* 1. **ICN**

(IITM, NPTEL-NOC, 2023), (Afanasyev, et al., 2018)

Information Centric Networking (ICN) is a content/data-driven approach to transferring information through a network. This is different from the traditional location-driven approach that TCP/IP adopts where IP addresses marking locations of sender and receiver are prerequisite to data transfer.

Following are 3 key ideas associated with ICN that facilitates achievement of main objectives of ICN being improving content accessibility, delivery efficiency, content availability and data authenticity.

1. **Content Naming**

Content naming is an integral part of ICN. It lies at the heart of content-driven data transfer.

Instead of senders and receivers in TCP/IP, ICN comprises puslishers and subscribers. Also, unlike TCP/IP where it’s the sender that drives transfer; here, it’s the receiver/subscriber that does this, as briefly explained below.

* When a **subscriber** requires a certain kind of data, it sends out an interest packet into the network with the “name” of the data it is looking for. The subscriber does not need to know where to find (at what address/device) to find this data.
* Data **publishers** create and introduce new data into the via **named** data packets.
* Devices in the network forward/propagate interest and data packets such that senders of interest packets with the same/similar “name” as existing data packets, receive them.

Thus, in ICN, content is retrieved based on names of desired data packets. This means content names must be globally unique and persistent. Following are popular namespace designs (strategies for naming data).

1. **Flat Naming:** This approach involves passing content to a hashing algorithm and then binding the resulting unique hash to the content along with a user generated name.
2. **Hierarchical Naming:** This convention is inspired from HTTP URLs where the name comprises portions each capturing some context regarding content separated by ‘/’s with added portions at the end that ensure uniqueness. Example: /edu/tcd/computer\_science/cs101/lecture.pdf/1 could be the name assigned to the 1st data packet related to a computer science lecture released by TCD. This type of naming is more human readable and is also more scalable than the previous one in that routers can match prefixes of the names instead of the entire name which in turn reduces size of forwarding tables.
3. **Attribute Based Naming:** This type of naming looks like HTTP GET requests. Names are created by combining various attribute values associated with a piece of data such as date, location, creator, etc.

This kind of data naming, in-effect decouples data from location, application, transportation, and storage (Datatracker, n.d.). This makes it possible to implement the following features.

1. **In Network Storage and Caching**

Nodes in an ICN network, may cache a copy of data packets that they receive and forward/use. This allows for replication. Thus, same data being available at different points in the network makes access more efficient (fewer hops required) and robust to failure (even one copy of data is unavailable, chances are, another node will have another copy of the same data). This replication and improved data availability and accessibility comes at the price of need for more storage.

1. **Content Level Security**

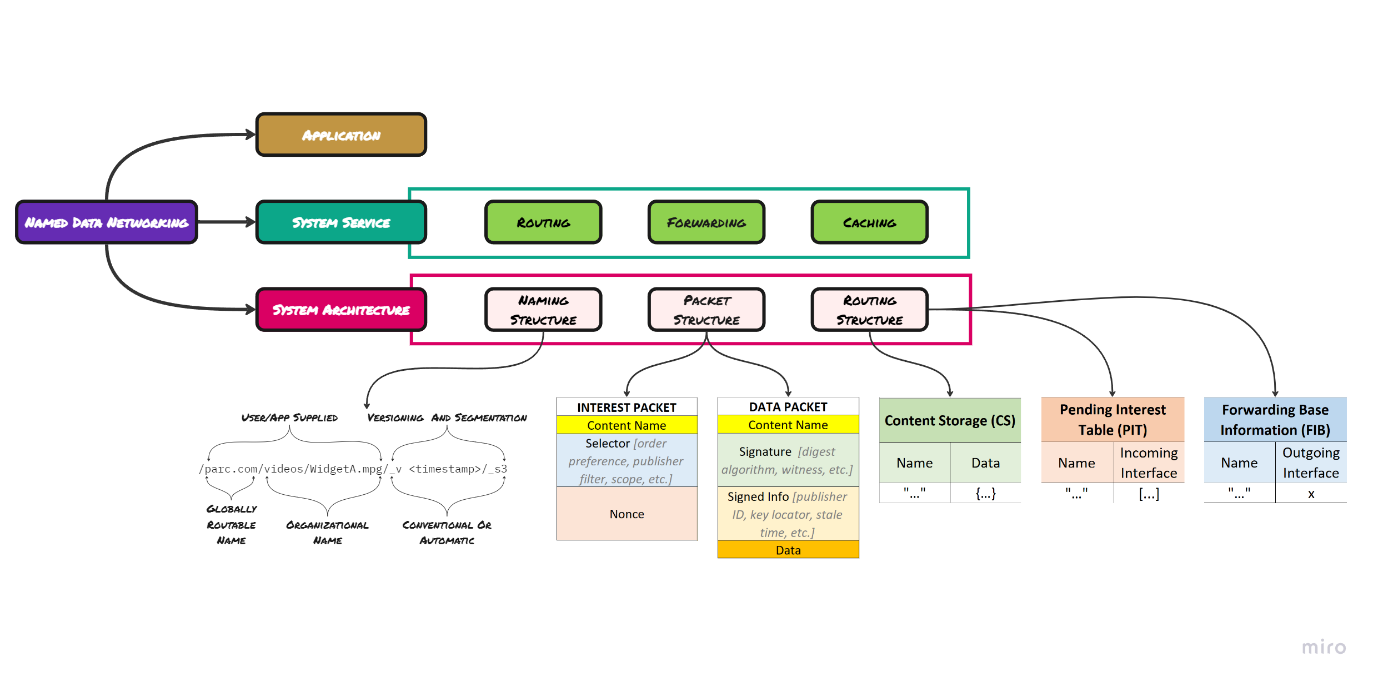
In TCP/IP security often involves securing the communication path though establishment of secure tunnels. But with ICN, it’s the data content itself that is secured (Fu, Kutscher, Misra, & Li, 2018). A simple approach to this would be to modify the classic Digital Signature Algorithm (DSA) (Scenes, 2023) each publisher adds a cryptographic signature to data packets. Subscribers verify signatures using the corresponding universal public key (all subscribers know about this key) to ensure content authenticity. Since the signature is a part of the data itself, every copy of it shall also contain the same signature. Thus, source of each copy of the data in the network can be verified (Bilal & Pack, 2020).

* 1. **NDN**

(Afanasyev, et al., 2018) (NPTEL-NOC IITM, 2023) (NPTEL-NOC IITM, 2023) (SG NDN Telkom University, 2021)

Named Data Networking (NDN) is a network architecture design that implements ICN principles.

Different applications may be built on top of the NDN architecture with characteristic naming, packet, and routing structures by leveraging system services like routing, forwarding, caching, etc, that it facilitates (SG NDN Telkom University, 2021).



**Naming Structure**

NDN does not enforce strict namespace management rules as part of the architecture. This gives one the freedom to experiment. As long as the end points in a network follow the same name assigning and querying convention, NDN can be adopted. That said, typically, NDN implementations follow a hierarchical naming convention.



**Packet Structure**

Packets may be of 2 types. An interest packet, and a data packet.

An interest packet comprises content name and can also contain conditions/rules for network devices to consider during forwarding/routing such as order in which data packets are expected, time for which this packet is valid, etc along with a nonce that facilitates linking of data packets to the interest packet that requested it.

A data packet contains content name and the data payload itself. Additionally, it may comprise meta information like the kind of content the packet contains (size, audio/video etc.), as well as security imbibing data like the signature, key locator, digest algorithm, etc.

**Forwarding/Routing Structure**

Forwarding/routing in an NDN setting is facilitated by 3 key tables/mappings as given below.



The Content Store (CS) maps content name to data received. This often acts as the cache. When a node does not contain requested data, it maps content identifier of received interest to one or more incoming interfaces in the Pending Interest Table (PIT) to keep track of all interested parties for a specific content. The Forwarding Information Base (FIB) table maintains a dictionary of interest packet content names mapped to the outgoing interface that it was forwarded to.

**Caching**

NDN implements in-network caching which means that network devices may choose to store a copy of content received within data packets locally. This means that on subsequent hits, a subscribers can find data previously requested by itself/another subscriber on a device that is physically closer to itself (reachable in few, ideally a single hop). This greatly reduces response time. However, there must be some mechanism to clear the cache from time to time to minimize storage overhead.

**Forwarding/Routing**

The core idea behind flow of packets through an NDN based network can be summarised as follows.

|  |  |
| --- | --- |
| *Receive Interest Packet* | *Receive Data Packet* |
| 1. Check CS. 2. If corresponding data is found in CS, send this data in a data packet to interested party/parties. 3. Else, add interested party to PIT. 4. Check FIB to see if known outgoing interface for this interest exists. 5. If so, forward packet out through this interface. 6. Else, pick a suitable interface (not the one the interest came from) to forward the packet to if possible. 7. If valid outgoing interface identified, populate FIB and forward interest packet our chosen interface. | 1. Check PIT for any parties interested in received content. 2. If no pending interest is found, drop packet. 3. For every pending interest found, forward data packet. 4. Possibly cache content in CS. |

All data transfer, staying true to ICN, is receiver driven. Whenever a node (subscriber) requires information, it sends an interest packet into the network for corresponding data packet. Network devices then forward this interest packet until it gets to a node which contains requested data (publisher/in-network cache). This data then arrives back to the subscriber via reverse path forwarding (right hand side column in above table).

**Our Implementation**

Every action team comprises 5 nanobots, each to detect one of 5 markers in cells indicative of cancer:

- "tumour" marker = This is the most important (primary) marker and refers to chemical markers most closely associated with particular kinds of cancer (e.g. Prostate-specific antigen (PSA) associated with Prostrate Cancer).

- "acidity" marker = Cancerous cells are more acidic in nature than healthy cells, hence this marker.

- "growth" marker = This marker represents growth factors like Insulin-like Growth Factors (IGFs) and the Epidermal Growth Factor (EGF) that mark uncontrolled growing nature of cancerous tissue.

- "survivin" marker = This protein counteracts growth inhibitors in the body that allows cancer to keep growing.

- "ecmr" marker = Extracellular Matrix Remodeling Enzymes like metalloproteinases (MMPs) and lysyl oxidase (LOX) help reshape surroundings of the cancer cell and aid in immune suppression.

The nanobot that detects the tumour marker is the primary bot which triggers collective diagnosis, while all others are non-primary/secondary bots.

We had demonstrated functioning of 1 such team with 5 bots.

There is also a 6th node in the network called a rendezvous server whose presence may be viewed as perhaps a smart watch worn by the patient. This server was added to simulate NP bots detecting beacon signals emitted by a P bot. If this were in real life, NP bots would simply sense the beacon when it got within range of it. Thus, the rendezvous server here, merely serves to pick up beacon data packets from P bots (active beacon actuator) and relay them the interested NP bots (searching for beacon).

The **blood stream** is captured by a circular array whose length, and speed of blood is predefined in the config.json file. Speed of a bot is computed as CONFIG['blood\_speed'] + self.\_\_actuators['head\_rotator'] + self.\_\_actuators['propeller\_rotator']. Movement related logic may be found in the move() function within the nanobot.py file.

**Nanobots** here, have the following attributes. *(AP means only the primary bot has attribute A and ANP means only non-primary bots have it)*

* Name: Unique name of the bot.
* Host: Host address of this bot
* Port: Port at which bot shall listen for communications.
* Position: Index in blood stream.
* Marker: Type of cancer marker that this bot is detecting which may be one of “tumour”, “acidity”, “growth”, “surviving” and “ecmr” as defined in config.json.
* Knowledge: Value of each kind of marker represented using a dictionary {“marker1”: -1, “marker2”: -1, …, } where -1 stands for unknown, 0 for negative and 1 for positive detection. This knowledge is what every bot tries to populate with information from peers about marker values that are not their own with the aim of collectively arriving at a diagnosis decision “healthy”/”cancer”).
* Neighbor Discovery Complete: Boolean indicating whether this bot has detected the 4 others in its team or not).
* Diagnosis: Decision made by bot which may be None = no decision, “cancer” or “healthy”).
* Sensors = {“cancer\_marker”: 0/1 indicating whether or not marker was found on tissue, “beacon”: index in bloodstream of detected beacon.}
* Primary BotNP: Information about primary bot (host, port, name).
* Actuators = {“tethers”: Boolean value where 0 means that the bot is moving with tethers retracted and 1 indicates that its tethers are extended to grab tissue and that it is fixed to a spot in the bloodstream array, “head\_rotator”: Float value indicating speed of spinning of the head, “propeller\_rotator”: Float value indicating speed of spinning of the tail/propeller which spins in the opposite direction of the head\_rotator to together cause added forward motion, “hargo\_hatch”: boolean value where 1 means the hatch is open and the cargo of thrombin protein that causes blood to clot around the tumour thereby starving it, is released, “self\_destruct”: boolean value which when 1 cause the bot to self-destruct and deal damage to the cancer cell, “diffuser”: Boolean value when 1 indicates that the bot has been turned-off/broken down etc, i.e. diffused safely meaning the body can now discard it naturally., “beacon”P: Boolean value that when 1 means that this primary bot’s beacon signal emitter is active. }
* Ready to DecideP: Boolean value which marks whether team as a whole is ready to make a decision based on diagnosis.
* Content Store: Maps content names to data.
* Pending Interests Table: Maps content name to list of interested peer names.
* Forwarding Information Base: Maps content name to peer it was forwarded to.
* Neighbors: Maps peer name to peer host, port, and marker.

**Threading**

Each nanobot has 3 threads associated with it: Main thread, Connection Thread and an Event Thread.

Connection Thread: Listens for incoming connections on the socket of this bot.

Event Thread: The even thread keeps listening for the following.

1. Availability of a diagnosis.
2. Whether bot is tethered to a cell indicating having detected a possibly cancerous tumour (Primary Bot Only).
3. Presence of a beacon signal from a primary bot (Non Primary Bot Only).
4. Connection having become stale (if this bot was connected to a team and it has received no communication from this team in the last set x amount of time, then connection is considered stale, and state of the bot shall be reset).

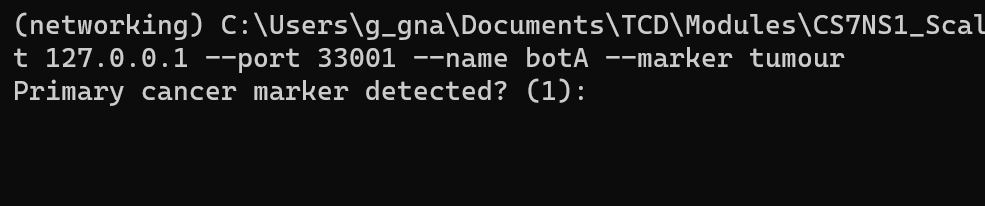
**Program Flow**

Control flow throughout the program can be understood when divided into 4 main phases.

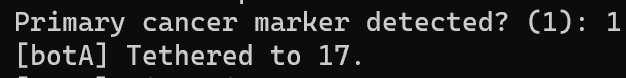
1. Tether Phase

Initially **all bots are moving through the blood** upon creation.

**The P bots listens/searches for a primary tumour marker**. This search is simulated through requesting user input such that an input of 1 indicated tumour detected while anything else indicates nothing detected. If nothing is detected for too long (here, if user inputs something other than 1 for a set x no. of trials) then this bot diffuses as its presence in the body is deemed unnecessary.



Meanwhile, **NP bots** **listen for a beacon signal** produced by a p bot indicating presence of a possibly cancerous structure that requires their attention.

When a P bot detects a cancer marker, it updates its content store with this latest measurement against content name “marker/tumour”. It then tethers to the spot by slowing down the head and propeller rotator actuators and extending its tethers actuator. Now, this bot’s position shall be fixed, and it activates its beacon actuator which sends an interest packet of the form “<node-id>/beacon/on” to the rendezvous server.

1. **Neighbour Discovery Phase**
2. **Diagnosis Phase**
3. **Response Phase**

# References

Afanasyev, A., Burke, J., Refaei, Wang, T., Zhang, L., Zhang, B., & Lixia. (2018). A Brief Introduction to Named Data Networking. *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*.

Bilal, M., & Pack, S. (2020). Secure Distribution of Protected Content in Information-Centric Networking. *IEEE Systems Journal, 14*(2), 1921-1932. doi:10.1109/jsyst.2019.2931813

Datatracker. (n.d.). *Charter for Research Group* . Retrieved from datatracker.ietf.org: https://datatracker.ietf.org/rg/icnrg/about/

Fu, X., Kutscher, D., Misra, S., & Li, R. (2018). Information-Centric Networking Security. *IEEE Communications Magazine, 56*(11), 60-61. doi:10.1109/mcom.2018.8539022

IITM, NPTEL-NOC. (2023, May 12). *Information Centric Networking - Part 1*. Retrieved from YouTube: https://www.youtube.com/watch?v=F\_NpB6trbzo&list=PLyqSpQzTE6M9voigbAaFzKiybgN7Xgvka&index=64

NPTEL-NOC IITM. (2023, May 12). *Information Centric Networking - Part 2*. Retrieved from YouTube: https://www.youtube.com/watch?v=aVZw1Qd6RLo&list=PLyqSpQzTE6M9voigbAaFzKiybgN7Xgvka&index=65

NPTEL-NOC IITM. (2023, May 12). *Named Data Networking - Part 1*. Retrieved from YouTube: https://www.youtube.com/watch?v=OM-cBILswZk&list=PLyqSpQzTE6M9voigbAaFzKiybgN7Xgvka&index=67

NPTEL-NOC IITM. (2023, May 12). *Named Data Networking - Part 2*. Retrieved from YouTube: https://www.youtube.com/watch?v=Eh5yHPeqBI0&list=PLyqSpQzTE6M9voigbAaFzKiybgN7Xgvka&index=69

Scenes, B. B. (2023, January 17). *What is Digital Signature Algorithm ?* Retrieved from medium.com: https://medium.com/@21\_000\_000/digital-signature-algorithm-60c8318cf9b6

SG NDN Telkom University. (2021, October 2). *NDN vs IP*. Retrieved from YouTube: https://www.youtube.com/watch?v=ud9YOlekH6c