정 보 보 호 학 회 지
제 33 권 제5호, 2023. 10

안전하고 효율적인 키-밸류 스토어 기술 동향

오 현 영*

요 약

디지털 시대에 클라우드 서비스가 급성장함에 따라 데이터 보호 및 보안에 대한 지속적인 관심이 요구되고 있다. 수많은 클라우드 서비스들의 필수 구성 요소인 인메모리 키-밸류 스토어 (Key-Value Store)는 데이터 컨텐츠 캐싱부터 공유 상태 유지에 이르기까지 다양한 작업을 처리한다. 인메모리 KVS가 클라우드 서비스에서 차지하는 비중을 고려할 때 이러한 인메모리 KVS는 특히 신뢰할 수 없는 클라우드 서비스 제공자 혹은 외부 공격자로부터의 위협 시나리오에서 데이터 침해의 주요 대상이 된다. 본 논문에서는 인메모리 KVS와 관련된 설계, 구현 및 보안 위협 대응의 발전 동향과 최근 기술들을 소개한다.

Ⅰ. 서 론

클라우드 기반 서비스에는 여러 가지 구성 요소가 결합되어 사용자가 원하는 민첩성, 확장성 및 유연성을 제공한다. 이러한 서비스의 중심에는 광범위한 기능을 지원하는 인메모리 키-밸류 스토어 (In-memory Key-Value Store, 이하 KVS로 통칭)가 있는데, KVS를 통해 자주 액세스하는 컨텐츠 캐싱에서부터 분산 서비스 간 공유 상태 관리에 이르기까지 KVS의 기능은 다양하고 클라우드 구조에 통합되어 있다[1-3]. 이러한 KVS의 역할을 고려할 때, KVS에는 종종 민감한 데이터가 다량으로 저장되는 것은 놀랍지 않으며, 공격자의 주요 공격 대상 중 하나가 되고 있다. 외부의 해커든 아니면 악의적인 의도를 가진 내부자 (insider)든 KVS에 위협적인 환경은 방대하며 진화하고 있다.

이러한 보안적인 위협에 대응하기 위해 KVS가 제공하는 기능을 안전하게 실행하기 위한 여러 연구가 진행되고 있다. 기본적인 데이터 암호화를 적용한 기술에서부터 신뢰실행환경 (TEE)를 활용한 연구까지 다양한 방법으로 KVS의 보안을 강화하는 기술들이 제안되었다. 하지만, 이렇게 보안을 강화하려는 노력으로 인해 KVS의 특징인 효율성과 성능이 뒷전으로 밀려날 수 없다. 보안적으로는 매우 안전하지만, 성능이 매우 부진한 시스템은 클라우드 서비스 사용자들의 서비스 품질 (QoS)를 심각하게 저하시킬 수 있으므로 보안과 성능 사이에서 신중한 균형을 유지해야 한다. 이러한 배경하에서 본 논문에서는 KVS가 당면한 이중적인 과제를 효과적으로 해결하고자하는 연구들에 집중하여 최근 동향을 소개하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 KVS에 대한 배경지식을 소개하며, 3장에서는 소프트웨어 기반의 KVS 연구, 4장에서는 SGX 기반의 KVS 연구 그리고 5장에서는 SGX 및 하드웨어의 하이브리드 방식의 KVS 연구 현황을 살펴보고, 6장에서 본 논문의 결론을 맺는다.

Ⅱ. 인메모리 키-밸류 스토어

2.1. 키-밸류 스토어

KVS의 본질은 사용자가 데이터를 키-밸류 쌍의 집합으로 저장할 수 있다는 간단한 원칙에 따라 작동한다. 여기서 '키'는 고유한 식별자의 역할을 하며, 관련된 '밸류'에 신속하게 액세스할 수 있도록 보장한다. 이러한 기본 데이터 구조는 '밸류'가 단순한 숫자 데이터에서 복잡한 객체 또는 블록에 이르기까지 무엇이든 될 수 있기 때문에 뛰어난 유연성을 제공한다. KVS의 장점은 단순성에 있으므로, 다양한 응용 분야

이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원(No. RS-2022-00166529), 2022년도 가천대학교 교내연구비 지원(GCU-202208860001)을 받아 수행된 연구임 .
* 가천대학교 AI소프트웨어학부 (조교수, hyoh@gachon.ac.kr)

에서 다용도로 사용되고 있다. KVS는 읽기 및 쓰기 작업이 매우 빠르다는 특징을 보인다.

2.2. 인메모리 키-밸류 스토어

KVS는 디스크 기반 스토리지 시스템에서 작동할 수 있지만, 인메모리 KVS는 주로 시스템의 메인 메모리 (RAM)에 데이터를 저장한다. 이러한 차이점은 다음과 같은 중요한 의미를 갖는다. 속도 측면에서, 디스크 스토리지보다 RAM에서 데이터에 액세스하는 속도가 훨씬 빠르기 때문에 인메모리 KVS는 최소 마이크로초 단위의 지연 시간을 제공할 수 있으므로 실시간 데이터 액세스가 필요한 애플리케이션에 매우 유용하다. 하지만, RAM의 데이터는 휘발성의 특징을 지니며, 이는 시스템 크래시가 발생하거나 재시작될 경우 데이터 손실이 일어남을 의미한다. 이를 방지하기 위해 인메모리 KVS는 데이터의 백업 전략에 대한 메커니즘을 제공해야한다. 또한 비용측면에서도 RAM은 일반적으로 디스크 스토리지보다 더 비싸기 때문에 인메모리 KVS는 뛰어난 속도를 제공하지만, 자주 액세스하지 않는 방대한 데이터를 저장하는데에는 최적의 선택이 아니다.

2.3. 인메모리 키-밸류 스토어의 기본 동작

일반적으로 인메모리 KVS가 지원하는 핵심 동작은 다음과 같이 비교적 간단하다. PUT (또는 SET) 동작은 저장소에 새 키-밸류 쌍을 추가하는 작업이며, 키가 이미 존재하는 경우 해당 값이 새 데이터로 업데이트 된다. GET 동작은 키를 사용하여 키와 관련된 밸류를 검색하며, 키가 존재하지 않으면 “not found” 응답을 반환한다. DEL 동작은 제공된 키를 기준으로 저장소에서 키-밸류 쌍을 제거하는 작업이다. 일부 고급 KVS에서는 사용자가 특정 기준 또는 키 범위에 따라 키-밸류 쌍의 범위를 검색할 수 있도록 SCAN 동작을 지원하기도 한다.

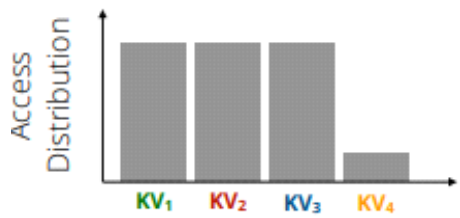
Ⅲ. 소프트웨어 기반의 KVS 연구

KVS의 보안을 보장하기 위해 소프트웨어 기반 기법들은 주로 데이터의 기밀성을 보장하기 위한 암호화 프로토콜에 의존하고 있다. 이러한 기법들은 데이터가 저장되기 전에 암호화되어 검색시 암호를 해독함으로써 무단 접근이 발생하더라도 침입자가 실제 내용을 해독할 수 없도록 한다. Amazon의 ElastiCache[4]는 클라우드에서 확장성을 지닌 인메모리 KVS 및 캐시 서비스를 제공하는데, 저장중이거나 노드 간 또는 애플리케이션과 캐시 간에 통신할 때 모두 데이터 보호를 위해 유휴 상태 (rest)와 전송 상태 (transit) 암호화 기능을 제공한다. 하지만, 이러한 암호화 및 복호화는 계산 집약적인 작업으로, 성능 오버헤드를 유발하여, 암/복호화를 하드웨어 가속기 없이 전적으로 소프트웨어로 처리하는 경우, 특히 요청 빈도가 높은 작업이나 대량 데이터 처리 중에 지연 시간이 크게 증가할 수 있다. 또한 암호키를 저장하는 스토리지가 서비스 관리자가 접근할 수 있는 영역에 존재하는 경우 내부자의 위협으로부터 보안성을 보장할 수 없다.

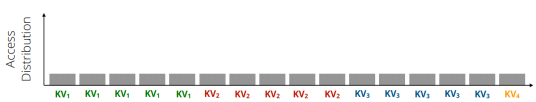
또 다른 기법으로는 Pancake[5]가 있는데, 이는 암호화 적용을 통해 데이터 기밀성을 지킬 뿐만 아니라 키-밸류 접근 패턴에 대한 부채널 공격에 대한 내성을 보장하는 기술이다. 그림 1과 같이 데이터에 대한 접근이 특정 킷값에 편중되는 경우 데이터 내용을 모르더라도 이러한 접근 패턴을 기반으로 하는 부채널 공격에 노출이 되는데, Pancake는 이를 동일 데이터의 복제 (replication)와 가접근 (fake accesses)을 적절히 조합하여 효율적으로 접근 패턴의 편향성을 제거하였다.

즉, 복제 (replication) 방식만을 사용하는 경우에는 그림 2와 같이 지나치게 많은 저장소를 추가적으로 요구하고, 가접근 (fake accesses) 방식만을 사용하는 경우에는 그림 3과 같이 대역폭이 크게 증가한다.

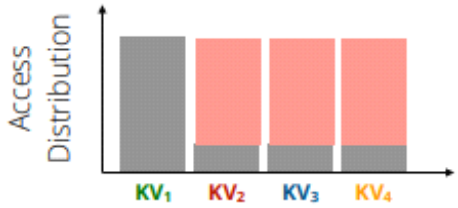
하지만, Pancake는 그림 4와 같이 1차적으로 데이터 복제를 통해 부분적인 편향성 제거를 한 이후에 가



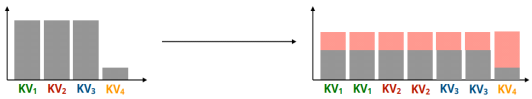
〔그림 1〕 키-밸류 접근 패턴에 존재하는 편향성



〔그림 2〕 복제 방식만을 사용한 경우의 접근 패턴



〔그림 3〕 가접근 추가만을 적용한 경우의 접근 패턴



〔그림 4〕 복제와 가접근 조합으로 접근 패턴의 평탄화

접근을 추가하는 방식으로 저장소의 추가 요구량과 대역폭 증가량을 최소화하였다.

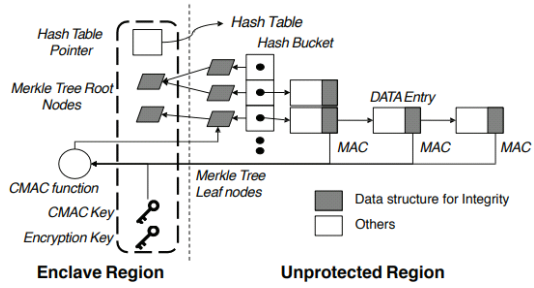
결과적으로, 기존 패턴 기반의 부채널 방어 기법인 PathORAM[6]에 비해 지연시간은 12배 감소시키고, 처리량 (Throughput)은 220배가 증가하였지만, 여전히 소프트웨어 기반의 방식이다 보니, 보안 기법이 추가되지 않은 일반적인 KVS에 비해서는 처리량이 7.6배가 하락하였다.

Ⅳ. SGX 기반의 KVS 연구

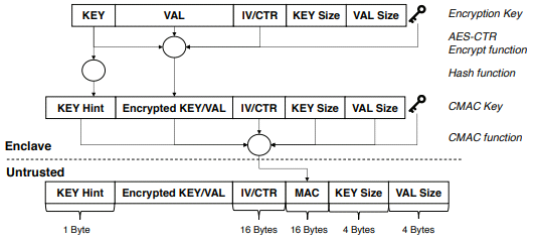
Ⅲ장에서 살펴봤던 소프트웨어 기반 KVS의 문제점이었던, 암호키에 대한 내부자의 접근이 가능하다는 점과 암호화/복호화 연산이 소프트웨어로 동작하기 때문에 발생하는 성능 오버헤드를 극복하고자 하드웨어 TEE (Trusted Execution Environment)를 활용한 KVS 기술이 존재한다. ShieldStore[7]는 Intel SGX[8]를 활용하여, KVS의 데이터들에 대한 기밀성 및 무결성을 보장하는 기술이다. SGX는 관리자의 권한을 가지고 있는 OS조차 부터로도 접근이 차단되어 격리된 실행 환경인 엔클레이브 (enclave)를 사용자에게 제공한다. 따라서 SGX는 제 3의 클라우드 서버에 자신의 데이터를 저장하게 되는 사용자에게 내부자 위협으로부터 안전한 데이터 저장소를 제공하는데 적합한 기술이다. 또한, 메모리에 저장되는 데이터에 대해서 CPU 칩 내부에 탑재된 전용 암호화 하드웨어 엔진 (memory encryption engine)을 사용하여 암호화/복호화 연산을 고속으로 수행하기 때문에 성능적으로도 뛰어나다고 할 수 있다. 하지만, SGX가 제공하는 안전한 메모리 공간인 EPC (Enclave Page Cache)는 128MB로 한정되어, 그 이상의 데이터를 저장하고자 할 때에는 non-EPC 영역으로의 데이터 축출 (eviction)이 요구되는데, 이때에는 데이터 페이지 단위 스와핑에 추가적인 암호화 연산이 필요하여 급격하게 성능 하락이 발생하는 한계가 있다. 이러한 한계는 다량의 데이터를 저장하는 KVS 응용에 있어서 치명적인 단점으로 작용한다.

ShieldStore는 이를 극복하기 위해, 키-밸류 데이터 쌍을 non-EPC 영역에 커스텀 암호 알고리즘을 적용하여 페이지 스와핑으로 인한 성능 하락을 회피하였다. 그림 5는 ShieldStore의 전체 키-밸류 데이터를 저장하는 구조를 나타낸다.

즉, 킷값 검색을 효율적으로 하기 위해 해시 테이블 구조를 적용하였고, 각 키-밸류 쌍의 데이터에 대한 MAC (message authentication code) 값들이 결합된 해시값을 통해 데이터 무결성을 효율적으로 검증하기 위해 각 해시값들을 트리 구조로 엮었다. 트리의 루트 값은 안전한 저장소인 SGX의 EPC 영역에 저장하여 데이터와 해시값에 대한 리플레이 공격을 방어할 수 있게 설계하였다. 또한 데이터 암호화 및 MAC 생성에 사용되는 암호키들을 안전한 영역인 EPC에 저장하여 데이터에 대한 기밀성이 지켜지도록 하였다. 각 키-밸류 쌍에 대한 ShieldStore의 커스텀 암호화 체인은 그림 6과 같다.



〔그림 5〕 ShieldStore의 키-밸류 데이터 저장 구조



〔그림 6〕 ShieldStore의 키-밸류 쌍 암호화 처리

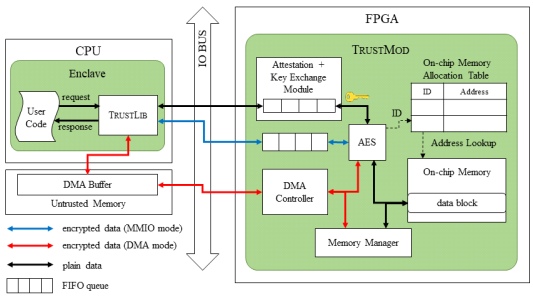
또한, ShieldStore에서는 추가 성능 최적화를 위하여, 킷값 검색을 함에 있어서 킷값의 일부 (KEY Hint)를 암호화하지 않고 노출시켜 킷값에 대한 완전한 복호화과정 없이 대략적인 키-밸류 쌍의 위치를 찾을 수 있도록 하였다. 결과적으로, SGX의 EPC를 키-밸류 쌍의 저장소로 사용하여 다량의 페이징이 일어나는 구현에 비해 다양한 워크로드에서 최대 30배의 처리량 향상 효과를 가져왔다. 하지만, 보안 기법이 추가되지 않은 일반적인 KVS에 비해서는 최대 4.5배 낮은 처리량을 보였는데, 이는 여전히 커스텀 암호화에 대한 처리를 소프트웨어가 처리하기 때문이다.

Ⅴ. SGX-하드웨어 하이브리드 기반의 KVS 연구

앞서 살펴본 KVS에 대한 연구는 모두 어느 정도의 보안성을 제공하고는 있지만, 특정 암호 연산에 대한 처리를 소프트웨어가 담당함에 따라 무시 못할 성능 오버헤드가 발생하였다. 이를 해결하기 위해 하드웨어를 적극적으로 활용하는 연구들이 있는데, 본 논문에서는 FPGA (Field Programmable Gate Array)를 활용한 연구들을 소개하고자 한다.

FPGA는 Amazon AWS 및 Microsoft Azure 같은 클라우드 상용 시스템에 탑재되어 서비스 되고 있는데, 그 이유는 FPGA의 프로그래밍 가능성, 병렬성 및 저전력의 장점 때문이다. 이러한 장점을 활용하여 데이터 분석, 영상 처리, AI, 네트워크 보안, 금융, 유전자 분석, 동형암호에 이르기까지 여러 응용 연산의 가속기로 활용이 되고 있다. 이러한 범용적인 가속기로서의 용도 뿐만 아니라, 최근 연구[9,10]에 의하면, FPGA를 TEE로 동작하게 하는 기술이 제안되어 보안 용도로까지 그 영역이 확대되었다.

TrustOre[9]는 SGX를 기반으로 FPGA로의 신뢰를 확장하여 데이터를 부채널로부터 안전한 FPGA에 저장하는 데이터 스토리지 시스템을 제안하였다. 그림 7은 TrustOre의 전체 구조를 나타내고 있다.



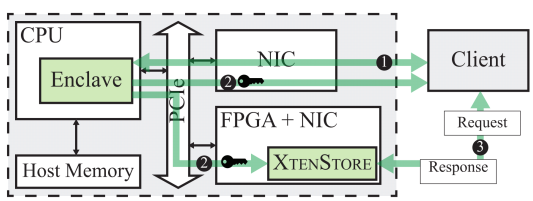
〔그림 7〕 TrustOre의 전체 구조도

FPGA 제조 과정 중에 암호키가 외부에 노출되지 않도록 FPGA 내부 ROM에 저장시켜, FPGA 안에서 데이터의 저장 및 읽기 동작을 수행하는 TrustMod 하드웨어 모듈이 부팅 과정 중에 자동으로 내부에 저장된 암호키를 사용한 인증 및 복호화를 통해 FPGA로 시큐어 로딩되도록 하였다. 사용자의 SGX 엔클레이브는 데이터 저장 및 읽기 요청을 암호화된 패킷 형태로 ID/데이터를 포함하여 FPGA로 전송한다. 여기에서 데이터 위치와 관계된 ID 정보가 암호화되기 때문에 부채널 공격으로 부터의 내성을 획득하게 된다. 전송된 요청에서 ID 값을 바탕으로 실제 FPGA 내부 메모리의 주소값으로의 매핑을 통해 데이터 접근이 이루어진다.

여기서 FPGA 내부에 저장된 데이터 접근을 처리하는 방식이 특정 키와 데이터를 쌍으로 처리하는 KVS와 닮아있다. 하지만, TrustOre는 KVS로의 특화가 아닌 범용적인 저장소로의 기능을 제공하기 때문에 KVS 시스템에 적용하였을 때의 성능 최적화가 미비하다.

FPGA를 좀 더 적극적으로 KVS에 활용한 연구로는 XtenStore[11]가 있다. XtenStore는 FPGA가 탑재된 NIC (Network Interface Card)를 사용하여, 동일한 PC내의 다른 프로세스가 KVS 요청을 발생시키고 또 다른 프로세스가 처리하는 standalone 사용 모드 뿐만 아니라 네트워크로 전송되는 KVS 요청을 처리하는 네트워크 모드에 대해서도 성능 향상을 달성하고자 하였다. 그림 8은 XtenStore의 전체 구조를 나타내고 있다.

우선 다른 네트워크 노드 상에 존재하는 사용자는 Intel SGX가 제공하는 원격 검증 서비스를 사용하여 클라우드 내 엔클레이브와 안전한 통신 채널을 수립한다. 그 이후 이 채널을 사용하여 FPGA와 사용자가 세션키를 공유한다. 공유된 세션키를 사용하여 사용

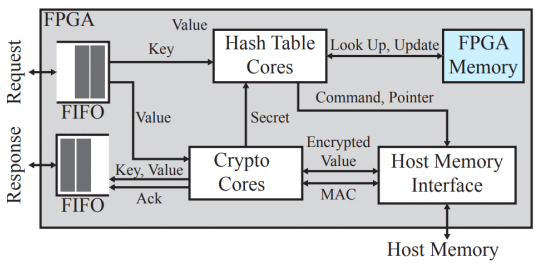


〔그림 8〕 XtenStore의 전체 구조도

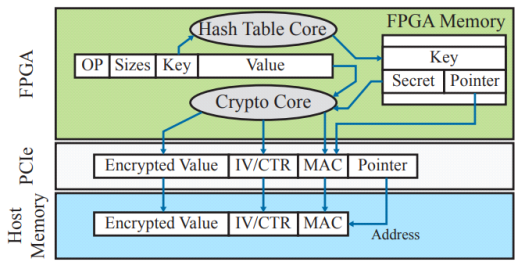
자는 FPGA가 장착된 NIC와 직접 통신을 하며 KVS 서비스를 제공 받는다. XtenStore는 FPGA에 암호연산을 가속하기 위하여 병렬처리를 위한 다수의 하드웨어 엔진들을 탑재하고 있다. 그림 9는 FPGA 내부에 구현되어 있는 하드웨어 구조를 나타낸다.

FPGA 하드웨어 자원을 최대한 활용하여, 킷값 검색을 위한 해시값 계산 엔진인 Hash Table Core를 2개를 두어 PUT 요청과 GET 요청을 동시에 처리할 수 있도록 하였고, Hash Table Core는 파이프라인 설계를 적용하여 연속적인 요청에 대해서 최적의 처리량을 보장할 수 있도록 하였다. 또한 FPGA 메모리의 용량이 한계가 있기 때문에 다량의 KVS 데이터 저장을 위해서는 호스트 메모리를 활용할 수 밖에 없는데 이때 데이터의 기밀성과 무결성 보장을 위해 필요한 암호 연산을 고속으로 처리하기 위한 Crypto Core가 60개 탑재되어 있다. 호스트 메모리로의 키-밸류 데이터 저장을 안전하게 하고, 저장 공간을 효율적으로 사용하기 위해 그림 10과 같이 FPGA 메모리와 호스트 메모리를 two-tier 형태로 구성하여 키 값은 FPGA 내부에 저장하고, 밸류 데이터는 암호화하고 MAC 코드를 첨가하여 호스트 메모리에 저장하도록 하였다.

호스트 메모리에 저장된 밸류 데이터에 대한 리플레이 공격 방어를 위해 밸류 데이터 매 암호화 과정에 결합되는 Secret 값을 각각의 킷값에 대해 매번 랜덤하게 생성하여 FPGA 메모리에 저장하도록 하였다. 공격자가 리플레이 공격을 통해 유효한 패킷으로 조작을 하더라도, 공격자의 접근이 어려운 FPGA 메모리에 저장되있는 Secret 값을 통해 패킷의 조작 유무를 검증할 수 있게 된다.



〔그림 9〕 고속 KVS를 위한 XtenStore 하드웨어 구조



〔그림 10〕 XtenStore의 two-tier 메모리 구조

XtenStore의 성능은 다양한 워크로드에 대해서 SGX 기반의 KVS인 ShieldStore보다 최대 33배 높은 처리량을 보였다. 이는 보안 기법이 추가되지 않은 일반적인 소프트웨어 기반의 KVS 기술에 비해서도 7배 이상 높은 처리량이다. 최근 FPGA의 발전 속도를 보면, HBM2 등 고속의 대용량 FPGA 메모리가 탑재되고, 많은 양의 하드웨어가 집적된 고성능 FPGA 칩[12]이 릴리즈되고 있는데, XtenStore는 비교적 로우-엔드 급의 FPGA를 활용한 KVS 기술이라 좀 더 고성능의 FPGA를 활용한다면, 성능 향상이 더 큰 폭으로 이루어질 것으로 생각된다.

Ⅵ. 결 론

본 논문에서는 클라우드 서비스에 널리 사용되는 인메모리 키-밸류 스토어의 보안성 및 성능 향상을 위한 최신 연구 동향에 대해서 살펴보았다. 소프트웨어 기반의 KVS 시스템은 데이터의 기밀성 및 무결성 보장을 위해서 암호화 알고리즘을 적용시키는데 이러한 연산들이 모두 소프트웨어 기반으로 이루어지다 보니, 성능 측면에서 가장 낮은 수준을 제공하였다. Intel SGX와 같은 하드웨어 기반의 TEE를 활용한 KVS 시스템은 소프트웨어 기반의 KVS 대비 조금 더 효율적으로 보안성을 지키나 여전히 일부 암호 연산들이 소프트웨어로 실행되는 한계가 있었다.

SGX와 동시에 FPGA를 하이브리드로 활용하여 성능까지 향상시킨 연구에서는 보안성을 기존 KVS 기술들과 동일하게 유지시키는 동시에 성능 측면에서는 괄목할만한 성과를 달성하였다. 하지만, 좀 더 고성능의 FPGA를 활용한다든지, 아니면 여러 개의 FPGA를 클러스터로 엮어서 KVS 서비스를 제공하는 등 동일한 보안성을 유지하면서도, 성능 측면에서 좀 더 향상시킬 수 있는 가능성이 열려있다고 사료된다.

참 고 문 헌

[1] B. Fitzpatrick, “Distributed caching with memcached,” Linux J., vol. 2004, no. 124, pp. 5–, Aug. 2004.

[2] R. Labs, “Redis,” https://redis.io/, accessed: 2023-09-27.

[3] “Ignite,” https://ignite.apache.org/, accessed: 2023-09-27.

[4] Amazon, "ElastiCache for Redis," https://docs.aws.amazon.com/AmazonElastiCache/latest/red-ug/, accessed: 2023-09-27.

[5] Paul Grubbs, Anurag Khandelwal, Marie-Sarah Lacharité, Lloyd Brown, Lucy Li, Rachit Agarwal, Thomas Ristenpart, "Pancake: Frequency Smoothing for Encrypted Data Stores," USENIX Security Symposium 2020, pp. 2451-2468, 2020.

[6] Emil Stefanov, Marten Van Dijk, Elaine Shi, T.-H. Hubert Chan, Christopher Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas, "Path ORAM: An Extremely Simple Oblivious RAM Protocol," J. ACM 65, 4, Article 18, Aug 2018.

[7] Taehoon Kim, Joongun Park, Jaewook Woo, Seungheun Jeon, and Jaehyuk Huh, “ShieldStore: Shielded In-memory Key-value Storage with SGX,” In Proceedings of the Fourteenth EuroSys Conference 2019 (EuroSys '19). Association for Computing Machinery, 2019.

[8] Victor Costan, Srinivas Devadas, "Intel SGX Explained," IACR Cryptol. ePrint Arch. 2016.

[9] Hyunyoung Oh, Adil Ahmad, Seonghyun Park, Byoungyoung Lee, and Yunheung Paek, "TRUSTORE: Side-Channel Resistant Storage for SGX using Intel Hybrid CPU-FPGA," In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS '20). Association for Computing Machinery, pp. 1903–1918, 2020.

[10] Hyunyoung Oh, Kevin Nam, Sungil Jeon, Youngpil Cho and Yunheung Paek, "MeetGo: A Trusted Execution Environment for Remote Applications on FPGA," in IEEE Access, vol. 9, pp. 51313-51324, 2021.

[11] Hyunyoung Oh, Dongil Hwang, Maja Malenko, Myunghyun Cho, Hyungon Moon, Marcel Baunach and Yunheung Paek, "XTENSTORE: Fast Shielded In-memory Key-Value Store on a Hybrid x86-FPGA System," 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 560-563, 2022.

[12] Xilinx, "Alveo U280 Data Center Accelerator Card," https://www.xilinx.com/products/boards-and-kits/alveo/u280.html#specifications, accessed: 2023-09-27

〈저 자 소 개〉

오 현 영 (Hyunyoung Oh)

종신회원

2005년 7월 : 연세대학교 전기전자공학과 졸업

2007년 2월 : 연세대학교 전기전자공학과 석사

2007년 2월~2017년 2월 : 삼성전자 책임연구원

2022년 2월 : 서울대학교 전기정보공학부 박사

2022년 3월~현재 : 가천대학교 AI·소프트웨어학부 조교수

<관심분야> 하드웨어기반 보안, AI 보안, 프라이버시