WiP: AABAC - Automated Attribute Based Access Control for Genomics Data

David Reddick
Tennessee Tech University
 Cookeville, Tennessee, USA
dereddick42@tntech.edu

Frank Alex Feltus
Clemson University
Clemson, South Carolina, USA
ffeltus@clemson.edu

Justin Presley
Tennessee Tech University
jcpresley42@tntech.edu

Susmit Shannigrahi
Tennessee Tech University
 Cookeville, Tennessee, USA
sshannigrahi@tntech.edu

ABSTRACT
The COVID-19 crisis and the subsequent vaccine development have demonstrated the potential of cutting-edge genomics research. However, the privacy of these sensitive pieces of information is an area of significant concern for genomics researchers. The current security models make it difficult to create flexible, automated, and secure data-sharing frameworks. These models also increase the complexity of adding or revoking access.

In this work, we investigate an automated attribute-based access control (AABAC) model for genomics data over Named Data Networking (NDN). AABAC secures the data itself rather than the storage location or transmission channel, provides automated data invalidation, and automates key retrieval and data validation while maintaining the ability to control access. When combined with NDN, we show that AABAC offers a secure and flexible access control framework.

CCS CONCEPTS
• Security and privacy → Key management. • Networks → Network architectures.

KEYWORDS
Named Data Networking, Genomics Access Control, CP-ABE

1 INTRODUCTION
As exemplified by COVID-19 and the subsequent rapid vaccine development, genomics research has the potential to revolutionize healthcare. With computing becoming cheaper and genome sequencing machines becoming ubiquitous, the genomics community is generating a massive number of valuable, geographically distributed datasets. Researchers often desire to share those datasets with the research community and healthcare providers. However, as genomics data becomes larger and more distributed, an acute problem arises - the complexity of sharing data with other researchers while providing fine-grained and easy access control.

Consider this example: a Principle Investigator (PI) wants to share access to a restricted dataset with a new graduate student. With a traditional Public Key Encryption model (PKI), either the data needs to be stored unencrypted in a "secure" location that only the PI and the students can access, or every person needs to have a copy of the data encrypted with their public key, creating yet another copy of the data. Genomics data is rapidly approaching Exabytes, and this approach of making multiple copies of the data is not sustainable [10][8]. While traditional attribute-based access control methods have been proposed, they suffer from performance bottlenecks and the complexity of key discovery and retrieval [2]. In the genomics community, data access revocation is generally achieved by revoking access to the storage location. However, access control based on file permissions does not work when supersusers have access to all the directories on a system.

As the genomics community moves towards the cloud computing model where the hosts and computing platforms are potentially untrusted, the data itself must be secured both in transit and at rest. While a large-scale confidentiality breach for genomics has not been documented, it is an active concern for individuals in the field [9]. Finally, the act of access control by centralized re-encryption and key revocation may not scale.

Building on our previous work [11], we propose a novel scheme that addresses these problems through an attribute-based access control model supported by Named Data Networking (NDN). We have worked with domain scientists to understand their requirements better. Our contributions are threefold: (a) an attribute-based encryption scheme that invalidates data after a specific time, enabling time-based control access, (b) a hybrid access model using...
the combination of local and remote ledgers that allow both data publisher and institutes to control access to published data, an essential requirement for the genomics community, and (c) automated and simplified key discovery, delivery, and verification based on the content names. Our approach has several advantages over traditional methods. First, we allow both the publisher and trusted collaborators (such as an administrator) to control access to data. For example, when a student graduates and no longer needs access to the data, the university can revoke access without involving the publisher. However, unlike today, the administrators do not gain access to the actual data. Second, in NDN, the decryption keys are linked to the data itself, automating key retrieval and data decryption. Finally, a time-based, partial re-encryption model maintains confidentiality without incurring a significant overhead.

2 RELATED WORK

2.1 NDN Background

NDN is a future Internet architecture that bases its communication protocols on content names rather than IP addresses of hosts for delivery [13][12]. NDN also facilitates other in-network features such as caching, multicast, and location-agnostic data retrieval. All data is signed by the publisher, ensuring data integrity. A human-readable, hierarchical, and application-defined naming scheme is used to identify content and in-network operations. This flexibility in naming allows integration with existing naming schemes, such as those used for scientific data [8]. A consumer sends an Interest packet that is forwarded based on the content name prefix to a data source to get Data packets. Once the Interest reaches a data source, the Data packet follows the return path. For brevity, we do not discuss NDN in more detail but refer the reader to previous work [12][8].

2.2 Access Control for Genomics Data

Previous efforts have tried to address access control for genomics data. Brewster et al. [4] have presented an ontology-based access control for distributed scientific data. This work provides access control to data using a role-oriented database that stores data attributes and their relationships. Mukherjee et al. [5] talk about Fast Healthcare Interoperability Resources (FHIR), a standard for swift and efficient storage/retrieval of health data. This effort uses an Attribute-Based Access Control (ABAC) system to act as a middle layer between the client application and the FHIR server to facilitate fine-grained access to FHIR resources. Narouei et al. [7] introduced a top-down engineering framework for ABAC that autonomously retrieved policies from unrestricted natural language documents and then used deep neural networks to extract policy-related data. Namazi et al. [6] presented work on developing an attribute-based privacy-preserving susceptibility method that outsourced genomics data to an unreliable platform. The computations’ challenges were determined to process the outsourced data and grant access concurrently within patient-doctor interactions.

However, none of these works address a crucial gap in access control for genomics data. Unlike healthcare settings where data is small in size and potentially shared with a small number of people (e.g., medical team and patient), research collaborations require flexibility where participants often change and a hybrid access control is desired.

3 THE CURRENT DATA SECURITY MODEL

Figure 1 shows a representative data security model currently used by the genomics community. We developed this model in collaboration with genomics researchers. While some details might differ based on the actual use case across (sub)communities, the figure should provide a general overview. First, an institutional review board reviews the request for privacy-sensitive data. Once approved, the Principal Investigator (PI) then requests access to the data. Depending on the type of data, this can be hosted at another institute or with a group such as the National Institutes of Health (NIH). The PI needs to name the graduate students and anyone they want to give access to at the time of this request and add them to the IRB. Once the request is approved, the PI securely transfers data into a secure local location. The location of the data can then be secured in various ways, such as file system permissions, Linux group restrictions, or some custom access control method. If a student needs access to the data, the PI adds the student to the Linux group. When computations need to run on the data, the data is securely transferred (TLS/SSL) over to a secure computational facility. The results are then securely written back to the secure directory.

Adding a layer of encryption to data at rest would assist security but is not easily implemented with the current model. One approach is to share a private key among the students and their collaborators, which is not recommended and complicates the ability to revoke access. The alternative approach is to create a per-person copy of the data, which does not scale.
4 AABAC DESIGN

Once the IRB is approved, the PI’s institute and the data publisher agree on the attributes for data encryption. In this example, the published data is named as “/Genome1/SRA/9605/9609/RNA-Seq/1” and encrypted with mutually agreed-upon attributes such as “PI and PI’s graduate students”. The data publisher trusts the PI to control access to the sensitive data and does not wish to be involved in issuing or revoking decryption keys. The PI utilizes his/her university’s authentication system to issue or revoke keys. We also assume there is a Network Operation Center (NOC) that both the data publisher and the university trust. This NOC is in charge of issuing keys to the users and publishing the keys for data encryption. Finally, to assist with understanding the NDN naming schemes discussed in this section, Table 1 is provided to outline the names in AABAC.

### 4.1 Key Generation and Publication

In AABAC, the NOC is in charge of creating the master key ($m_k$) and distributing the public key ($p_k$). $p_k$ is used for encrypting data while the $m_k$ is used for creating decryption keys. Once these keys are generated, the public key is published into an NDN network where anyone can request and utilize these keys for encryption. Since NDN is location agnostic, the key can be published to a repository or cached in the network. The master key is not published. The public key can be named as “/genomics/pub_key/sequence=random_number”, where the seq is a random number used to distinguish between different $p_k$s.

### 4.2 Namespace and attribute mapping

During the IRB review, the PI and the data publisher agree on attributes. In our example, “/genomics/data/” may have attributes “PI=Tom or PI’s graduate student”. These attributes can also be published into an NDN network under “/genomics/data/attributes”.

### 4.3 Data Publication

When a file named “/Genome1/SRA/9605/9609/RNA-Seq/1” is published, the publisher will request a $p_k$ from the NDN network. This key may be a specific or random key provided by the network. After this step, the data publication process is simple: the $p_k$ is applied to the data to create encrypted content $c$. This encrypted data is also published into the NDN network under a name such as “/Genome1/SRA/9605/9609/RNA-Seq/1/encrypted_by=/genomics/pub_key/timestamp=1645780366”. Once data is encrypted, it can be published in an NDN repo that makes the data available for anyone asking for the content. As we discussed earlier, a file in NDN is divided into several Data packets and signed. In AABAC, each Data packet is individually encrypted before being digitally signed by the publisher. The signature provides provenance and enables us to publish this data from anywhere. While data is available from anywhere, it is encrypted and can not be used unless the key with proper attributes is available. Being encrypted before publication ensures that even if unauthorized users can locate the published data, they will be unable to utilize the encrypted data.

The other important part of data publication is providing a pointer to the local (institutional) attribute authority through which the user can ask for a decryption key. In NDN, this is also accomplished by using a namespace. When data is published, the name of the decryption key service (local ledger) is also associated with the data. “/genomics/data/sra/annotations: encrypted_by=/genomics/pub_key/timestamp=<time>/<local_ledger=tntech/ledger”

### 4.4 Data Retrieval

In NDN, data can be retrieved by sending an Interest into the network. For accessing “/Genome1/SRA/9605/9609”, a user sends the Interest by that name and receives the encrypted data. The data can come from anywhere: from the publisher, an intermediary repo, or an in-network cache.

### 4.5 Decryption Key Generation and Retrieval

Once the user (let us call her Alice) receives the data, she looks at the annotations in the name. Note that Alice can read the name of the data she received but can not decrypt the payload yet. Alice (or the application Alice is using) needs to request a decryption key ($d_k$). From the forwarding hint in the name, Alice knows she needs to request the $d_k$ from “/tntech/ledger”. She sends an Interest to the local ledger in the form of “/tntech/ledger/decryption-key/data: genomics/data/sra/attributes:/Alice-pub-key:tntech/alice/pub_key”, where attributes are “PI and PI’s graduate students”. She also signs the request with her private key; this way, the ledger knows the request to be authentic. Tennessee Tech’s ledger looks up Alice’s attributes on receiving this request. If Alice is a graduate student working under PI Tom, she will have both attributes in the ledger. The ledger will sign and forward this request to the NOC. Such a request would look like: “/tntech/ledger/decryption-key/data:genomics/data/sra/attributes:/Alice-pub-key:

| Table 1: NDN names used to facilitate AABAC. |
|---------------------------------------------|
| Content Name                              |
| Encryption key name                       |
| Local ledger locator                      |
| Name forwarded to NOC                     |
| Name of NOC’s reply with decryption key   |

| Name of NOC’s reply with decryption key   | /tntech/ledger/decryption-key_alice/genomics/data/SRA/timestamp=1645780366 |

| Encryption key name | /genomics/data/SRA/9605/… |
|---------------------|---------------------------|
| Local ledger locator| /genomics/pub_key/sequence=random_number |
| Name forwarded to NOC| /tntech/ledger/decryption-key/data:genomics/data/SRA/attributes:/alice-pub-key/tntech/alice/pub_key |
| Name of NOC’s reply with decryption key | /tntech/ledger/decryption-key_alice/genomics/data/SRA/timestamp=1645780366 |
Note that the local ledger can also add additional attributes, such as the validity period of the requests. The NOC will generate a decryption key for Alice using the attributes and the ABE keys on receiving the key:

$$\text{masterkey} + \text{publickey} + \text{attributes} = \text{decryption} - \text{key}_{\text{alice}}$$

The NOC and the local ledger establish the trust beforehand, and only signed requests from the ledger will create a decryption key. If Alice directly requests the decryption key, the NOC will not respond since it does not trust Alice.

On receiving the request from the local ledger, the NOC generates and encrypts the decryption key using Alice’s public key located at “/tnitech/alice/pub_key”. The NOC has two choices to return the key to Alice. The first way is to reply to the local ledger which then returns the key to Alice. The second way is it publishes the key into the NDN network “/tnitech/ledger/decryption-key_alice/genomics/data/sra” and encrypts the decryption key using Alice’s public key located at “/tnitech/alice/pub_key”. Alice then requests the key from the network. Either way, Alice can receive the key. In our implementation, we use the local ledger for the distribution of keys.

Note that the key generation and retrieval is a lightweight operation. The application stores the decryption key locally and utilizes it in the future. When a new key is needed, the application retrieves a new key. The NOC, data publishers, and accessing institutes establish the granularity and lifetime of these keys. For example, setting access control attributes over a broader namespace (e.g., /genomics) would require less decryption key generation than setting access control over more specific namespaces (e.g., /genomics/data/sra).

### 4.6 Timing attribute and partial re-encryption for revoking access

One of the challenging parts of attribute-based encryption is access revocation. Since genomics data is long-lived, utilizing different keys as data is generated is not feasible. On the other hand, re-encrypting data frequently to revoke access is also not cost-effective. There are two distinct threat models that we aim to address. First, a superuser or an intermediary should not be able to access the data even though they can access raw files. Second, a graduate student or other collaborator working on sensitive data should no longer have access to the data after leaving the institution. The key management problem arises when user access needs to be revoked. AABAC uses a time-based attribute between the local ledger and the NOC to enforce this.

Here is an example: a student named Alice requests a key at Time $T_{10}$ and the data is encrypted using attributes “PI and PI graduate student” and Timestamp $T_{10}$. The attributes that the local ledger will send to the NOC are “PI and PI’s graduate student and Timestamp: $T_{10}$”. Note that in NDN, a file is made of several smaller Data packets. If the Data packets were encrypted and published at $T_{9}$ with a time attribute, Alice would be able to decrypt the individual packets and reassemble the file. However, if the Data packets of a file are published at $T_{11}$, Alice will not be able to decrypt the Data packets. We worked with the genomics scientists to understand the parts of the files that are more critical. Rather than re-encrypting the whole file, we periodically re-encrypt the file metadata as well as random Data packets and update the data repository. If a file is divided into two Data packets (an example, a file would likely be divided into thousands of Data packets) with timestamps $T_{10}$ and $T_{11}$, and Alice requested a key at $T_{10}$, Alice can decrypt the packet with timestamp $T_{10}$ but not $T_{11}$. Since the Data packets already have a key locator built-in, Alice will then request the new key $T_{11}$ to be able to decrypt the data. Note that Alice only needs to request one key with our scheme; a key with $T_{11}$ will be able to decrypt both $T_{10}$ and $T_{11}$. If Alice is no longer authorized to decrypt the data, the local ledger will not forward the request to the NOC to get newer keys to continue decrypting future versions.

The other thing to note here is that NDN allows us to set content lifetime on Data packets. Even though NDN caches content, we can ensure data with older timestamps will not be available from in-network caches by setting content lifetime to a value lower than re-encryption time. Our experience shows that encrypting the metadata and a random portion of the data is sufficient to preserve data privacy. Even if we perform full re-encryption, the average re-encryption requires only around 15 seconds for a 2GB file.

![Figure 2: Encryption and Decryption Model](image-url)

### 4.7 System Model

Figure 2 provides an overview of the proposed encryption and decryption model.

There are a few main parts: the NOC, the publisher, the user, the remote ledger, the local ledger, and the NDN network that facilitates the NDN repository. This paper also utilizes CP-ABE for the encryption and is explained in more depth in the paper by Bethencourt et al. [3]. The first step is understanding the NOC, which generates the CP-ABE keys and maintains control of the master key, which is needed to generate user keys. For a publisher to encrypt and publish data, the first step is to reach out to the NOC and get the CP-ABE public key needed to encrypt the data. The publisher can encrypt data with various attributes in a security policy using this key. For initial deployment, all data will be encrypted, but, in the future, segments of the data will be periodically encrypted again to maintain security and republish to NDN. The next important entity is the user, when the user wishes to decrypt some data from the network, the user contacts the local ledger that authenticates permission and then passes on the request to the NOC to generate a user key with the current time attribute. Once the NOC confirms the request is valid, the NOC will generate a user key with the...
requested attributes and pass the new user key back to the user. Once the user gets the CP-ABE key containing their attributes, they can decrypt the data using the key until the data is partially re-encrypted.

### 4.8 A Possible Real-World Scenario

Having explored the motivation and design for the proposed system in this paper, this example demonstrates the system’s effectiveness in real-world situations. There are two PIs from different universities and departments at those institutions. Both PIs also have multiple graduate assistants working in collaboration on project Genome1. The data is also encrypted with periodically changing encryption keys with increasing sequences to revoke previous access. Based on this scenario, the attribute for providing access to the project-related resources would potentially rely on the university’s name, department, project, and time. The graduate assistants must be employed under the PIs and assigned to the specific project to access the resources. Anyone outside this group will not be able to update or view the data. The scheme for this would be as follows:

**Attributes**: Project, Principal Investigator, University, Department, Role, Time Sequence

**Requested Content Name**: "/Genome1/SRA/9605/9609/RNA-Seq/1"

**User receives content and requests decryption Key Name from Local Ledger**: "/Genome1/SRA/9605/9609/RNA-Seq/1
/DecryptionKey/Attributes/Name=John Smith
/Project=Genome1/University=TNTECH/Department=Biology
/Role=Graduate Assistant/timestamp=1645780366"

**Access Control Rule and Trust Schema**: Return user decryption key - 

\[(\text{Project} = \text{Genome1}) \land ((\text{PI} = \text{John Smith}) \land (\text{University} = \text{TNTECH}) \land (\text{Department} = \text{Biology} \lor \text{Department} = \text{Computer Science}) \land (\text{Role} = \text{Graduate Assistant})) \lor ((\text{PI} = \text{Jack Robinson}) \land (\text{University} = \text{UCLA}) \land (\text{Department} = \text{Biology} \lor \text{Department} = \text{Computer Science}) \land (\text{Role} = \text{Graduate Assistant})) \land (\text{timestamp} = 1645780366))\]

**Example**:

- **Student 1** - 
  \[(\text{Project} = \text{Genome1}; \text{PI} = \text{John Smith}; \text{University} = \text{TNTECH}; \text{Department} = \text{Biology}; \text{Role} = \text{Graduate Assistant}; \text{timestamp} = 1645780366)\] - Receives decrypted data
- **Student 2** - 
  \[(\text{Project} = \text{Genome1}; \text{PI} = \text{John Smith}; \text{University} = \text{UCLA}; \text{Department} = \text{Biology}; \text{Role} = \text{Graduate Assistant}; \text{timestamp} = 1645780366)\] - Does not receive decrypted data

### 5 EVALUATION

This section evaluates our framework in terms of performance and overhead. One of the criticisms of attribute-based encryption has been that they are slow. However, genomics data is long-lived, and we show that cost of encryption is manageable. We also find that the per-packet encryption time is low. We analyzed the metadata size of a Kidney dataset [1] samples to be on average under 17KB,
We ran experiments with a varying number of attributes from five to fifty. While increasing the number of attributes will affect the system performance negatively. For statistical accuracy, we repeated all experiments ten times.

5.1 Encryption time with CP-ABE

Performance is essential when working with large data sets that need to be published with a comparatively inefficient encryption algorithm compared to symmetric key encryption. We could encrypt a 1GB file in about one second using AES with OpenSSL in our testing. We ran multiple experiments to demonstrate that using CP-ABE directly instead of an intermediate symmetric key encryption is viable. The first test shown in Figure 3-A demonstrates the encryption time in milliseconds when working with standard NDN packet sizes that can vary between 500 and 8800 bytes. The figure indicates for these sizes that encryption can be accomplished in between 14 and 15 milliseconds (averaged over ten runs for each file size with a standard deviation of 0.259 milliseconds). The second experiment shows the total time needed to encrypt the most common genomics samples that average less than 2 GB each. Figure 3-B shows the average time in seconds for encryption of 500MB, 1GB, and 2GB files.

5.2 Number of attributes vs encryption time

When working with an encryption algorithm like CP-ABE, one concern is whether added complexity would affect performance. We ran experiments with a varying number of attributes from five to fifty to determine if this would prove to be a potential problem for some deployments. For this test, the file size was kept constant with an original file equaling 2GB, and the experiment ran for each number of attributes. As Figure 3-C shows while increasing the number of attributes does increase the encryption time in a predictable pattern, for the test file, all results range between 14.5 and 15 seconds. However, the number of actual attributes in a real deployment is likely to be less than fifty.

5.3 Overhead of encryption

When working with CP-ABE to secure genomics data, the final concern studied is the file size overhead when encrypting. All encryption algorithms will add some overhead, but the degree of overhead can vary between algorithms. We ran experiments to determine the degree of file overhead for a 2GB file depending on the number of attributes. Figure 3-D demonstrates the overhead in bytes over the original 2GB file for the number of attributes varying from five to fifty. While increasing the number of attributes will increase the encryption overhead, this overhead will require only one or two extra NDN packets for delivery when working with less than fifty attributes.

6 CONCLUSIONS AND FUTURE WORK

The vast availability of genomics data has highlighted the need to ensure security and privacy when sharing sensitive healthcare information. This paper introduced attribute-based access control for genomics data along with proposed policies. These policies can be implemented by academic institutions working on genomic research. We present a possible real-world scenario showing that AABAC does perform well with reasonable overhead. In future work, we plan to evaluate our work with an actual genomics workflow.

ACKNOWLEDGMENT

This work is funded by National Science Foundation (NSF) grants OAC-2019163, OAC-2126148, and OAC-2019012. All opinions and statements in the above publication are of the authors and do not represent NSF positions.

REFERENCES

[1] 2017. Homo sapiens (ID 359795) - BioProject - NCBI. https://www.ncbi.nlm.nih.gov/bioproject/359795. Online; accessed 28. April 2022.
[2] A. Abinaya and S. Santhi. 2021. A survey on genomic data by privacy-preserving techniques perspective. Comput. Biol. Chem. 93 (Aug 2021), 107588. https://doi.org/10.1016/j.compbiolchem.2021.107588
[3] John Bethencourt, Amit Sahai, and Brent Waters. 2007. Ciphertext-Policy Attribute-Based Encryption. In 2007 IEEE Symposium on Security and Privacy (SP ’07). 321–334. https://doi.org/10.1109/SP.2007.11
[4] Christopher Breitzer, Barry Nouwt, Stephan Raaijmakers, and Jack Verhoosel 2020. Ontology-based Access Control for FAIR Data. Data Intelligence 2, 1-2 (01 2020), 66–77. https://doi.org/10.1162/dint_a_00029
[5] Subhrojeet Mukherjee, Indrakshi Ray, Indrajit Ray, Hasseun Shizuri, Toan Ong, and Michael G. Kahn. 2017. Attribute Based Access Control for Healthcare Resources. In Proceedings of the 2nd ACM Workshop on Attribute-Based Access Control (Scottsdale, Arizona, USA) (ABAC ’17). Association for Computing Machinery, New York, NY, USA, 29–40. https://doi.org/10.1145/3041948.3041955
[6] Mina Namazi, Chian Eryonucu, Erman Aydyan, and Fernando Pérez-González. 2019. Dynamic Attribute-Based Privacy-Preserving Genomic Susceptibility Testing. In Proceedings of the 34th ACM SIGSPATIAL Symposium on Applied Computing (Limasol, Cyprus) (SAC ’19). Association for Computing Machinery, New York, NY, USA, 1467–1474. https://doi.org/10.1145/3297280.3297428
[7] Masoud Narouei, Hamed Khounpour, Hassaan Takabi, Natalie Parde, and Rodney Nielsen. 2017. Towards a Top-down Policy Engineering Framework for Attribute-Based Access Control. In Proceedings of the 22nd ACM on Symposium on Access Control Models and Technologies (Indianapolis, Indiana, USA) (SACMAT ’17). Association for Computing Machinery, New York, NY, USA, 103–114. https://doi.org/10.1145/3078816.3078874
[8] Cameron Ogle, David Reddick, Coleman McNight, Tyler Biggs, Rini Pauly, Stephen P. Ficklin, F. Alex Feltus, and Susmit Shannigrahi. 2021. Named Data Networking for Genomics Data Management and Integrated Workflows. Frontiers in Big Data 4 (2 2021). https://doi.org/10.3389/fdata.2021.582468
[9] Mark Phillips, Fruszma Molnár-Gábor, Jan O. Korbel, Adriaan Thorogood, Yann Joly, Don Chalmers, David Townend, and Bartha M. Knoppers. 2020. Genomics data sharing needs an international code of conduct. Nature 2021 578:7793 578 (2 2020), 31–33. Issue 7793. https://doi.org/10.1038/s41586-020-00082-9
[10] Yanhe Qin, Stephan Kobler, Shengming Zhao, Bubin Mai, Zhou Liu, and Hao Lu. 2020. High-throughput, low-cost and rapid DNA sequencing using surface-coating techniques. bioRxiv (12 2020), 2020.12.10.418962. https://doi.org/10.1101/2020.12.10.418962
[11] David Reddick, F. Alex Feltus, and Susmit Shannigrahi. 2022. Case Study of Attribute Based Access Control for Genomics Data Using Named Data Networking. In IEEE CCNC.
[12] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, kc claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. 2014. Named Data networking. ACM SIGCOMM Computer Communication Review 44 (7 2014). Issue 3. https://doi.org/10.1145/2566877.2566887
[13] Lixia Zhang, Deborah Estrin, Jeffrey Burke, Van Jacobson, Jim Thornton, K. Di- ana, Beichuan Zhang, Gene Tsudik, Dan Mansey, Christos Papadopoulos, Patrick Crowley, James D Thornton, and Diana K Smetters. 2010. Named Data Networking (NDN) Project. (10 2010). http://named-data.net/techreports.html