Enhancing NAC-ABE to Support Access Control for mHealth Applications and Beyond

Saurab Dulal*, Suravi Regmi†, Tianyuan Yu‡, Adam Thieme‡, Lixia Zhang‡, Lan Wang†
*Meta †University of Memphis ‡UCLA
dulal.saurab@gmail.com {sregmi1, lanwang}@memphis.edu {tianyuan, adam, lixia}@cs.ucla.edu

***Abstract*—Name-based Access Control (NAC) in Named Data Networking (NDN) enforces fine-grained access control by encrypting data at production time with keys across multiple granularities following structured name hierarchies. NAC-ABE extends NAC with Attribute-Based Encryption (ABE) to improve scalability. However, the current NAC-ABE library relies on Ciphertext-Policy ABE (CP-ABE), which requires access policies to be fixed at encryption time. This is problematic for mobile health (mHealth) applications where access policies evolve dynamically. NAC-ABE also generates a new content key for each data packet, incurring significant overhead for high-frequency data streams, and lacks trust-schema-based data validation, leaving it vulnerable to unauthorized data injection. We present enhancements to the NAC-ABE library in the context of mGuard, a secure real-time mHealth data sharing system. Evaluation on Mini-NDN shows that our approach reduces content key overhead by several orders of magnitude, enabling practical real-time encrypted data sharing for mHealth and beyond.**

***Index Terms*—Named Data Networking (NDN), mHealth, Access Control, Real-time Data Sharing**

I. Introduction

Rising health awareness has driven widespread adoption of wearable devices for tracking activity and physiological signals, fueling market growth [1]. Mobile health (mHealth) applications increasingly rely on real-time data from these devices to enable remote monitoring, timely interventions, and personalized feedback [2]. Securing such data requires fine-grained access control over attributes such as data stream, time, location, and activity.

Consider the scenario in Figure 1: Alice's phone and wearables continuously generate streams of data such as blood glucose and heart rate. The data streams must be shared under distinct constraints: her doctor accesses blood glucose data generated at home, her trainer accesses heart rate data recorded at the gym after April 7, 2025, and a researcher accesses both streams generated at work. Each party should only access data matching its specific attribute combination.

Named Data Networking (NDN) [3] is a data-centric architecture where data is retrieved by name rather than IP address. mGuard [4] is a secure real-time mHealth data sharing application built on NDN and uses NAC-ABE [5] for fine-grained access control. However, the original NAC-ABE [6] supports only Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [7], which requires access policies to be fixed at encryption time. In mHealth settings, data attributes are known at production, but policies evolve—new users may require access after data publication, and re-encrypting previously published data is impractical at scale. For example, suppose we apply CP-ABE to the scenario in Figure 1, then each access control policy needs to be embedded in the encrypted data at production time so if a new party, say a nutritionist, later needs access to Alice's data, all affected content must be re-encrypted.

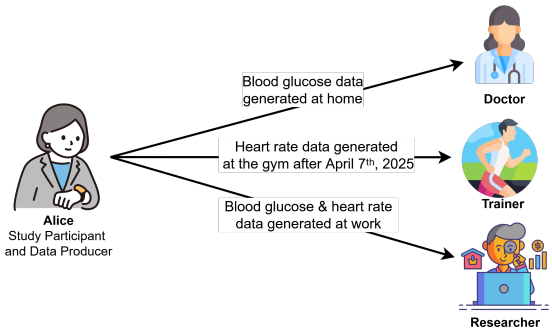


Fig. 1. Motivating scenario: Alice's mHealth data is shared with her doctor, gym trainer, and a researcher, each under different access constraints.

To address this limitation and other identified issues, we extend NAC-ABE (Section III) to make the following contributions. First, we incorporate Key-Policy Attribute-Based Encryption (KP-ABE) [8], enabling flexible policy assignment without re-encryption. Second, we revise the data/key naming scheme to support KP-ABE and key segmentation. Third, we design a producer that reuses content keys and supports configurable key update intervals. Fourth, we incorporate trust-schema-based data validation to ensure the authenticity of data exchanged by NAC-ABE. Finally, we release our enhanced NAC-ABE library [9] as open source. Evaluation on Mini-NDN [10] shows that our approach reduces content key computation overhead (from over 4 hours to 0.04 seconds per hour of 100 Hz data), enabling practical real-time encrypted data sharing for mHealth and other data-intensive applications (Section IV). We also discuss open issues, including access revocation and attribute encoding overhead, with directions for future work (Section V).

II. Background and Related Work

This section briefly reviews NAC, NAC-ABE, mGuard, and related work.

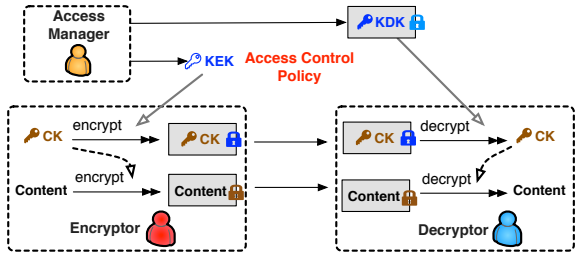


Fig. 2. Content encryption and decryption in NAC

A. ICN Security and Name-based Access Control (NAC)

Named Data Networking (NDN) is a leading architecture in Information-Centric Networking (ICN). Tourani et al. [11] survey ICN security and identify access control as an open challenge. Nour et al. [12] present a comprehensive survey of access control mechanisms in NDN, classifying existing approaches and identifying key challenges in scalability and dynamic policy management.

NAC [13] enables data confidentiality and access control in NDN by combining symmetric and asymmetric cryptography with structured naming. As shown in Figure 2, a producer encrypts content with a symmetric Content Key (CK), which is then encrypted using a Key-Encryption Key (KEK) published by an access manager M. M also generates a Key-Decryption Key (KDK)—the private counterpart to the KEK—and distributes it by encrypting it with each authorized consumer C's public key. A consumer retrieves its KDK by name, decrypts it using its private key, derives the CK, and finally decrypts the content.

While NAC automates key distribution through naming, it does not scale well: for m consumers and n access granularities, it may require up to $m \times n$ KDK packets.

B. ABE and NAC-ABE

Attribute-Based Encryption (ABE) [14] uses descriptive attributes and logical policies (AND, OR, $<$, $>$, $=$) to control decryption. The two main variants are Ciphertext-Policy ABE (CP-ABE) [7] and Key-Policy ABE (KP-ABE) [8]. Prior work has applied CP-ABE to resource-constrained settings such as mobile health [15], and explored decryption outsourcing to reduce client overhead [16]. In CP-ABE, data is encrypted under an access policy and each user's decryption key (DKEY) encodes their attributes. For example, a policy *("trainer" AND "gym")* allows a trainer with matching attributes to decrypt mHealth data. Therefore, policies must be fixed at encryption time. Li et al. [17] propose ICN access control using ABE but do not address efficient key management for high-frequency streams.

NAC-ABE [5] replaces NAC's public-key-based key distribution with ABE to improve scalability. The original implementation [6] (denoted NAC-ABE-2020) adopts CP-ABE. In NAC-ABE-2020, the Attribute Authority (AA) generates public parameters and a master key. For each consumer C, the AA issues a DKEY based on C's attributes, encrypts it with a symmetric key, which is in turn encrypted with C's public key before distribution. A producer encrypts data with a symmetric CK and encrypts the CK using the policy and public parameters. A consumer retrieves the DKEY, decrypts the CK, and then decrypts the data. This design reduces key distribution overhead from $m \times n$ KDKs in NAC to m DKEYs.

C. mGuard

mGuard [4] is a secure real-time mHealth data sharing system built on NDN and NAC-ABE. It consists of four components: a **Producer** that generates encrypted data streams, a **Repo** that stores data and keys, a **Consumer** that retrieves and decrypts data, and a **Controller** that manages access control via a Policy Parser, an Access Manager, and an Attribute Authority. In the mHealth scenario illustrated in Figure 1) a user's devices act as the Producers generating continuous data streams (e.g., heart rate and glucose), while authorized parties such as doctors and trainers act as Consumers with differentiated access rights.

mGuard supports 1–120 Hz data streams and targets four requirements: (1) fine-grained contextual access control based on stream, location, activity, and time; (2) efficiency for high-rate data; (3) dynamic policy updates without re-encryption; and (4) resilience under intermittent connectivity.

III. Design and Implementation Enhancements

In this section, we describe the design and implementation changes we made to NAC-ABE-2020 [6] to address its limitations. Figure 3 shows the updated interface and message exchanges, with our modifications highlighted in bold.

A. KP-ABE Addition

NAC-ABE-2020 uses CP-ABE which requires the encryptor to know the access control policy when encrypting data. This poses two drawbacks. First, policy changes require re-encryption. If a nutritionist is later added to access Alice's blood glucose data (Figure 1), the access control policy must be updated and all affected CKs re-encrypted and re-published.

Second, using data-centric attributes with CP-ABE can lead to incorrect access. Consider Alice's two data streams: blood glucose ("`<stream prefix>/blood-glucose`") and heart rate ("`<stream prefix>/heart-rate`"). Suppose we encrypt blood glucose data at home with the CP-ABE policy ("`<stream prefix>/blood-glucose`" AND "`home`"). A researcher authorized for blood glucose at work and heart rate at home would have four attributes in their DKEY: "`<stream prefix>/blood-glucose`", "`work`", "`<stream prefix>/heart-rate`", and "`home`". Since the researcher possesses both "`<stream prefix>/blood-glucose`" and "`home`", they can decrypt blood glucose data at home, even though their authorization only covers blood glucose at work.

KP-ABE [8] addresses both issues. First, in KP-ABE, producers encrypt data using the data's attributes, while each consumer's DKEY encodes an access control policy. In the same scenario, adding a nutritionist requires only issuing a new

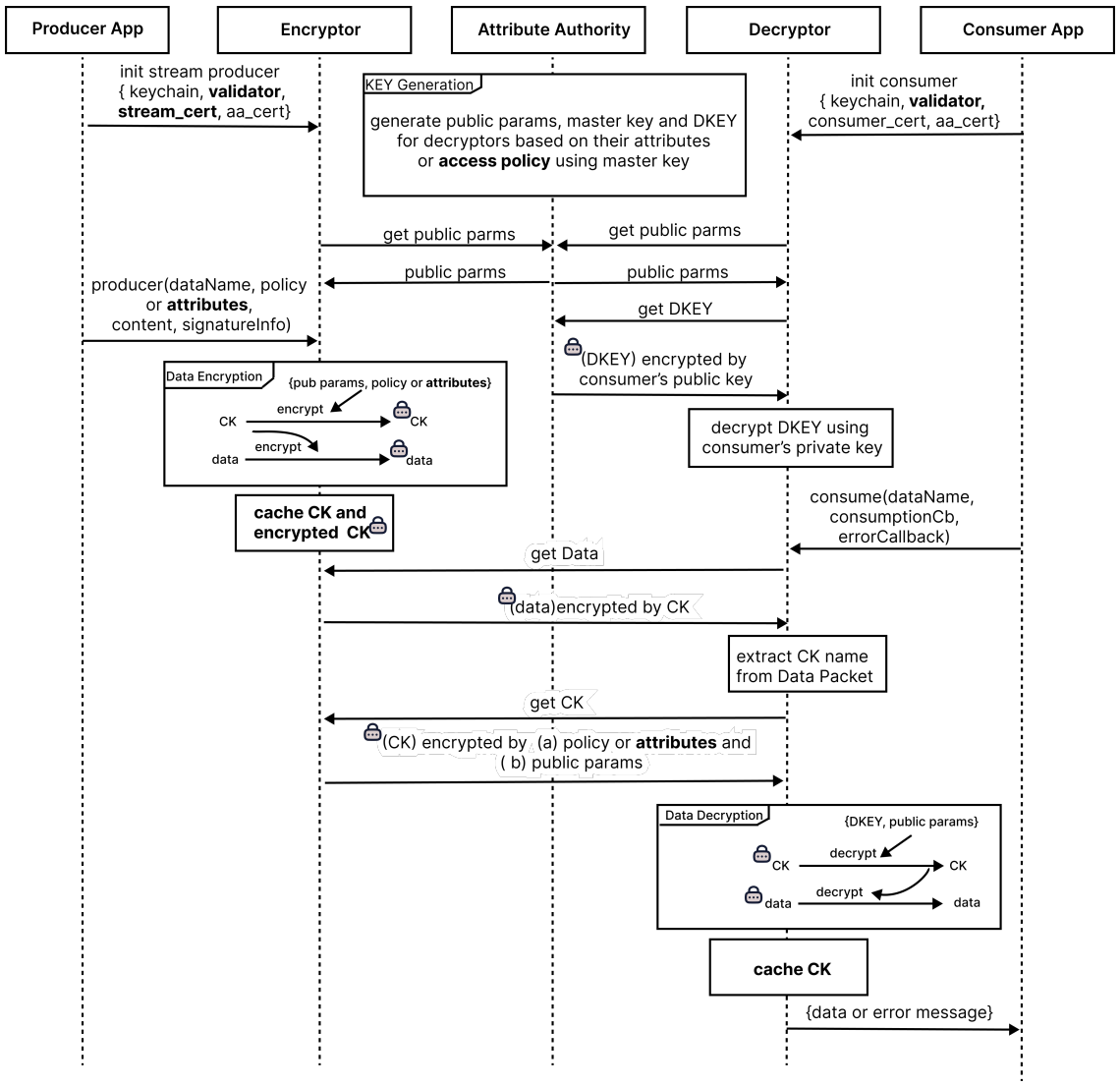


Fig. 3. NAC-ABE Sequence Diagram (our changes are highlighted in bold.)

DKEY to the nutritionist; no CKs associated with previously published data need to be re-encrypted.

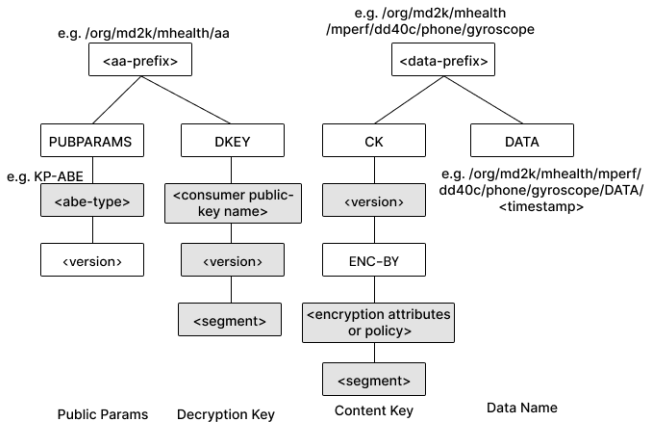
Second, KP-ABE eliminates the unintended data accesses illustrated earlier for CP-ABE. Specifically, Alice's blood glucose data collected at home is encrypted with the attributes "`<stream prefix>/blood-glucose`", "`home`". The researcher's DKEY encodes the policy ("`<stream prefix>/blood-glucose`" AND "`work`") OR ("`<stream prefix>/heart-rate`" AND "`home`"). The attribute set "`<stream prefix>/blood-glucose`", "`home`" satisfies neither clause: the first requires "`work`" rather than "`home`" for blood glucose data, while the second requires "`<stream prefix>/heart-rate`" rather than "`<stream prefix>/blood-glucose`". Consequently, access is correctly denied.

As illustrated in Figure 3, our design introduces three key changes. First, the Attribute Authority generates per-consumer DKEYs from access control policies rather than attributes. Second, producers supply data attributes, rather than access policies, to the encryptor. Finally, the encryptor uses these data attributes—rather than policies—to encrypt each CK.

B. Updated Naming Scheme

The naming scheme in NAC-ABE-2020 [6] deviates from the original design [5], leading to interoperability issues. We align it with the original design and extend it to support KP-ABE and key segmentation (Figure 4).

First, we introduce "`<abe-type>`" in public parameter names to support both CP-ABE and KP-ABE. Second, since each DKEY is encrypted with a consumer's public key, we use the public key name—rather than the consumer identity—to identify DKEYs, accommodating the rotation of the consumer's public key. Third, we add version and segment components to DKEY names for key evolution and segmentation. For CK naming, we (i) replace "`<random-word>`" with a version component, (ii) generalize "`<encryption policy>`" to


e.g. /org/md2k/mhealth/aa
<aa-prefix>
PUBPARAMS
DKEY
e.g. KP-ABE
<abe-type>
<version>
<consumer public-key name>
<version>

e.g. /org/md2k/mhealth /mperf/dd40c/phone/gyroscope
<data-prefix>
CK
DATA
<version>
ENC-BY
<encryption attributes or policy>

e.g. /org/md2k/mhealth/mperf/ dd40c/phone/gyroscope/DATA/ <timestamp>
Public Params
Decryption Key
Content Key
Data Name

Fig. 4. NAC-ABE Naming Scheme (additions shaded in grey).

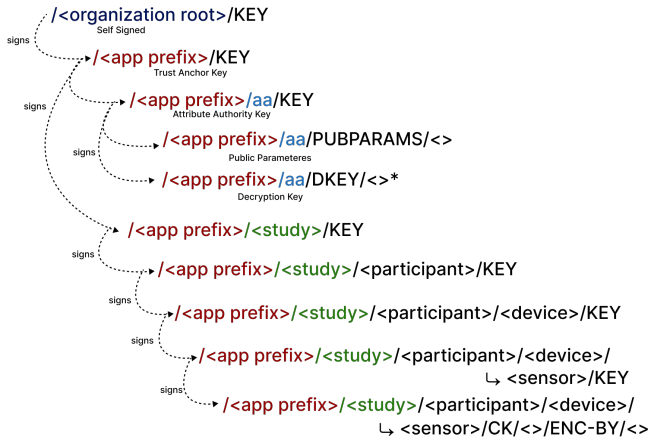


Fig. 5. Application trust schema example

"`<encryption attributes or policy>`" for KP-ABE support, and (iii) add segmentation.

Segmentation is necessary due to key size growth in NAC-ABE, which relies on the OpenABE library [18]. Time-based access control encodes timestamps as 32-bit UNIX integers, which OpenABE expands into $O(n)$ string attributes, increasing key size. While typical mGuard attributes (e.g., stream, location, and activity) keep CKs under 4KB, policies with additional time-based comparisons can produce larger CKs and DKEYs. When keys exceed the Maximum Segment Size, they are split into signed segments, each carrying a FinalBlockId. The decryptor retrieves segments using TCP-like slow-start and AIMD algorithm and validates each against an application-defined trust schema. The impact of attribute complexity on key size and performance is evaluated in Section IV.

C. CK Reuse, Caching, and Granularity

NAC-ABE-2020 generates a new CK for every data packet, incurring significant overhead for high-frequency mHealth streams. We address this by reusing CKs over a configurable time interval. On the encryptor side, a CK generated for a given attribute set is cached and reused for subsequent packets with the same attributes, avoiding redundant generation and encryption. On the decryptor side, decrypted CKs are cached locally to eliminate repeated retrieval and decryption.

To support this mechanism, each data stream is assigned a CK time granularity—the interval over which a CK is reused—which must be no coarser than the access-control granularity. For example, if a trainer is authorized to access heart rate data only from April 7, 2025 onward, a monthly CK would incorrectly expose data from earlier dates. Setting the granularity to daily aligns key reuse with the policy boundary, while finer granularities (e.g., hourly or per-second) further limit exposure at the cost of additional overhead.

Our implementation supports CK granularities ranging from per-second to yearly. Fine-grained settings are suitable for sensitive data (e.g., blood glucose) to minimize exposure upon key compromise, whereas coarser settings reduce overhead for less sensitive data (e.g., step counts). This tradeoff is quantified in Section IV.

D. Threat Model and Security Improvements

We assume an adversary who may impersonate a legitimate consumer, producer, repository, or controller; inject forged packets; intercept or fetch data and keys; and compromise system keys. The trust anchor itself is assumed to be secure.

Digital signature verification is a fundamental security mechanism in NDN, providing data authenticity and integrity. Building on NDN's semantically meaningful naming structure, trust schemas [19] automate signature validation by specifying which keys are authorized to sign particular data. Although NAC-ABE-2020 signs packets, it does not validate them against a trust schema, leaving the system vulnerable to impersonation and unauthorized data injection. To address this limitation, we introduce a validator interface in NAC-ABE. Applications initialize the validator with an application-specific trust schema, and all retrieved objects are validated before use.

Figure 5 illustrates the trust hierarchy used in mGuard. A trust anchor (e.g., a research center conducting an mHealth study) signs the Attribute Authority's public key, which in turn signs public parameters and DKEYs issued to authorized consumers. The trust anchor also signs the study's public key, which signs participant public keys. Participants sign device public keys, devices sign sensor public keys, and sensors sign the CKs used to encrypt their data streams. Public parameters, DKEYs, certificates, CKs, application data, and policy updates (for CP-ABE) are all verified against this trust hierarchy. Any packet that fails validation is rejected, preventing impersonation and unauthorized data injection.

Confidentiality is preserved even if an attacker can retrieve encrypted objects. An encrypted DKEY cannot be decrypted without the corresponding consumer's private key, and encrypted mHealth data remains inaccessible without the appropriate DKEY.

The impact of key compromise is limited by the scope of the compromised key. A DKEY is valid only within its authorized

TABLE I
SYMMETRIC ENCRYPTION AND DECRYPTION TIME VS. DATA SIZE.

Data Size (B)	Enc. Size (B)	Encrypt (ms)	Decrypt (ms)
82	379	0.006	0.011
902	1,474	0.022	0.029
1,805	2,690	0.024	0.044
3,605	5,081	0.035	0.081
5,409	7,490	0.045	0.120

TABLE II
CK OPERATION COSTS AND SIZE VS. ATTRIBUTE SET. (S=STREAM, T=TIME, L=LOCATION, A=ACTIVITY)

Attrs	Gen (ms)	Enc (ms)	Dec (ms)	Size (B)
S	1.89	2.92	10.01	841
S, L	1.89	3.22	10.20	981
S, T	1.81	15.43	23.64	3,260
S, T, L	1.81	15.26	23.36	3,400
S, T, L, A	1.89	16.30	24.30	3,613

time window when time attributes are included in the access policy. A CK is restricted to a specific attribute set and data granularity window. Likewise, compromising a private key affects only the downstream keys and data protected by that key. Consequently, the effects of compromise are localized rather than system-wide. Revocation in the current design relies on these bounded validity periods and therefore provides eventual rather than immediate revocation. We discuss the implications of this design choice and potential mitigations in Section V.

IV. EXPERIMENTAL EVALUATION

To evaluate the performance of our enhanced NAC-ABE system in mGuard, we conducted experiments using the Mini-NDN emulator [10]. Mini-NDN allows per-node CPU and memory limits to be configured, enabling experiments with the actual production code under resource footprints comparable to wearables and edge devices.

A. Experimental Setup

We emulated mHealth data collection using the Cerebral Cortex library [20], which provides representative sensor streams at varying frequencies. Both low-frequency (1 Hz) and high-frequency (100 Hz) streams were used to evaluate system behavior under different data rates. Access policies were defined by the controller with hourly, minute-level, or second-level CK granularity.

The mGuard Controller, Producer, Consumer, and NDN Data Repository [21] were deployed on separate nodes within an emulated NDN testbed topology [22]. All experiments were conducted on a desktop machine equipped with an Intel Core i7-11700KF (16 cores), 32 GB RAM, running Ubuntu 20.04.

B. Cryptographic Costs

Table I reports the cost of encrypting and decrypting individual data packets using AES-protected CKs as a function of data size. Both operations scale linearly with packet size and remain well below 1 ms across all tested sizes, indicating that per-packet symmetric encryption and decryption are not a performance bottleneck, even for high-frequency data streams.

Table II reports the cost of CK-related operations for different attribute sets. CK generation is both fast and insensitive to the number of attributes, remaining around 1.8 ms in all cases. In contrast, adding a time attribute increases KP-ABE encryption time from approximately 3 ms to 15 ms and expands the encrypted CK size from less than 1 KB to more than 3 KB. This overhead arises from OpenABE's representation of UNIX timestamps as integer attributes (Section III-B). For a typical mGuard attribute set, CK encryption using KP-ABE dominates the per-packet cryptographic cost, requiring roughly 16 ms compared to only 0.045 ms for AES data encryption. Consequently, the number of CK encryption and decryption operations is the primary determinant of system performance.

TABLE III
CK COUNT AND COST PER HOUR: NAC-ABE-2020 (PER-PACKET) VS. NAC-ABE (OURS) WITH TIME-BASED GRANULARITY.

NAC-ABE-2020			Updated NAC-ABE		
Freq.	CKs	Cost (s)	Gran.	CKs	Cost (s)
1 Hz	3,600	146.2	1 sec	3,600	146.2
10 Hz	36,000	1,461.6	1 min	60	2.4
100 Hz	360,000	14,616.0	1 hour	1	0.04

Note: for updated NAC-ABE, CK counts are identical across all tested frequencies.

C. CK Granularity

Table III compares CK overhead between NAC-ABE-2020, which generates one CK per data packet, and our time-based approach. In NAC-ABE-2020, a 100 Hz stream requires 360,000 CKs per hour, incurring over 4 hours of CK computation for one hour of data. Our approach decouples CK generation from sampling frequency: the number of CKs depends only on the chosen time granularity, dropping from 3,600 at second-level to 60 at minute-level to a single CK at hourly granularity. This reduces CK computation overhead for 100 Hz streams from over 4 hours to just 0.04 seconds.

D. Summary

Our results validate the design choices in Section III: per-packet encryption is sub-millisecond, CK operations dominate cost, and CK reuse combined with configurable granularity reduces CK overhead from over 4 hours to 0.04 seconds per hour of 100 Hz data.

V. DISCUSSION

A. Access Revocation

Access revocation is a critical requirement in mHealth deployments. Regulatory frameworks such as HIPAA mandate that participants can withdraw consent at any time, and revocation must take effect promptly. The eventual-revocation behavior described in Section III-D falls short of this requirement, particularly when a participant withdraws consent mid-study.

Potential directions include: (1) shorter CK and DKEY time windows to bound exposure, at the cost of more frequent key

re-issuance; (2) proxy re-encryption to transform ciphertexts without decryption, avoiding full re-encryption; and (3) integration with an NDN-native revocation notification mechanism that triggers consumer-side CK and DKEY invalidation. Each involves tradeoffs between revocation latency, cryptographic overhead, and deployment complexity that warrant further investigation.

B. Resource Constraints

The overhead of time-based attributes in OpenABE motivates exploring alternative ABE libraries or attribute encoding schemes that avoid the O(n) integer expansion. Deployment on resource-constrained mobile devices presents additional challenges in key storage and battery consumption that warrant further investigation.

C. Applicability beyond MHealth

While mGuard targets mHealth, the enhancements to NAC-ABE are application-agnostic. Any NDN application that produces data with known attributes but evolving access policies benefits from KP-ABE. Examples include smart building where sensor streams (HVAC, occupancy, energy) carry attributes such as floor, zone, and time, with changing tenants and manager access; military sensor networks [5], where unit, mission, and access tier attributes evolve over time; and vehicular NDN, where insurance companies, city traffic systems, or emergency responders access data streams (e.g., speed, location, and diagnostics) produced by vehicles under different policy constraints.

VI. Conclusion

In this work, we enhanced NAC-ABE to support fine-grained access control for real-time mHealth data sharing by addressing four limitations of NAC-ABE-2020: CP-ABE's policy-at-encryption constraint, shortcomings in the naming scheme, per-packet CK overhead, and the absence of trust schema validation.

These enhancements are more than incremental engineering improvements. They illustrate a broader principle: NDN's data-centric security model, in which trust is attached to named data rather than communication channels, provides a natural foundation for fine-grained, attribute-based access control in mHealth environments. The improved NAC-ABE enables access control policies to be expressed directly over name-based attributes, and its trust schema validation enforces data provenance at the granularity of individual data streams. Both capabilities derive from NDN's semantically meaningful naming structure and would be considerably more difficult to realize in a channel-centric security model.

At the same time, the revocation challenge discussed in Section V points to a deeper open problem. Effective mHealth systems require *consent-aware* access control, where a participant's withdrawal governs not only future data access but also the handling of previously published data. This challenge extends beyond cryptographic key management to the broader domain of data governance, motivating future research on NDN-native revocation mechanisms that treat consent as a first-class architectural concern.

References

[1] Grand View Research, "Wearable technology market size and trends, 2025–2030," 2025, accessed: 2025-05-15. [Online]. Available: https://www.grandviewresearch.com/industry-analysis/wearable-technology-market

[2] I. Sim, "Mobile devices and health," *New England Journal of Medicine*, vol. 381, no. 10, pp. 956–968, 2019.

[3] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[4] S. Dulal, N. Ali, A. R. Thieme, T. Yu, S. Liu, S. Regmi, L. Zhang, and L. Wang, "Building a secure mhealth data sharing infrastructure over ndn," in *Proc. ACM Conf. Inf.-Centric Netw. (ICN '22)*, 2022.

[5] Z. Zhang, Y. Yu, S. K. Ramani, A. Afanasyev, and L. Zhang, "NAC: Automating access control via Named Data," in *IEEE Military Communications Conference (MILCOM)*, 2018, pp. 626–633.

[6] Y. Zhang, Z. Zhang, and Y. Tu, "CP-ABE based NAC-ABE implementation," 2020. [Online]. Available: https://github.com/UCLA-IRL/NAC-ABE/tree/02961152f5ae2b15b3cddb917e8923f6c3c256bc

[7] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *2007 IEEE symposium on security and privacy (SP'07)*. IEEE, 2007, pp. 321–334.

[8] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006, pp. 89–98.

[9] "NAC-ABE Library GitHub Site," https://github.com/UCLA-IRL/NAC-ABE.

[10] NDN Project Team, "Mini-NDN GitHub," https://github.com/named-data/mini-ndn.

[11] R. Tourani, S. Misra, T. Mick, and G. Panwar, "Security, privacy, and access control in information-centric networking: A survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 566–600, 2018.

[12] B. Nour, H. Khelifi, R. Hussain, S. Mastorakis, and H. Moungla, "Access control mechanisms in named data networks: A comprehensive survey," *ACM Computing Surveys*, vol. 54, no. 3, pp. 1–35, 2021.

[13] Y. Yu, A. Afanasyev, and L. Zhang, "Name-based access control," *Named Data Networking Project, Technical Report NDN-0034*, 2015.

[14] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology–EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005. Proceedings 24*. Springer, 2005, pp. 457–473.

[15] J. A. Akinyele, M. W. Pagano, M. D. Green, C. U. Lehmann, Z. N. Peterson, and A. D. Rubin, "Securing electronic medical records using attribute-based encryption on mobile devices," in *Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM)*, 2011, pp. 75–86.

[16] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of ABE ciphertexts," in *USENIX Security Symposium*, 2011.

[17] B. Li, D. Huang, Z. Wang, and Y. Zhu, "Attribute-based access control for icn naming scheme," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 2, pp. 194–206, 2016.

[18] "OpenABE Library GitHub Site," https://github.com/zeutro/openabe.

[19] Y. Yu, A. Afanasyev, D. Clark, K. Claffy, V. Jacobson, and L. Zhang, "Schematizing trust in named data networking," in *proceedings of the 2nd ACM Conference on Information-Centric Networking*, 2015, pp. 177–186.

[20] N. Ali, A. Thieme, and S. Regmi, "Cerebral cortex random data generator — mguard extensions," https://gitlab.com/netlab-memphis/mguard-group/CerebralCortex-Random-Data-Generator, 2025, modified workload generator used in mGuard experiments.

[21] NDN Project Team, "A Named Data Networking (NDN) Repo implementation using python-ndn," https://github.com/UCLA-IRL/ndn-python-repo.

[22] N. P. Team, "NDN Testbed Topology," https://named-data.net/ndn-testbed/.