DID Security Analysis Progress

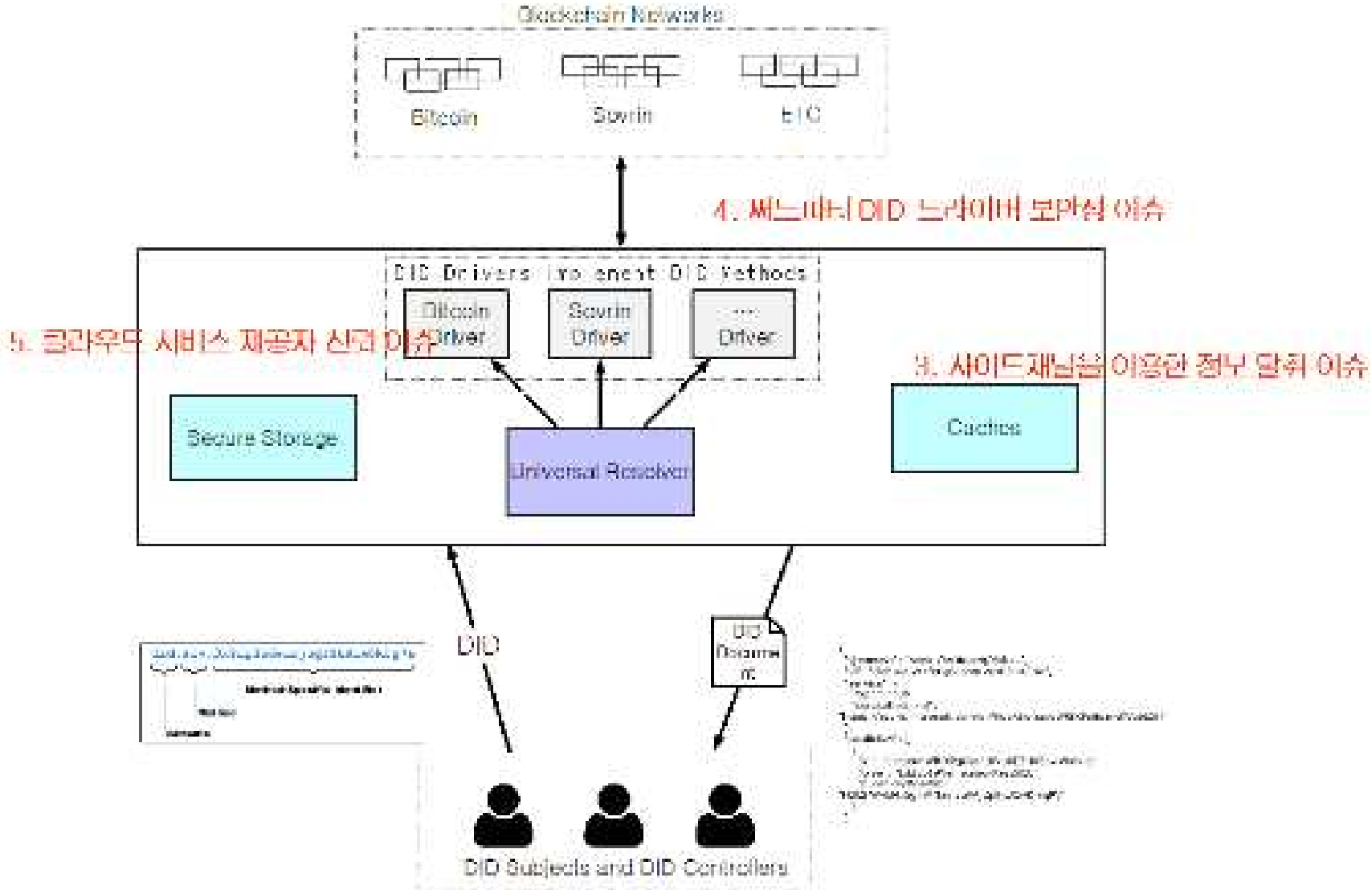
siwon heo

Systems Security Lab @ SKKU

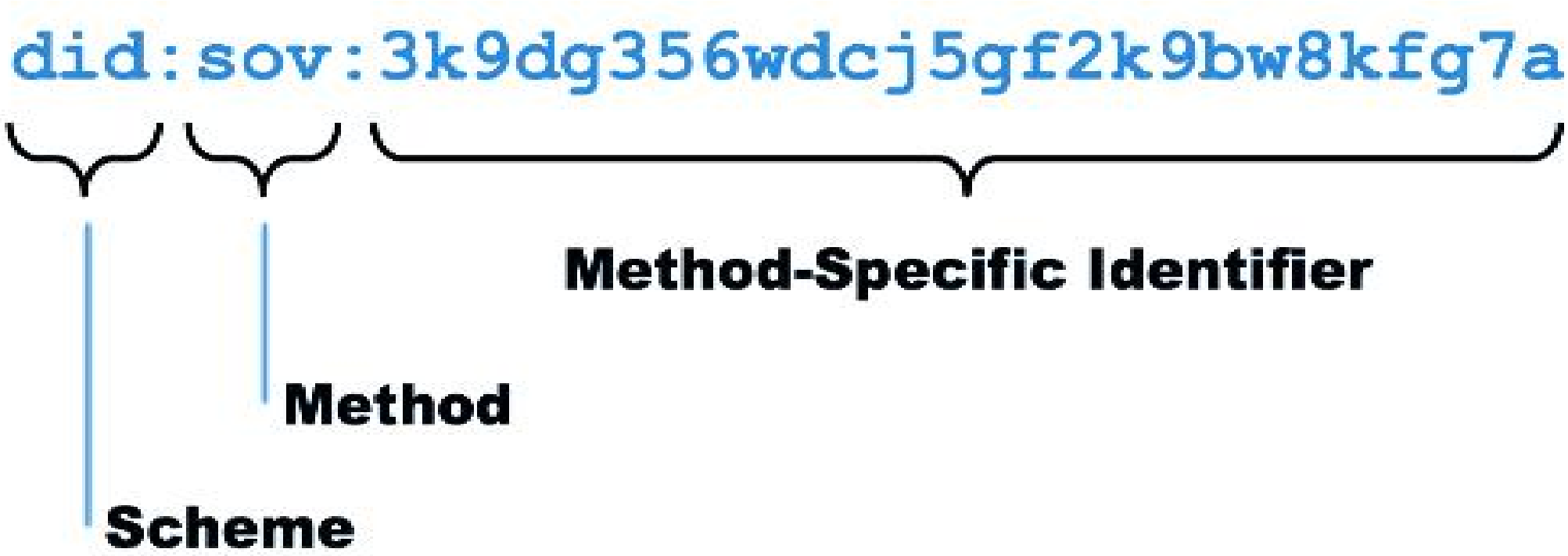
What is DID?

- Decentralized IDentity
- Identification of user without any central identifier
- How?

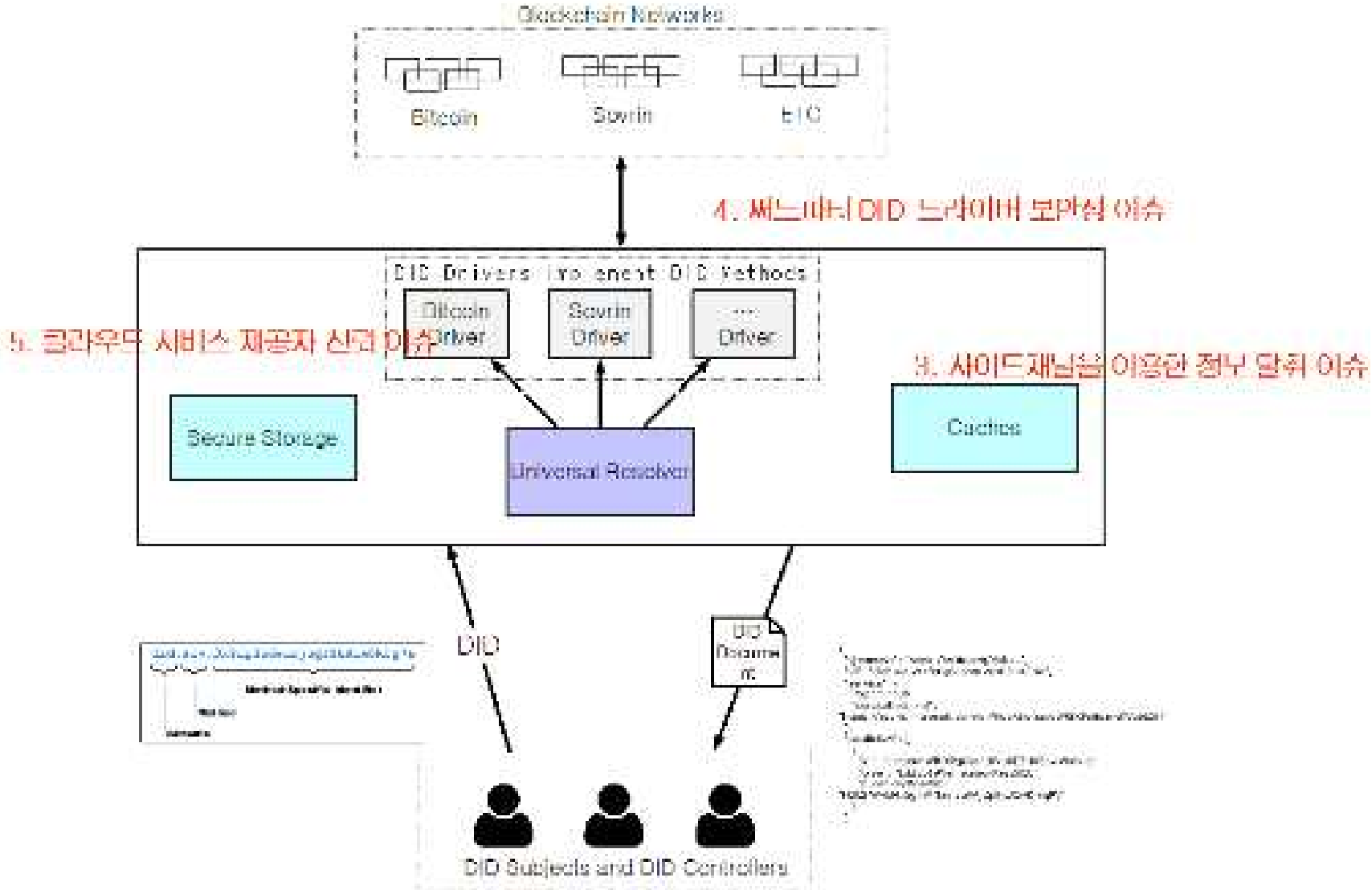
DID Analysis



DID Scheme


did:sov:3k9dg356wdcj5gf2k9bw8kfg7a
Method-Specific Identifier
Method
Scheme

DID Security Analysis



```
→  ~ time curl -I -X GET http://localhost:8080/1.0/identifiers/did:git:gjgd;
HTTP/1.1 404 Not Found
Date: Fri, 04 Dec 2020 05:52:48 GMT
Access-Control-Allow-Origin: *
Transfer-Encoding: chunked
Server: Jetty(9.4.18.v20190429)

curl -I -X GET http://localhost:8080/1.0/identifiers/did:git:gjgd  0.01s user 0.01s system 4
2% cpu 0.056 total
```

```
uni-resolver-web_1                  | [qtp1012194796-37] INFO servlet.ResolveServlet (doGet) -
Resolve result for did:git:gjgd: {"didDocument":null,"content":null,"contentType":null,"didR
esolutionMetadata":{"duration":25,"identifier":"did:git:gjgd","didUrl":{"didUrlString":"did:
git:gjgd","did":{"didString":"did:git:gjgd","method":"git","methodSpecificId":"gjgd","parseT
ree":null,"parseRuleCount":null},"path":"","query":null,"fragment":null,"parameters":{},"par
seTree":null,"parseRuleCount":null}},"didDocumentMetadata":{}}
```

Side channel information leak

- Blockchain computation is considerably slow
- The use of cache is inevitable for some services
- However, the built-in cache in DID resolver is implemented by default cache package of NodeJS

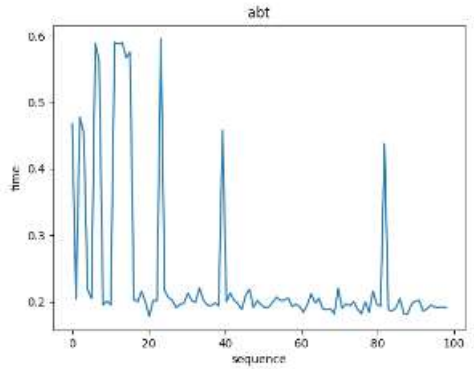
==> vulnerable in cache timing attack

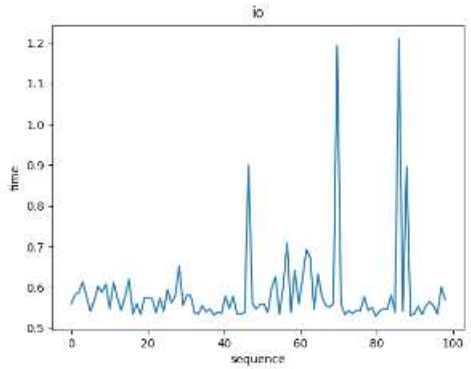
Side channel information leak

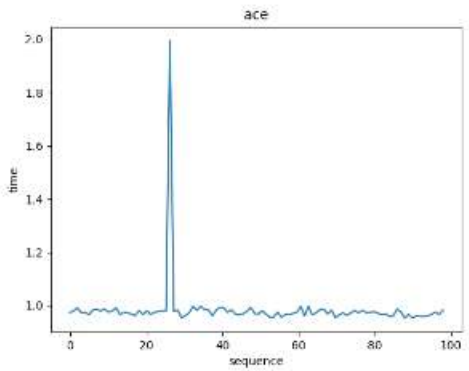
```
export type DIDResolver = (
  did: string,
  parsed: ParsedDID,
  resolver: Resolver
) => Promise<null | DIDDocument>
export type WrappedResolver = () => Promise<null | DIDDocument>
export type DIDCache = (
  parsed: ParsedDID,
  resolve: WrappedResolver
) => Promise<null | DIDDocument>

interface ResolverRegistry {
  [index: string]: DIDResolver
}

export function inMemoryCache(): DIDCache {
  const cache: Map<string, DIDDocument | null> = new Map()
  return async (parsed, resolve) => {
    if (parsed.params && parsed.params['no-cache'] === 'true')
      return await resolve()

    const cached = cache.get(parsed.did)
    if (cached !== undefined) return cached
    const doc = await resolve()
    if (doc !== null) {
      cache.set(parsed.did, doc)
    }
    return doc
  }
}

export function noCache(
  parsed: ParsedDID,
  resolve: WrappedResolver
): Promise<null | DIDDocument> {
  return resolve()
}
```

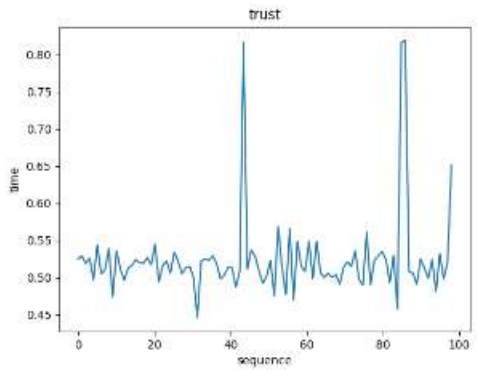
- The implementation of cache is simple
- We can easily see there is no protection for timing attack

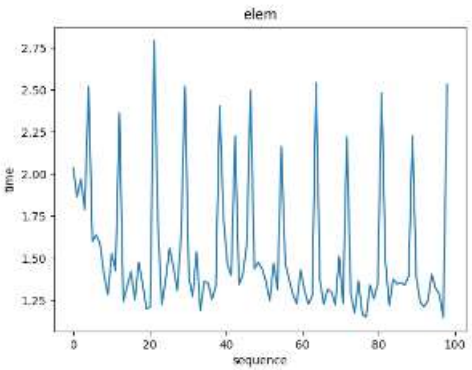
Side channel information leak











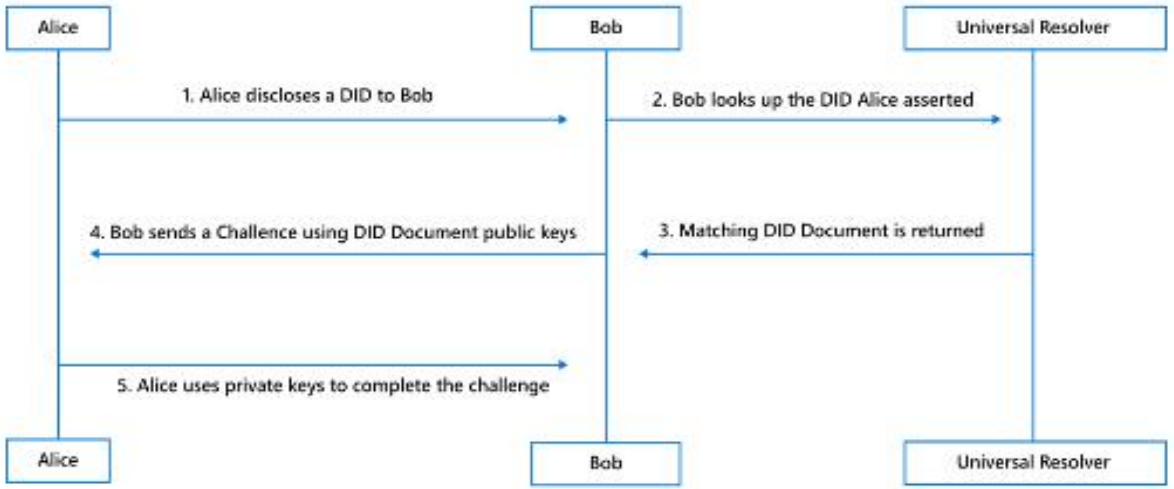
Side channel information leak

- The services that uses default did-resolver caching scheme:

NOTA BENE ceramic ...

- If these services are location-sensitive, user's privacy can be attacked.

Authenticating DID Queries



- DID is public; If Bob knows a DID, he can gather information of DID owner from mutiple services
- => may specify DID owner from a DID

Capability

- The authentication of DID Query is needed to solve these issues
- Capability : DID-owner’s private key should sign the query, and only signed request can be responsed

Problem

- How to implement capability without any centralized identifier?
- Using DID concept again would be circular reasoning
- Zero-knowledge proof may help

On progress

- Blockchain device driver analysis
- Capability
- ...