웹 기반 분산식별자 해석에서 클라이언트 no-cache 지시자의 실효성 측정

김건민*, 김경백**

요 약

분산식별자(Decentralized Identifier, DID)의 did:web은 HTTPS 기반으로 DID 문서를 조회하므로, 중간 HTTP 캐시에 의해 키 회전(문서 갱신) 이후에도 갱신 전 문서가 일정 시간 반환될 수 있다. 본 논문은 DID Resolution의 noCache 해석 옵션과 구분하여, 클라이언트의 HTTP 요청 지시자가 이러한 stale DID 문서 노출을 실제로 줄일 수 있는지를 측정한다. 주요 HTTP 캐시 구현으로 구성한 테스트베드에서 분석한 결과, 일부 캐시에서는 지시자 적용 후 갱신 전 문서의 노출이 제거된 반면, 다른 캐시에서는 지시자 적용 여부에 따른 차이가 나타나지 않았다. 또한 실운영 did:web 공유 캐시에서도 no-cache 요청에 따른 응답 Age 감소가 거의 관측되지 않았다. 따라서 클라이언트 no-cache 지시자의 효과는 중간 캐시의 구현과 구성에 의존하며, 해당 지시자만으로는 did:web 전달 경로 전반의 문서 신선도를 보장하기 어렵다.

Ⅰ. 서 론

분산식별자(DID) [1] [5] 의 did:web은 HTTPS로 문서를 조회하므로 [3], 키 회전·문서 갱신 이후에도 HTTP 캐시 [4] 가 갱신 전 문서를 제공할 수 있다. 이는 웹 PKI 인증서 폐기에서도 관찰된 신선도 지연과 유사하나 [6], DID Resolution noCache [2] 와 HTTP Cache-Control: no-cache [4] 는 다른 계층의 통제이다. 기존 DID 실증 연구는 조회 프라이버시 등을 다루었으나 [7], 본 논문은 의존 당사자가 중간 캐시에 직접 보낼 수 있는 후자의 실효성을 측정한다. nginx·Varnish·Squid 테스트베드와 Age가 관측된 실운영 배포 2곳에서 갱신 전 문서 노출과 응답 Age 변화를 비교하였다. (1) 두 no-cache의 계층 구분, (2) 동일 origin 아래 캐시 구성별 E(δ), (3) 실운영 공유 캐시에서의 관측 결과를 보고한다.

Ⅱ. 연구 배경 및 문제 정의

2.1. 두 가지 no-cache

DID Resolution의 noCache는 해석기가 캐시된 해석 결과를 쓰지 않도록 요청하는 옵션이며, HTTPS 바인딩에서는 질의 파라미터 등으로 전달된다 [2]. 해석기가 이를 지원하지 않으면 FEATURE_NOT_SUPPORTED로 응답한다. 이 옵션의 의미는 해석기 내부 상태에 미치며, 그 앞단의 임의 HTTP 캐시에는 자동으로 전파되지 않는다.

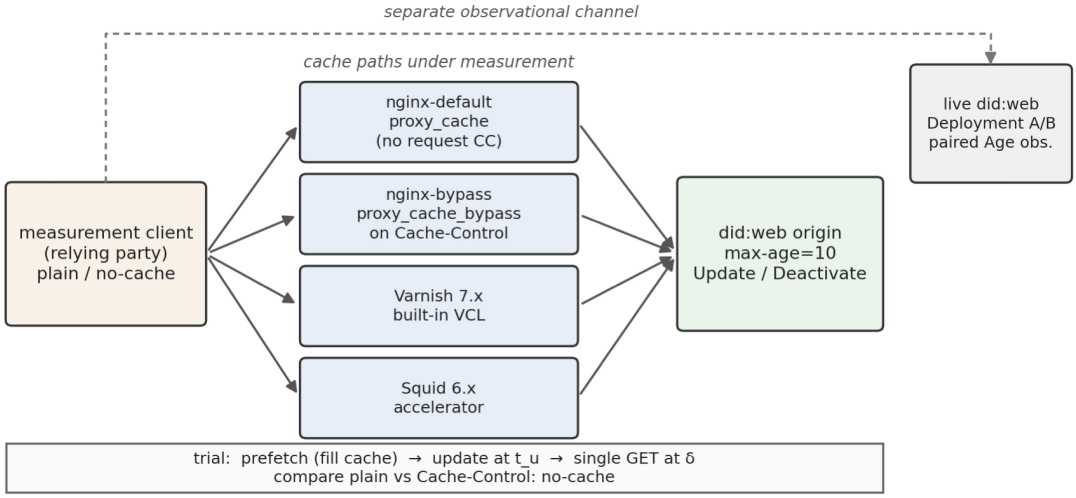
반면 HTTP Cache-Control: no-cache 요청 지시자는 저장된 응답을 원서버 검증 없이 사용하지 않기를 클라이언트가 선호함을 나타낸다 [4]. 의존 당사자가 중간 캐시에 대해 직접 행사할 수 있는 통제는 이 계층이다. RFC 9111에서 요청 캐시 지시자는 advisory 성격을 가지므로, no-cache 요청의 실제 반영 여부는 캐시 구현 및 운영 정책에 의존할 수 있다. 본 논문은 이 HTTP 지시자만을 측정하며, 해석 옵션 noCache의 end-to-end 적합성은 제외한다.

2.2. 폐기 노출

본 논문에서 폐기 노출은 키 회전에 따른 DID 문

본 연구는 한국인터넷진흥원(KISA)-정보보안 특성화대학 지원사업의 지원을 받아 수행된 연구임(34%).본 연구는 2026년도 중소벤처기업부의 기술개발사업 지원에 의한 연구임 (S3471392)(33%). 본 결과물은 농림축산식품부의 재원으로 농림식품기술기획평가원의 농식품과학기술융합형연구인력양성사업의 지원을 받아 연구되었음(RS-2024-00397026)(33%).

* 전남대학교 일반대학원 인공지능융합학과 (대학원생, geonminkim@jnu.ac.kr)
** 교신저자(corresponding author) : 전남대학교 일반대학원 인공지능융합학과 (교수, kyungbaekkim@jnu.ac.kr)



〔그림 1〕 실험 아키텍처. 측정 클라이언트가 nginx-default, nginx-bypass, Varnish, Squid 경로를 통해 did:web origin에 접근한다. 각 trial은 prefetch → t_u에서 문서 갱신 → δ에서 단일 GET이며, 일반 요청과 Cache-Control: no-cache를 비교한다.

서 갱신을 가정하고, 문서 갱신 시각 t_u 이후 갱신 전 문서가 여전히 조회되는 비율로 정의한다. 각 trial은 갱신 전에 경로별 캐시를 한 번 적재(prefetch)한 뒤 문서를 갱신하고, 경과 시간 δ에서 경로당 단일 요청을 발행한다. 일반 요청과 Cache-Control: no-cache 요청의 노출을 비교하며, δ에 따른 노출 함수 E(δ)를 수식(1) 과 같이 정의한다.

$$E(\delta) = \frac{\text{시각 } t_u + \delta \text{에서 갱신 전 문서를 반환한 } trial \text{ 수}}{\text{해당 } \delta \text{의 전체 } trial \text{ 수}} \quad (1)$$

여기서 δ는 문서 갱신 이후의 시간이며, 캐시 객체의 freshness lifetime(예: max-age)과 동일한 시계가 아님에 유의한다.

Ⅲ. 웹 캐시 환경의 실험 설계 및 측정 방법

3.1. 테스트베드 경로

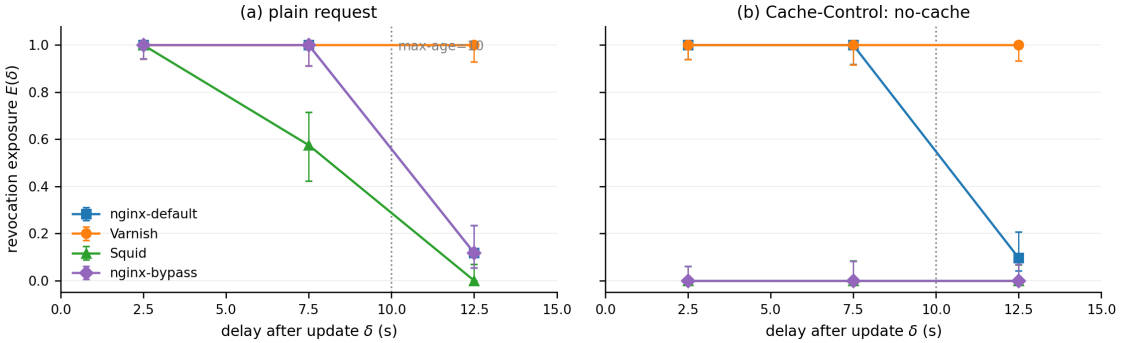
권한 origin 앞에 역프록시 캐시를 두고, 문서 갱신 시각을 제어한 뒤 경로별 응답을 수집한다. nginx는 설치 직후 상태가 아니라, proxy_cache를 켠 최소 캐시 구성을 의미한다. 그림 1에 전체 구성을, 표 1에 경로 정의를 정리한다.

〔표 1〕 측정 경로 정의

경로	구성 요지
nginx-default	nginx 1.29 + proxy_cache 기본 캐시 구성. 요청 Cache-Control을 참조하지 않음
nginx-bypass	위와 동일하되 proxy_cache_bypass / proxy_no_cache로 클라이언트 Cache-Control 우회를 명시적으로 켠 구성
Varnish	Varnish 7.x, built-in VCL (기본 grace 포함)
Squid	Squid 6.x, accelerator(역프록시) 기본 구성

3.2. 테스트베드 절차

Origin은 public, max-age=10을 광고한다. 각 trial은 미사용 식별자에 대해 prefetch(약 0.3초 후) → t_u에서 문서 갱신 → δ에서 경로당 단일 GET 순으로 진행한다. 일반 요청과 no-cache 요청은 별도 trial 집합(각 280 trial)으로 수집하였다. 표 2는 δ_actual〈10초인 trial만 합산한 k/n 비율이며, 그림 2는 δ 구간 [0,5), [5,10), [10,15)초별 Wilson 95% CI를 보인다(구간당 n≈40~63). 전체 1,680 trial은 delay 격자마다 독립 식별자를 사용한다. 본 절차의 제약은 다음과 같다. prefetch 시점의 객체 Age는 0으로 맞춰지지 않아 경로별 freshness 시계가 다를 수 있다. Squid 일반 요청의 δ∈ [5,10)초 구간 노출(23/40)만 낮고,



〔그림 2〕 갱신 후 경과 시간 δ에 따른 폐기 노출 E(δ).
(a) 일반 요청, (b) Cache-Control: no-cache 요청. 점선은 비교를 위해 origin의 max-age=10초를 표시한다.

nginx-default·nginx-bypass plain은 동일 구간에서 40/40으로 일치한다(그림 2a). 이 점을 고려할 때, Squid의 감소는 경로 전체의 동기화 실패보다는 Squid의 freshness 만료 차이로 해석한다.

3.3. 실운영 관측

공개 did:web 엔드포인트에 대해 약 60초 간격으로 plain·no-cache 요청 쌍을 보내는 능동 측정을 약 22시간(총 18,494건) 수행하였다. 측정 UA에 학술 연구 목적·연락처를 명시하였다. 분석은 응답 Age가 있어 공유 캐시가 확인된 2배포(Deployment A, B)의 1,980쌍만 사용하였으며, Age가 없는 응답은 공유 캐시 미확인으로 제외하였다. $|Age_{plain} - Age_{nc}| \le 1$이면 "Age 감소 없음"으로 정의한다.

Ⅳ. 실험 결과

4.1. 지시자 효과는 캐시 구성에 의존한다

표 2 및 그림2는 문서 갱신 후 10초 이내(δ 〈 10초)에서 관측된 폐기 노출이다. Squid는 일반 요청

〔표 2〕 문서 갱신 후 10초 이내(δ 〈 10초) 경로별 폐기 노출. 각 조건은 약 100개의 독립 trial로 구성된다.

경로	일반 요청	no-cache 요청
nginx-default	100.0% 〔96.4, 100〕	100.0% 〔96.4, 100〕
nginx-bypass	100.0% 〔96.4, 100〕	0.0% 〔0.0, 3.6〕
Varnish	100.0% 〔96.4, 100〕	100.0% 〔96.3, 100〕
Squid	**83.5% 〔75.1, 89.4〕**	0.0% 〔0.0, 3.6〕

83.5% 〔75.1, 89.4〕에서 no-cache 적용 시 0.0% 〔0.0, 3.7〕로 떨어졌다. nginx-bypass도 100% 〔96.4, 100〕에서 0.0% 〔0.0, 3.6〕로 감소하였다. 반면 nginx-default와 Varnish는 지시자 적용 후에도 각각 100% 〔96.4, 100〕, 100% 〔96.3, 100〕로, 일반 요청과 노출률 차이가 관측되지 않았다. 그림 2는 δ에 따른 E(δ)이다. no-cache 요청에서 Squid와 nginx-bypass만 전 구간 노출이 0이 되며, default와 Varnish는 일반 요청과 구분되지 않는다. Squid 일반 요청 83.5%는 제약 하 freshness 만료 차이로 해석한다(3.2절).

δ 〈 10초 구간에서 nginx-default·Varnish의 no-cache 요청은 HIT(102/102, 101/101),

〔표 3〕 실운영 배포 특성 및 no-cache 요청 효과

배포	캐시 형태	광고 Cache-Control	짝지은 요청	Age 감소 없음	Age 최대(일반/no-cache)
Deployment A	다단(CDN+Via: varnish)	max-age = 600	1,320건	97.4% (1,286건)	600/600
Deployment B	단일 managed node	max-age =0, must-revalidate	660건	100.0% (660건)	20,103/20,104

nginx-bypass는 BYPASS(102/102)로, 지시자 반영 여부가 캐시 구성에 의존함을 보여준다.

4.2. 실운영: 공유 캐시에서 Age 변화

표 3은 실운영 paired observation 결과이다. Deployment A에서는 1,320쌍 중 97.4%, Deployment B에서는 660쌍 모두에서 일반 요청 대비 no-cache 요청의 Age 감소가 관측되지 않았다. 특히 Deployment B는 max-age=0, must-revalidate를 광고함에도 일반 요청과 no-cache 요청에서 각각 최대 20,103초와 20,104초의 Age가 관측되었다. 이는 관측된 공유 캐시 경로에서 클라이언트 no-cache 요청이 응답 Age를 일관되게 감소시키지 못했음을 보여준다. 다만 어느 hop이 Age를 생성·증가시켰는지는 단정할 수 없으며, 두 배포만으로 did:web 전반에 일반화하지 않는다.

Ⅴ. 결론

본 논문은 웹 기반 DID 해석에서 의존 당사자가 중간 캐시에 직접 행사할 수 있는 HTTP Cache-Control: no-cache 요청 지시자의 실효성을 측정하였다. DID Resolution의 noCache와 HTTP no-cache는 서로 다른 계층의 통제이며, 후자를 클라이언트가 보내더라도 실제 웹 캐시 경로 전체의 신선도는 보장되지 않는다. 테스트베드에서는 Squid와 proxy_cache_bypass를 켠 nginx 구성만이 지시자를 반영하였고, nginx 기본 캐시 구성과 Varnish는 반영하지 않았다. 실운영 공유 캐시 2곳에서도 지시자로 인한 응답 Age 감소가 거의 관측되지 않았다. 따라서 키 폐기의 신선도를 클라이언트 요청 지시자에만 의존해서는 안 된다. did:web 운영자는 origin 신선도 정책과, 가능한 경우 proxy_cache_bypass·캐시 무효화 연동을 검토해야 하며, 의존 당사자는 해석 경로의 캐시 구성을 알 수 없다면 no-cache만으로 신선도를 가정해서는 안 된다.

참 고 문 헌

〔1〕 W3C, “Decentralized Identifiers (DIDs) v1.0,” W3C Recommendation, Jul. 2022.

[2] W3C, “Decentralized Identifier Resolution (DID Resolution) v1.0,” W3C Candidate Recommendation Snapshot, Aug. 2026.

〔3〕 W3C Credentials Community Group, “did:web Method Specification,” Draft Community Group Report, 2024.

〔4〕 R. Fielding, M. Nottingham, and J. Reschke, “HTTP Caching,” IETF RFC 9111, Jun. 2022.

〔5〕 C. Mazzocca, A. Acar, A. S. Uluagac, R. Montanari, P. Bellavista, and M. Conti, “A Survey on Decentralized Identifiers and Verifiable Credentials,” IEEE Communications Surveys & Tutorials, vol. 27, no. 6, pp. 3641-3671, 2025.

〔6〕 Y. Liu, W. Tome, L. Zhang, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, A. Schulman, and C. Wilson, “An End-to-End Measurement of Certificate Revocation in the Web’s PKI,” in Proc. ACM Internet Measurement Conference (IMC), 2015, pp. 183-196.

〔7〕 S. Huh, M. Shim, J. Lee, S. S. Woo, H. Kim, and H. Lee, “DID We Miss Anything? Towards Privacy-Preserving Decentralized ID Architecture,” IEEE Transactions on Dependable and Secure Computing, vol. 20, no. 6, pp. 4881-4898, 2023.