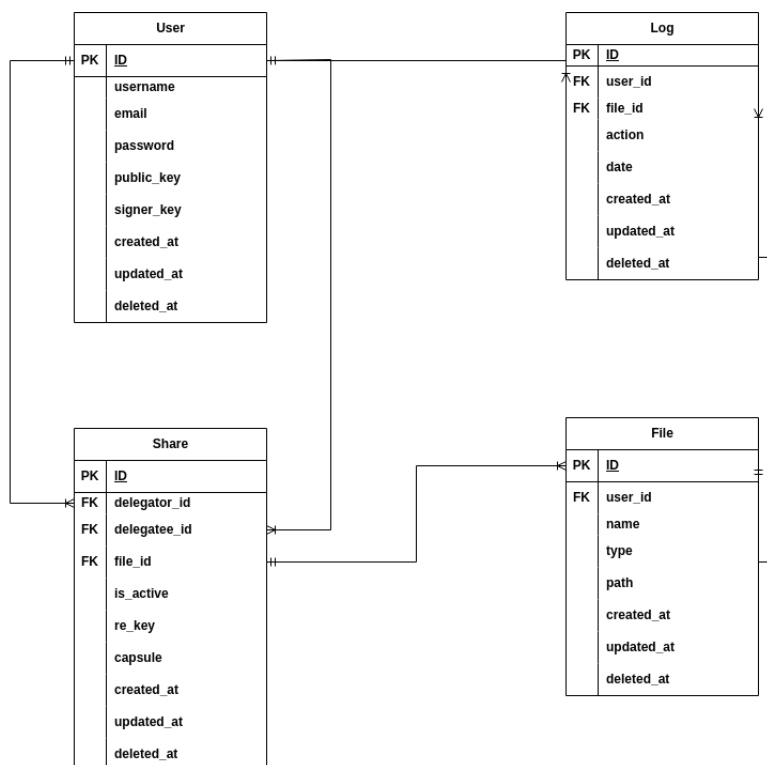


ЗУРАГ 3.4: Файл хэрэглэгчийн үйл ажилгааны диаграмм

Өгөгдлийн сангийн бүтэц

Өгөгдлийн сан нийт дөрвөн хүснэгттэй. Хүснэгт бүрд created_at, update_at, deleted_at гэсэн талбарууд байгаа GORM-аас санаа аван эдгээр талбаруудыг нэмсэн.

- **created_at** нь тухайн мөр үүсэх үед цагийг бүртгэж авдаг
- **update_at** нь тухайн мөр өөрчлөгдөх хийх цагийг бүртгэж авдаг
- **deleted_at** нь тухайн мөрийг устгаагүй байхад хоосон байдаг ба устгасан цагийг бүртгэж авдаг.



ЗУРАГ 3.5: Өгөгдлийн сангийн бүтэц

- **User** хэрэглэгчийн хүснэгт. Хэрлэгчийн хувийн түлхүүрийг өгөгдлийн санд хадгалахгүй ба файл хэлбэрээр тухайн төхөөрөмж дээр хадгалж явна.
 - ID
 - username
 - email
 - password
 - public_key
 - signer_key
- **Share** хүснэгт нь файлыг олон хүнтэй хуваалцах боломжтой болгоно.
 - ID
 - delegator_id
 - delagatee_id

- file_id
- is_active
- re_key
- capsule

- **File** Файлын мэдээлэл мөн түлхүүрийг хадгалана.

- user_id
- name
- type
- path

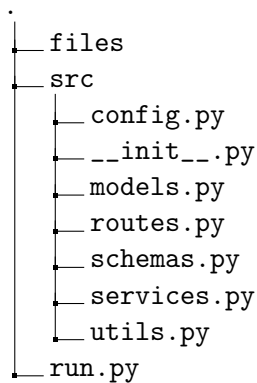
- **Log** лог хадгалах зориулалтай хүснэгт

- user_id
- file_id
- action
- date

3.3 Системийн хөгжүүлэх

Систем хоёр хэсгээс тогтоно.

API сервер буюу гуравдагч хагас итгэмжлэгдсэн талын файлын бүтэц. API нь хэрлэгчийн түлхүүрийг үүсгэнэ хэрлэгчдийг холбож өгнө. Шифрлэгдсэн файлыг хадгалах үүрэгтэй.



ЗУРАГ 3.6: API сервисийн файлын бүтэц

Хэрэглэгчийн интерфейс буюу десктоп программ хэсгийн файлын бүтэц. **api** серверт хүсэлт явуулах хэсэг. **views** хэсэгт tkinter-г ашиглах цонх үүсгэх классууд байна.

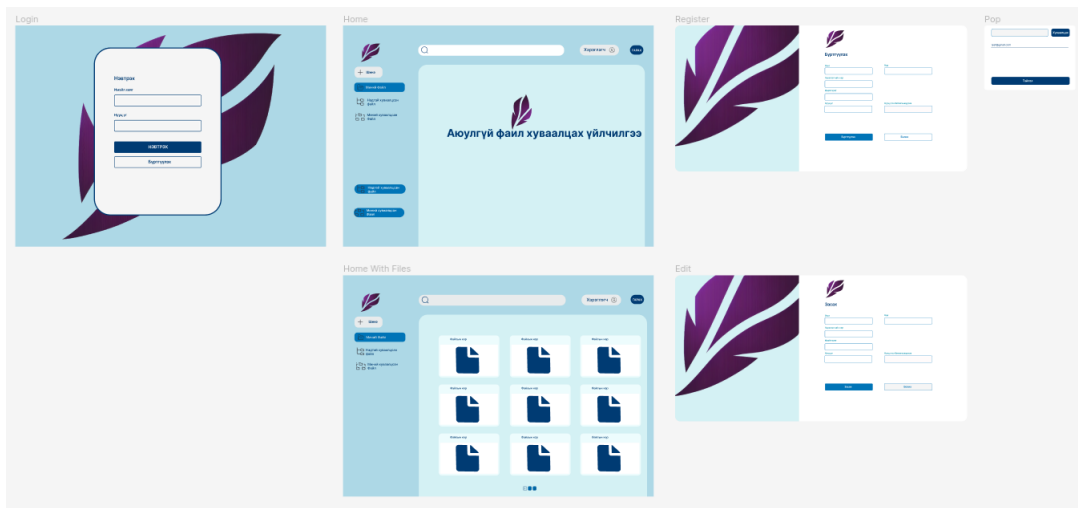
```
.
├── api
│   ├── __init__.py
│   ├── auth.py
│   ├── crud.py
│   └── encryption.py
├── main.py
├── models.py
├── requirements.txt
├── utils.py
├── setup.py
└── views
    ├── __init__.py
    ├── home.py
    ├── login.cpp
    ├── register.cpp
    ├── edit.cpp
    └── pop.cpp
```

ЗУРАГ 3.7: Десктоп файлын бүтэц

Хэрэглэгчийн интерфейс

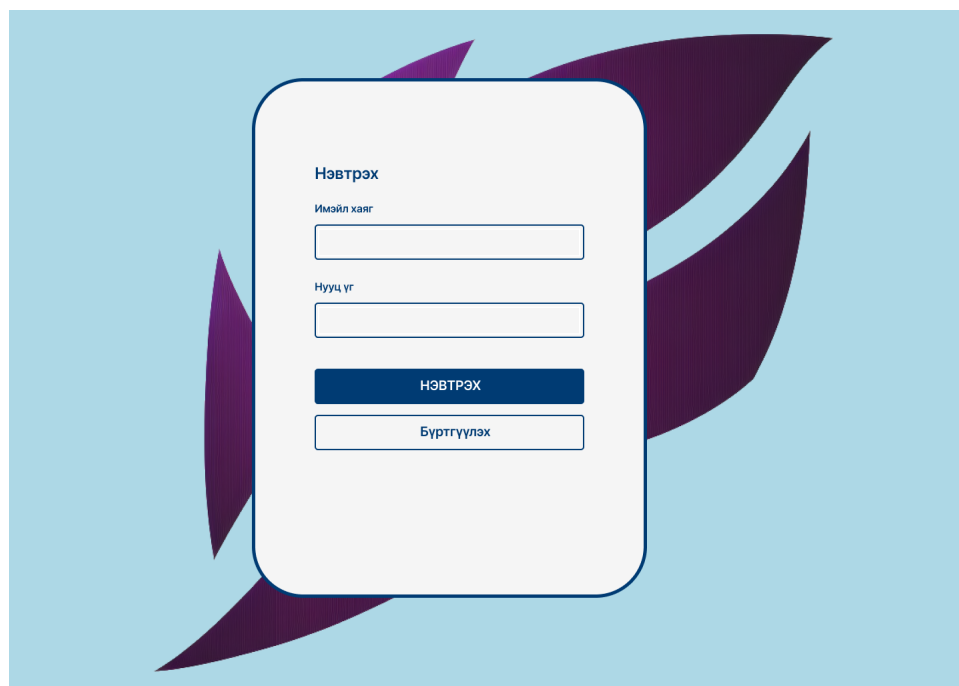
Хэрэглэгчийн интерфейсийн хувьд зөвхөн десктоп программ интерфейстэй байх ба сервер буюу прокси нь ямар нэг график хэрэглэгчийн интерфейс байхгүй зөвхөн API байна.

Figma дээр гаргасан загвар. (Зураг 3.8)



ЗУРАГ 3.8: Figma загвар

Хэрэглэгч өөрийн имэйл хаяг болон нууц үг ашиглан нэвтрэх ч орно (Зураг 3.9).



Нэвтрэх

Имэйл хаяг

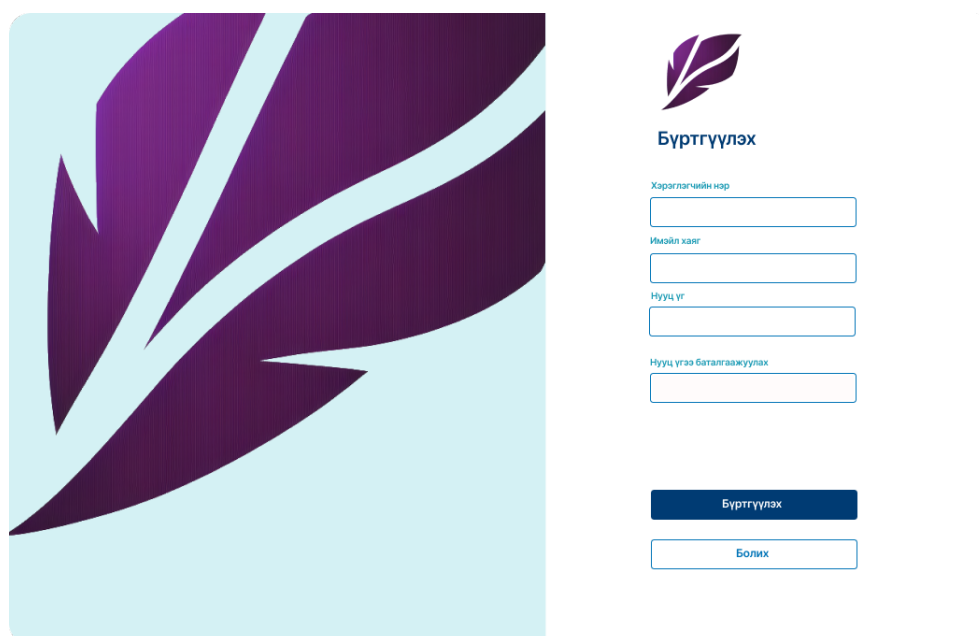
Нууц үг


НЭВТРЭХ

Бүртгүүлэх

ЗУРАГ 3.9: Нэвтрэх цонх

Хэрэглэгч өөрийн мэдээлэл болон нууц үг хэрлэгчийн нэрийг оруулж бүртгэл үүсгэнэ. (Зураг 3.10)





Бүртгүүлэх

Хэрэглэгчийн нэр

Имэйл хаяг

Нууц үг

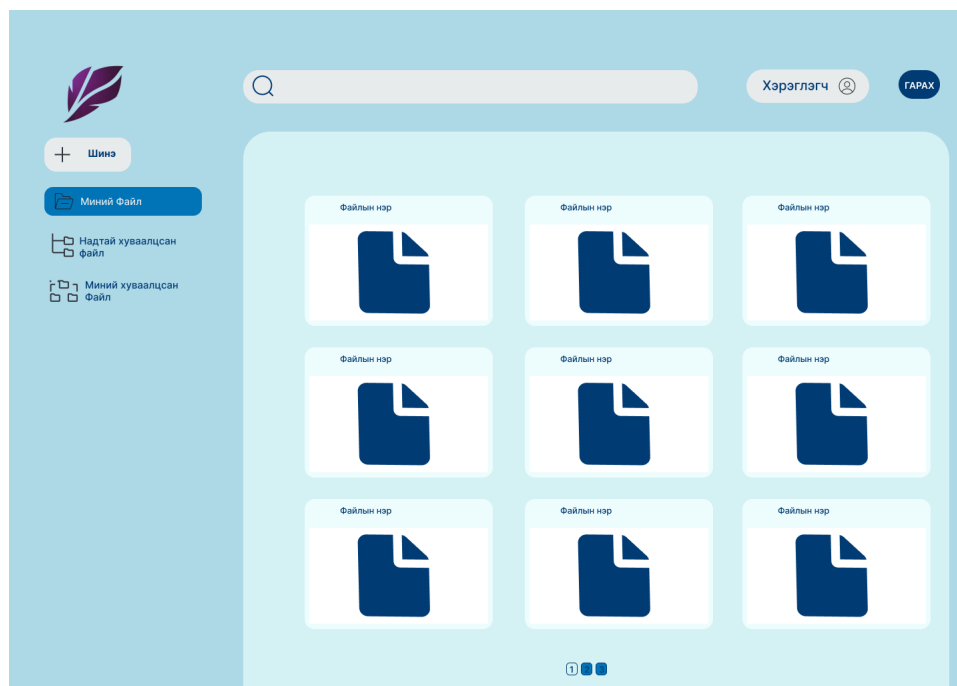
Нууц үгээ баталгаажуулах

Бүртгүүлэх

Болих

ЗУРАГ 3.10: Бүртгүүлэх цонх

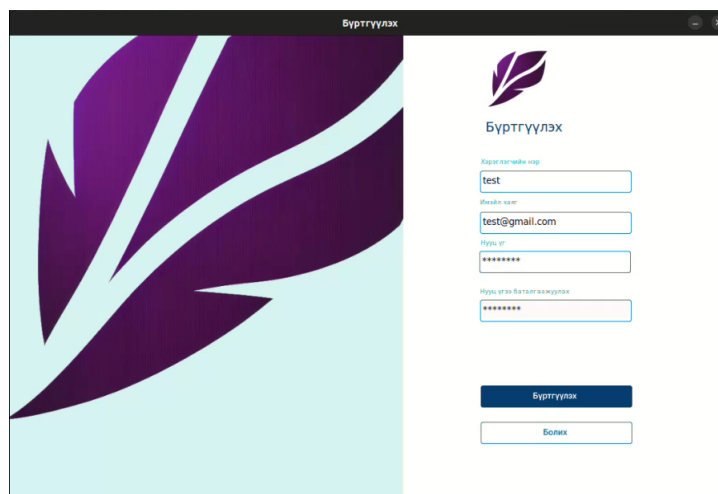
Үндсэн цонх хэрлэгч өөрийн файлыг байршуулах боломжтой. Мөн өөрт хуваалцсан файлуудын жагсаалт болон файлыг тайлж хадгалах боломжтой. (Зураг 3.11)



ЗУРАГ 3.11: Үндсэн цонх

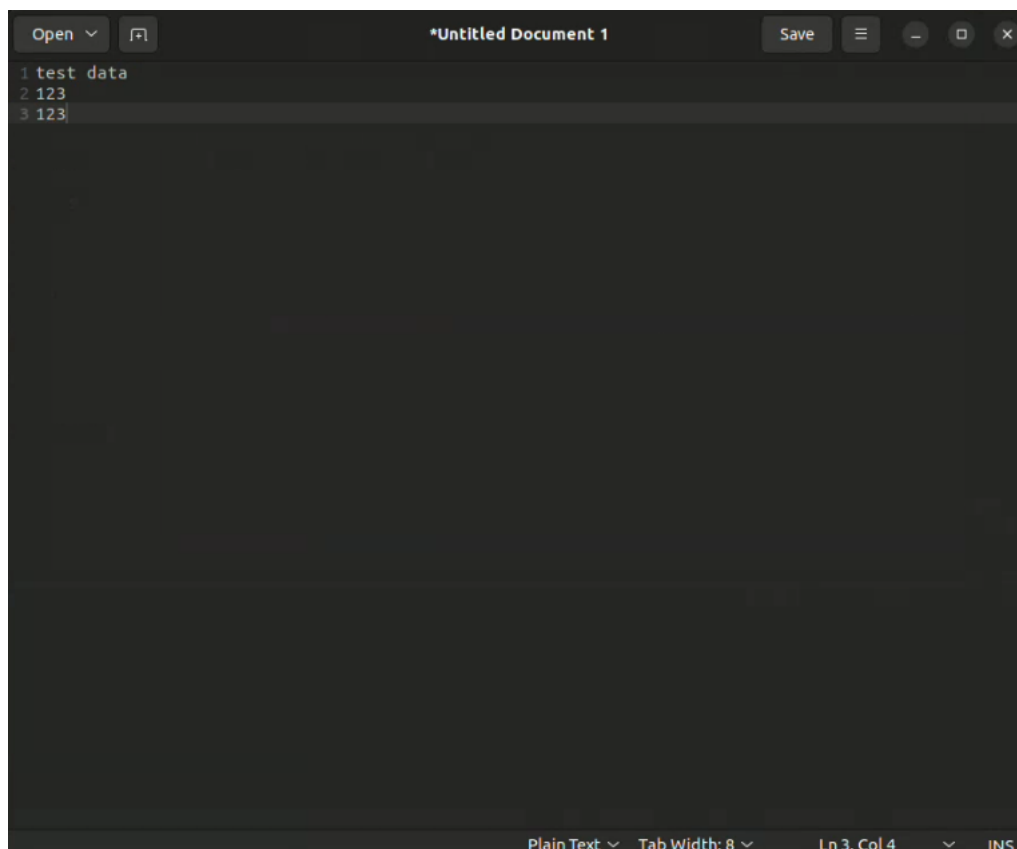
3.4 Файл хуваалцах системийг турших

Туршилтад гурван хэрлэгч ашигласан. user болон test нэртэй хэрлэгчдийг бүртгүүлнэ (Зураг 3.12).



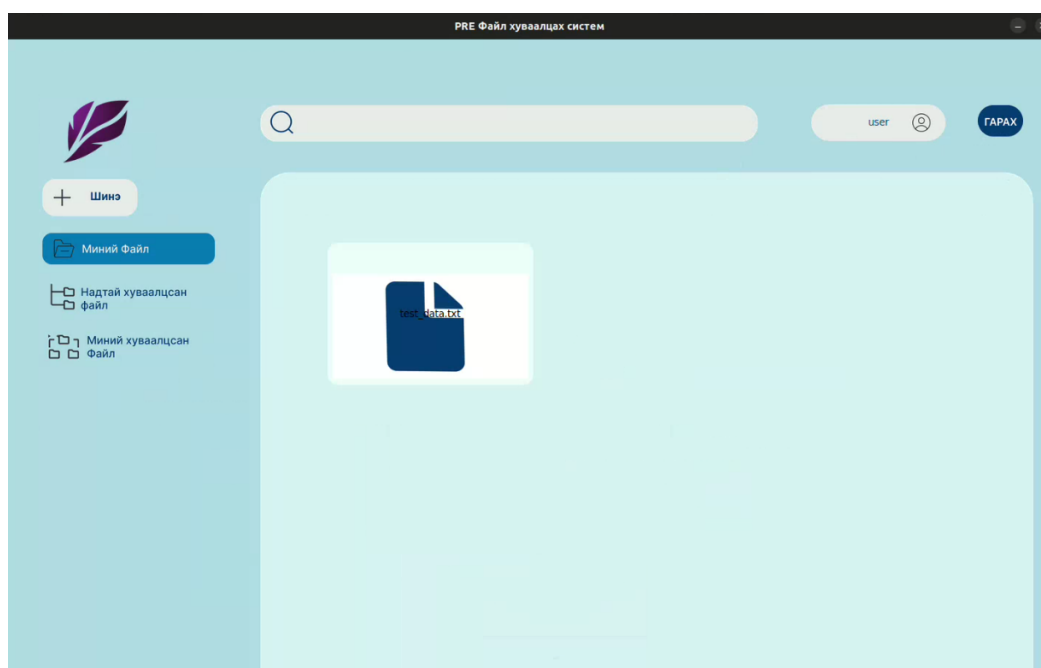
ЗУРАГ 3.12: Бүртгүүлэх цонх жишээ

Жишээ текст файл үүсгэж өгнө Зураг (3.13). Зураг болон бинари файл ч мөн адил болно. Энэ тохиолдолд хурдан үүсгэх зорилгоор текст файл үүсгэв.



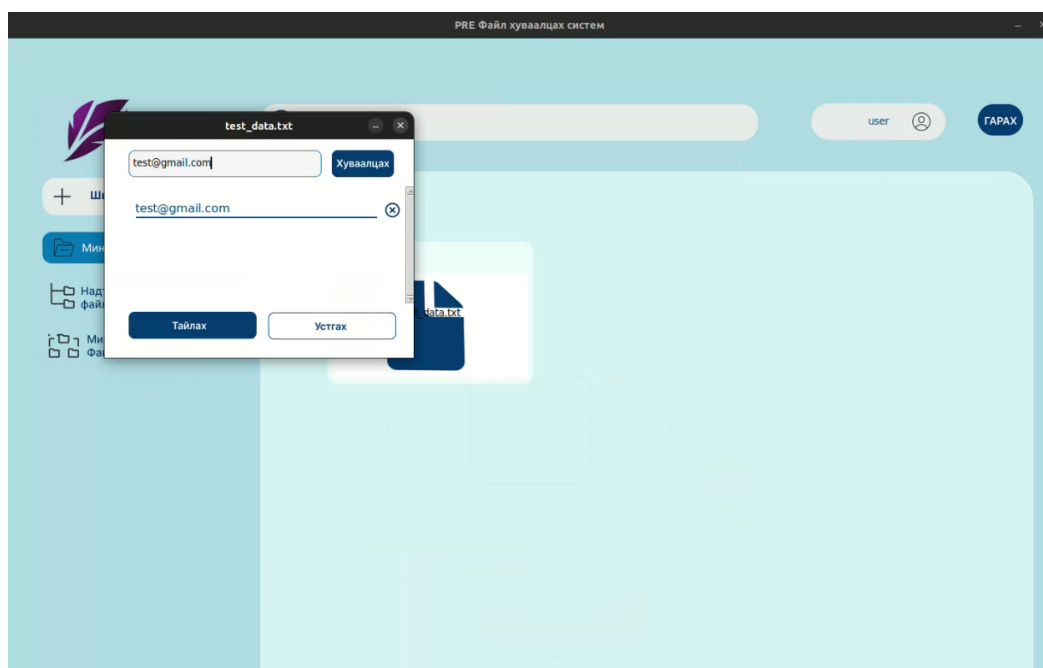
Зураг 3.13: Жишээ өгөгдөл үүсгэнэ

Файлыг серверт байршуулсны дараа хэрэглэгчийн интерфейс харагдах байдал.
Зураг (3.14)



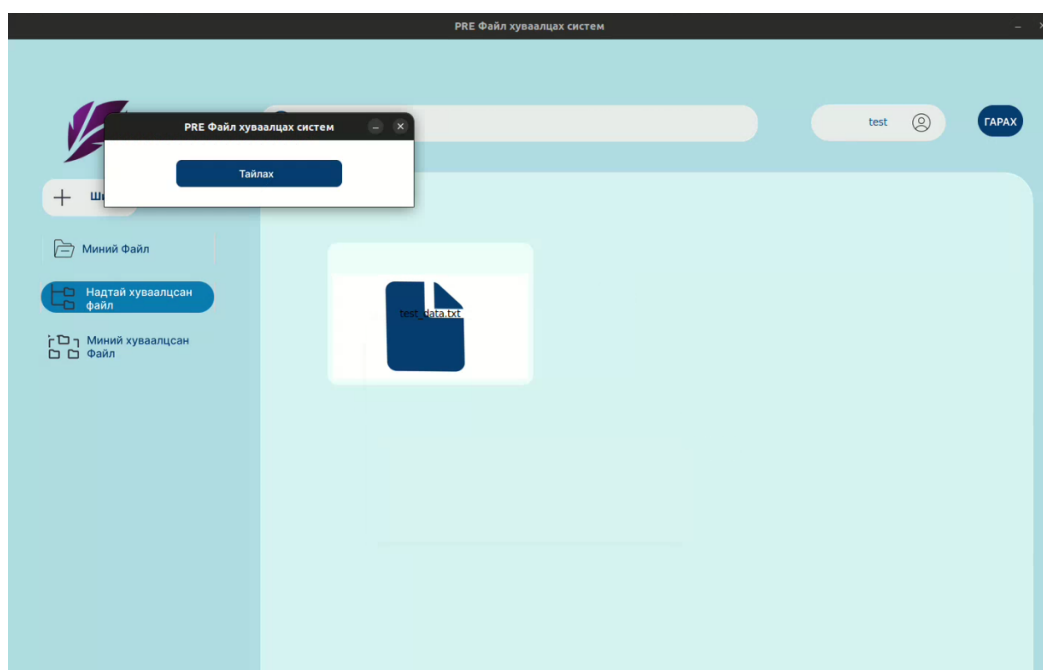
Зураг 3.14: Байршуулсан файл

Байршуулсан дээр дархад хуваалцах цонх гарж ирэх ба хэрэглэгчийн имэйл хаягийг оруулж хуваалцана. Зураг (3.15)



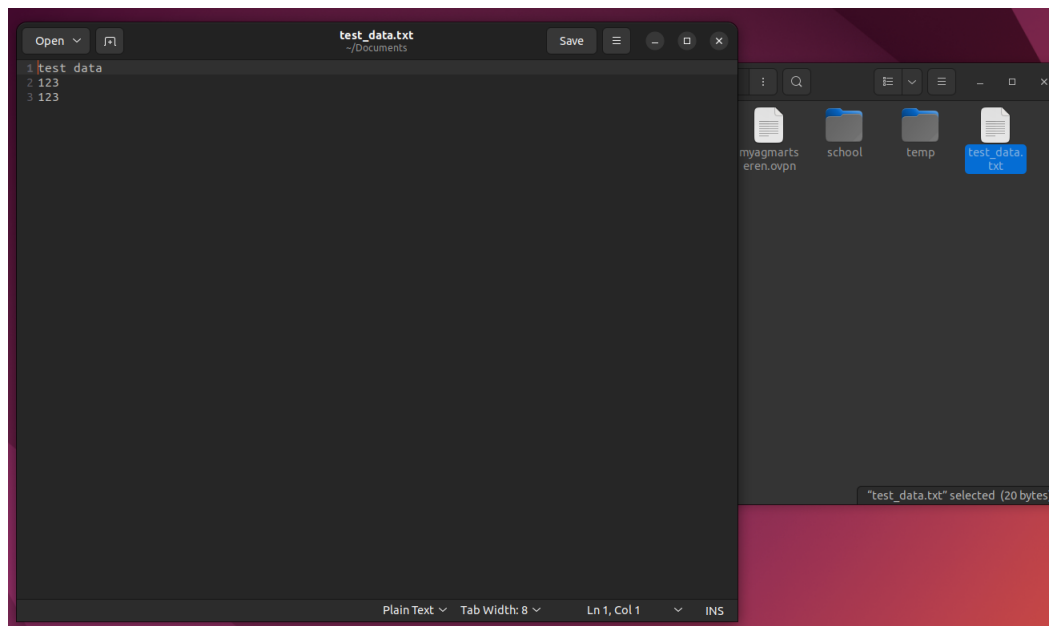
ЗУРАГ 3.15: Хуваалцах цонх

Файлын серверт хуулсны дараа test хэрэглэгчийн бүртгэлээр нэвтрэч орон "Надтай хуваалцсан файл" товч дээр дархад хуваалцсан файл гарч ирнэ (Зураг 3.16).



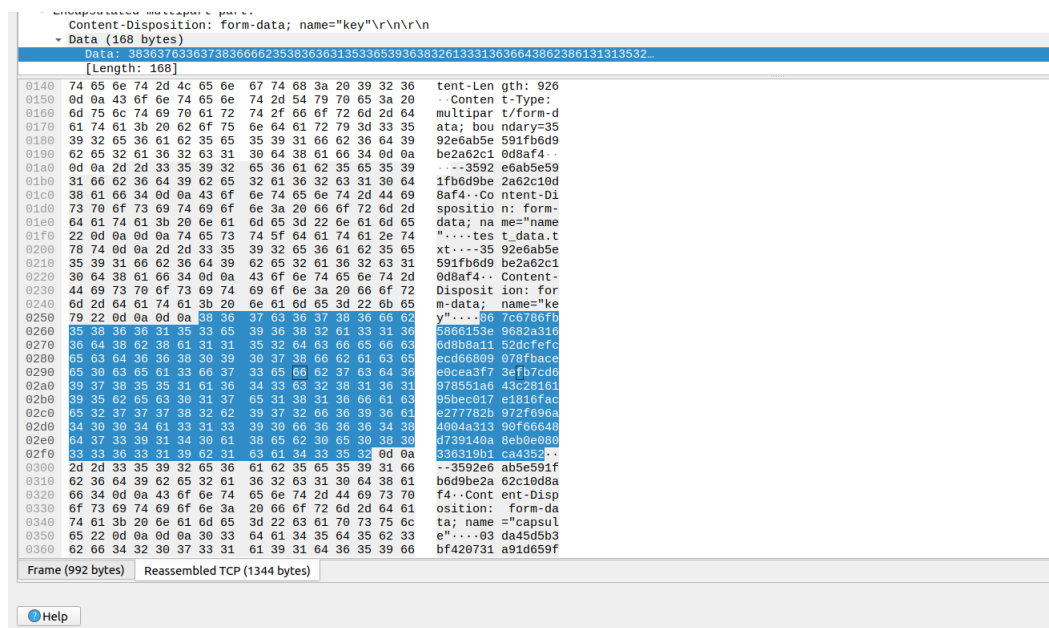
ЗУРАГ 3.16: Хуваалцсан файл

Хуваалцсан файлыг тайлж хадгалах замыг сонгож шифрийг тайлсны дараа харагдах байдал. 3.17



ЗУРАГ 3.17: Хуваалцсан файл шифрлэлтийг тайлсаны дараа

Wireshark-р файл байршуулж буй пакетыг барж үзэхэд файлын нэр болон бусад зүйл шифрлээгүй файлын түлхүүр файлыг шифрлэж явуулж байна (Зураг 3.18).



ЗУРАГ 3.18: Wireshark барисан пакет

3.5 Дүгнэлт

Энэ бүлэгт өмнө судалсан судалгааны дагуу прокси дахин шифрлэх схемийг ашиглан файл хуваалцах системийн схемийг гаргаж үйл ажилгааны болон хөгжүүлэлтийн явцийг тайлбарлалаа.

БҮЛЭГ 4

Ерөнхий дүгнэлт

Дүгнэлт

Мэдээллийн эрин зуунд их өгөгдөл хуваалцах нь маш олон давуу талтай. Өгөгдөл хуваалцах олон ашиг тустай ч өгөгдлийн аюулгүй байдал өгөгдлийг аюулгүй хуваалцах чухал юм.

Энэ дипломын ажлаар өгөгдлийг хэрхэн хуваалцах болон өгөгдлийн аюулгүй байдлын талаар судлав. Өгөгдлийг шифрлэж хуваалцах аюулгүй байдлын хувьд олон давуу талтай. Тэгш хэмт шифрлэлт нь файлыг шифрлэхэд тэгш хэмт бус шифрлэлтээс илүү хурдан байдаг. Гэвч түлхүүрийг аюулгүй хуваалцах мөн адил чухал. Тэгш хэмт бус шифрлэлт нь илүү аюулгүй учир түлхүүр хуваалцах ашиглахад илүү тохиромжтой. Эдгээрийг ашиглан файл хуваалцах системийг энийг хялбар хийхэд прокси дахин шифрлэх схемийг ашиглан файл хуваалцах туршилтын систем хөгжүүлэв.

Ном зүй

- [1] Amazon Web Services. *What Is Data Sharing?* URL: <https://aws.amazon.com/what-is/data-sharing>.
- [2] Wikipedia. *File sharing*. URL: https://en.wikipedia.org/wiki/File_sharing.
- [3] *Why data security is vital for the well-being of any enterprise today*. URL: <https://www.ibm.com/topics/data-security>.
- [4] Wikipedia. *Identity-based encryption*. URL: https://en.wikipedia.org/wiki/Identity-based_encryption.
- [5] Wikipedia. *Attribute-based encryption*. URL: https://en.wikipedia.org/wiki/Attribute-based_encryption.
- [6] Wikipedia. *Homomorphic encryption*. URL: https://en.wikipedia.org/wiki/Homomorphic_encryption.
- [7] Madhumita Panda. “Performance Analysis of Encryption Algorithms for Security”. **in**(2016).
- [8] Wikipedia. *Encrypting File System*. URL: https://en.wikipedia.org/wiki/Encrypting_File_System.
- [9] Giuseppe Ateniese **and others**. “Improved proxy re-encryption schemes with applications to secure distributed storage”. **in**(2005).
- [10] analyticsinsight.net. *Watch out for these top 10 programming languages for cryptography in 2023 to protect your data*. URL: <https://www.analyticsinsight.net/top-10-programming-languages-for-cryptography-in-2023/>.