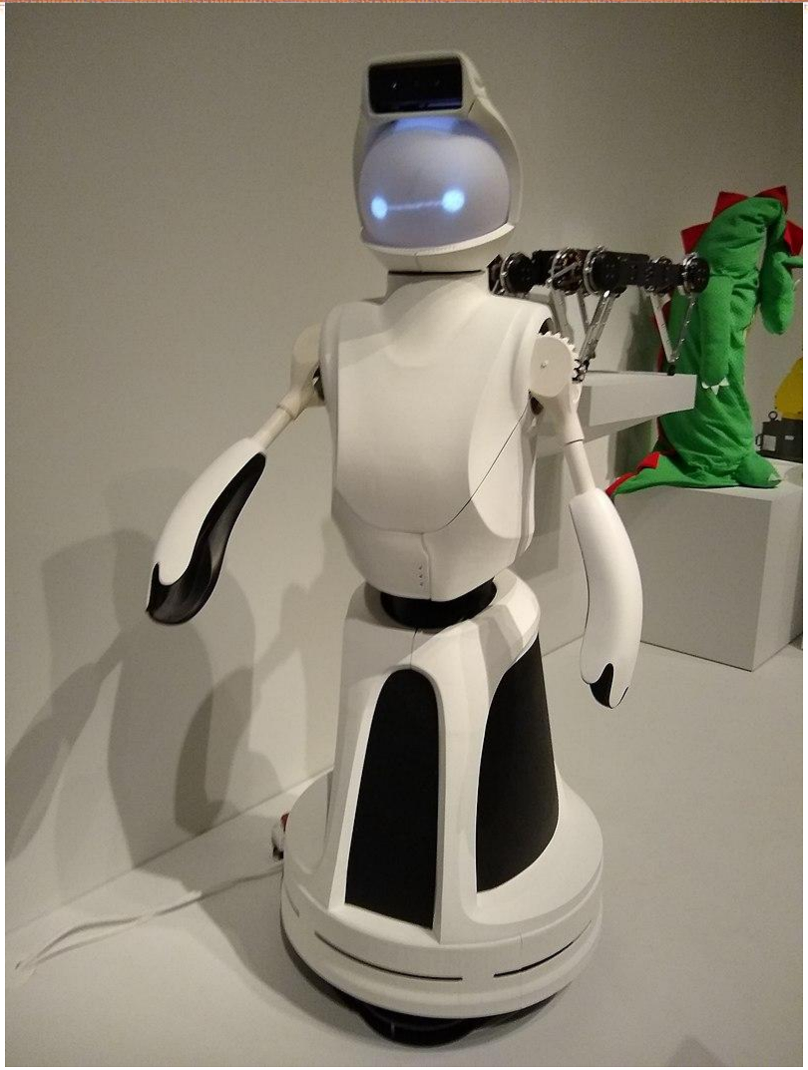
Decentralized Information Flow Control for ROS2

Nishit V Pandya* (*UC San Diego)*
Himanshu Kumar *(Indian Institute of Science)*
Gokulnath Pillai (*Indian Institute of Science)*
Vinod Ganapathy (*Indian Institute of Science)*

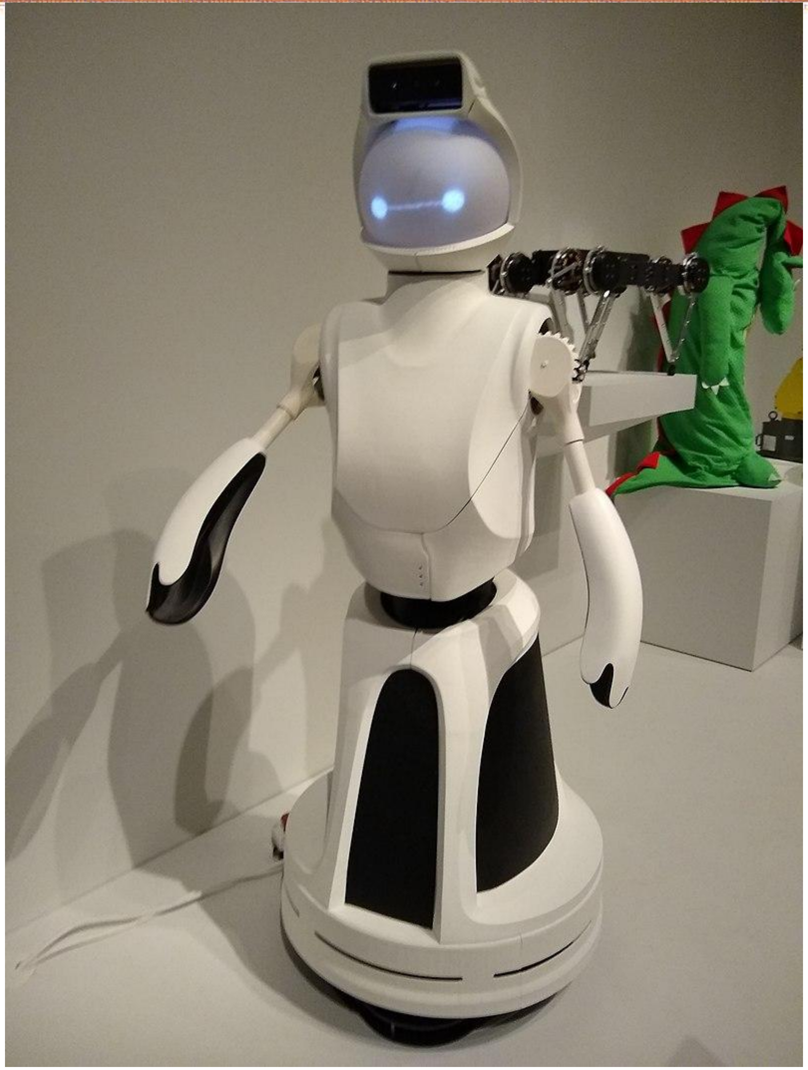
INDIAN INSTITUTE OF SCIENCE
भारतीय विज्ञान संस्थान

** - work done while at IISc*



Personal Assistance Robots

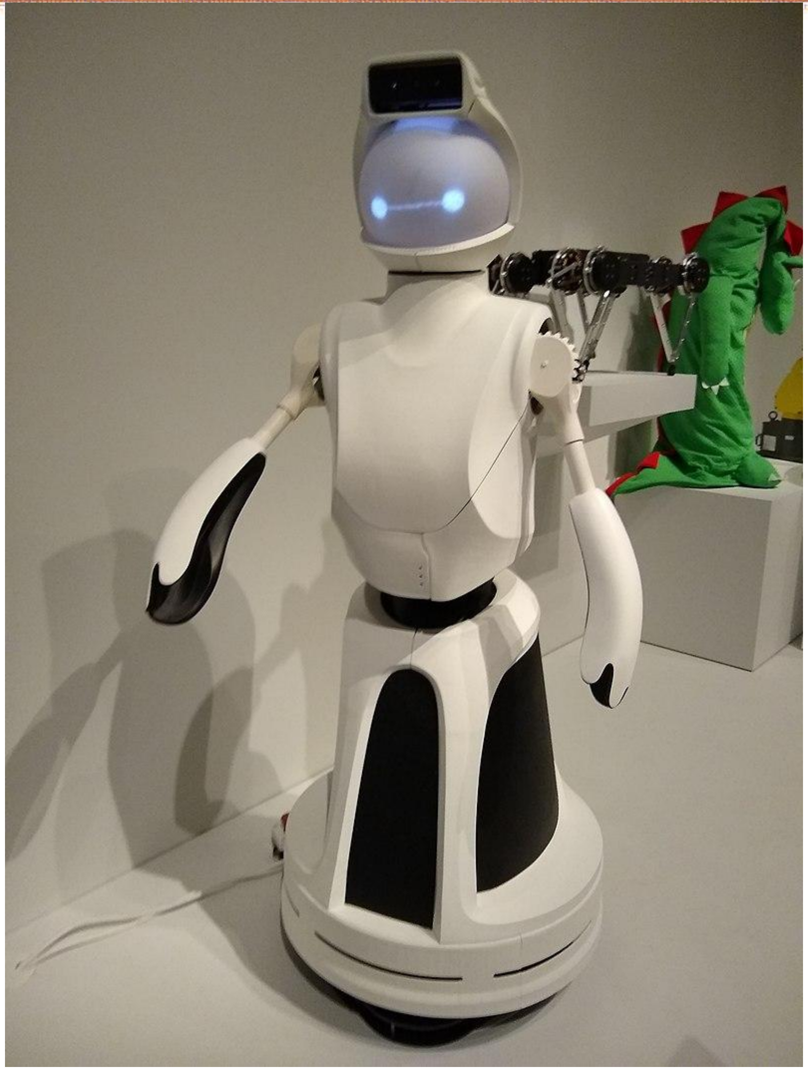
Fleets of Warehouse Robots
(Image Credit: User:Geni, CC-BY-SA 4.0)



Personal Assistance Robots
(Image Credit: Mary Mark Ockerbloom, CC-BY-SA 4.0)

Fleets of Warehouse Robots
(Image Credit: User:Geni, CC-BY-SA 4.0)

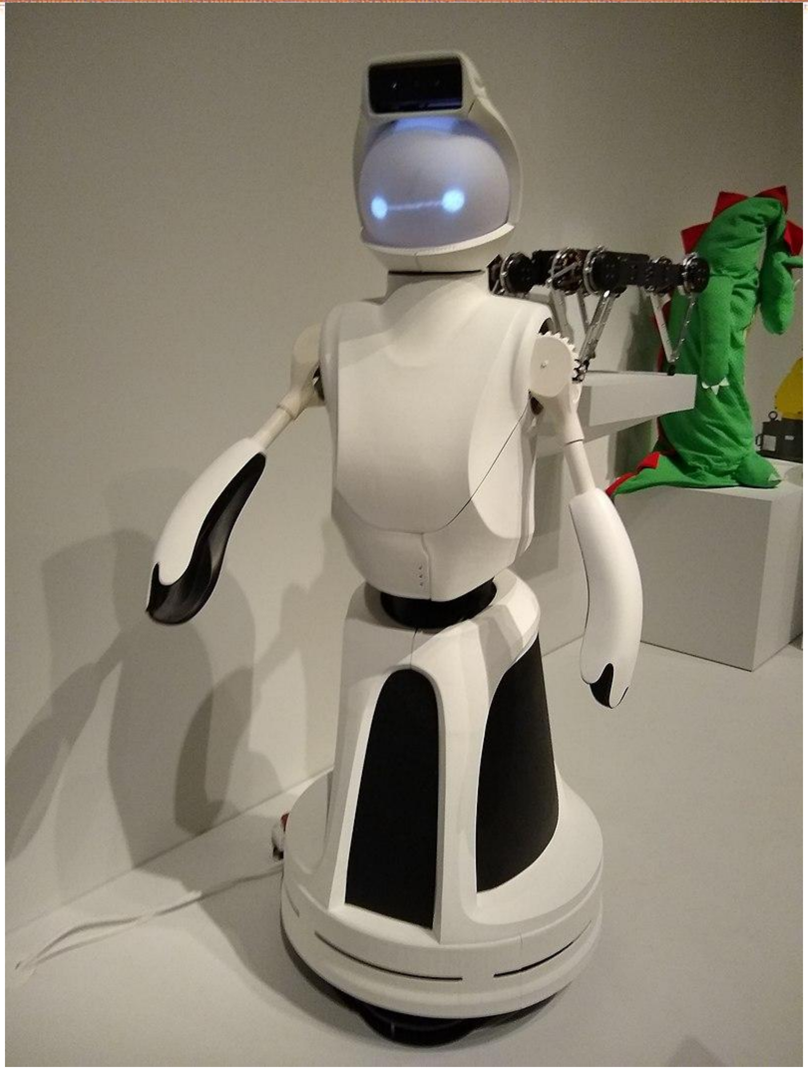
Military Robots
(Image Credit: Timothy Sandland)



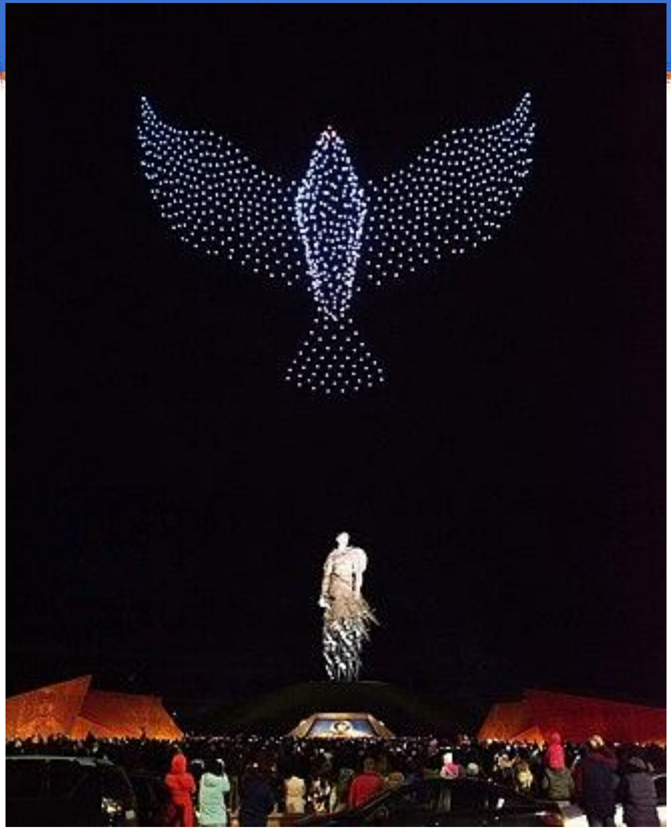
Personal Assistance Robots
(Image Credit: Mary Mark Ockerbloom, CC-BY-SA 4.0)

Fleets of Warehouse Robots
(Image Credit: User:Geni, CC-BY-SA 4.0)

Military Robots
(Image Credit: Timothy Sandland)



Personal Assistance Robots
(Image Credit: Mary Mark Ockerbloom, CC-BY-SA 4.0)



Swarm Drones
(Image Credit:Geoscan Group, CC-BY-SA 4.0)

Source: https://github.com/vmayoral/ros-robotics-companies

Robot Operating System 2

Robot Operating System 2

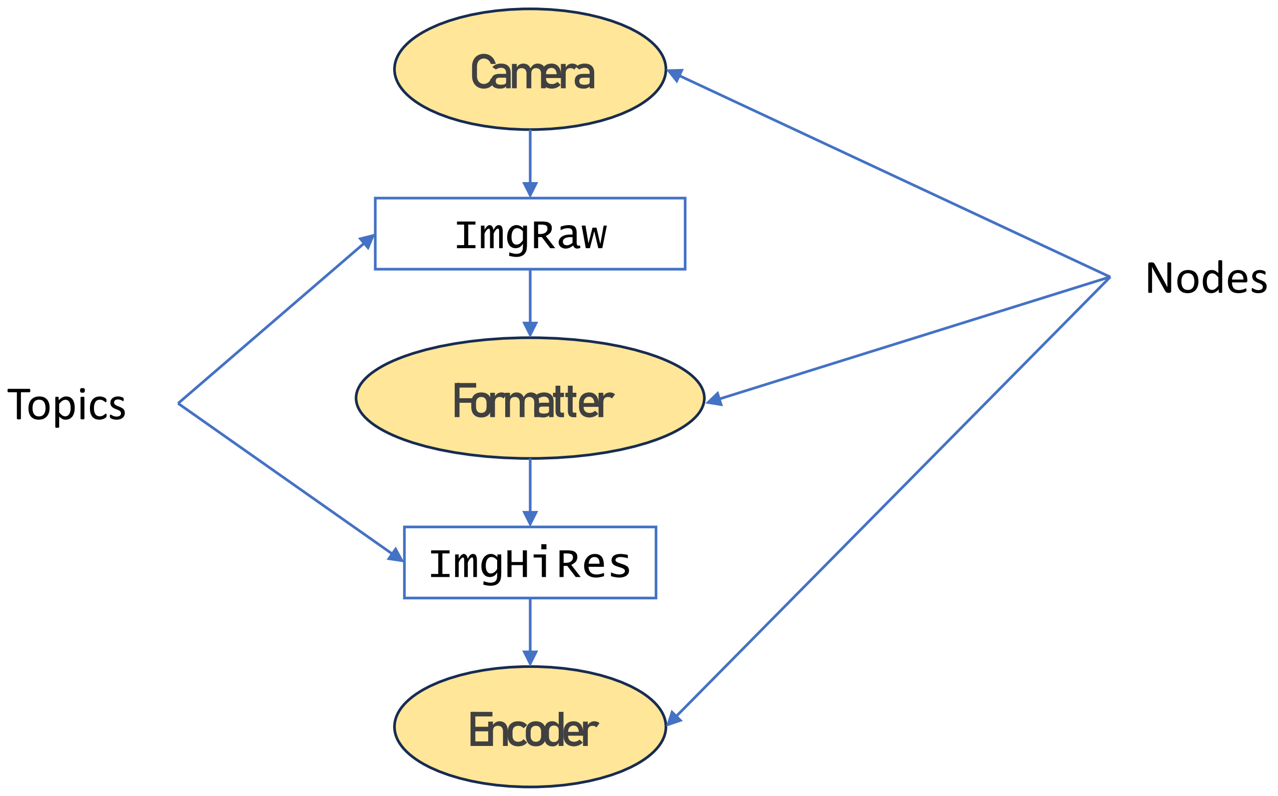
- User space libraries that facilitate communication between apps.

Robot Operating System 2

- User space libraries that facilitate communication between apps.
- “Nodes” publish and subscribe to “topics”

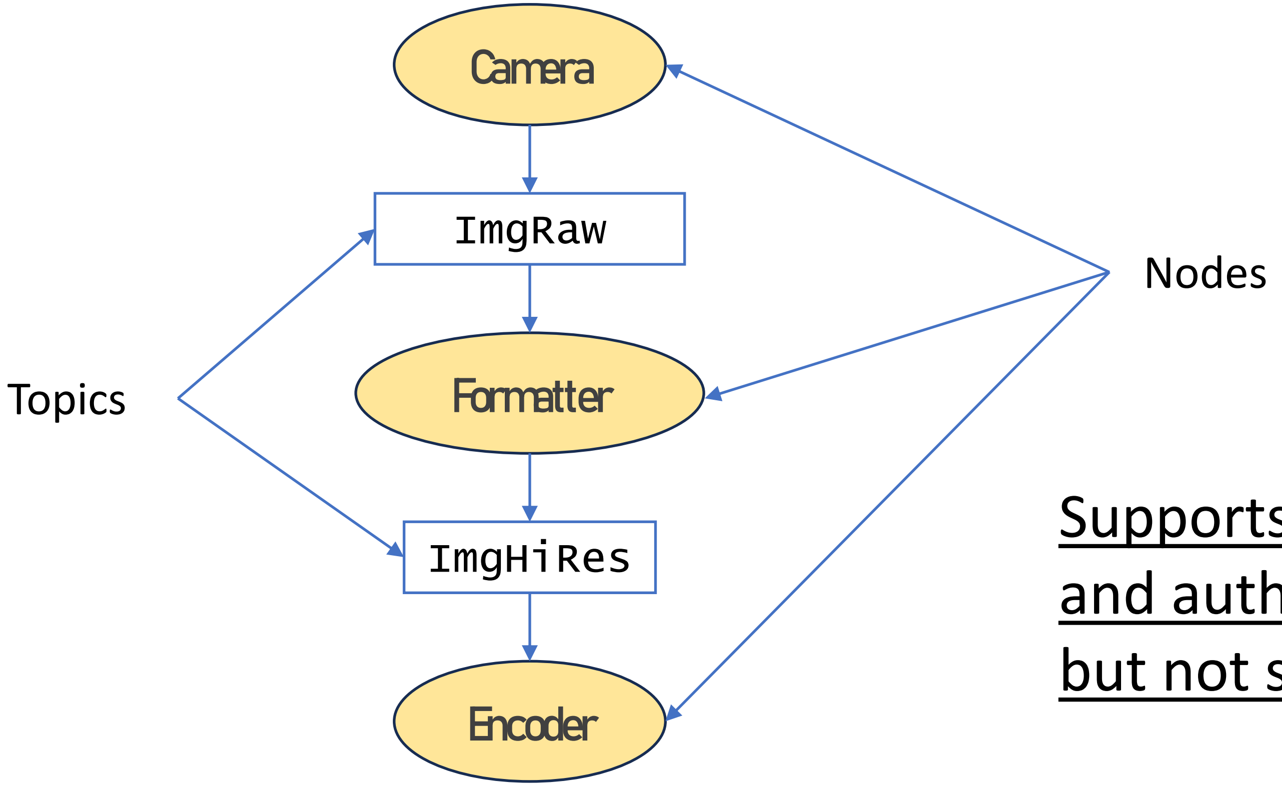
Robot Operating System 2

- User space libraries that facilitate communication between apps.
- “Nodes” publish and subscribe to “topics”



Robot Operating System 2

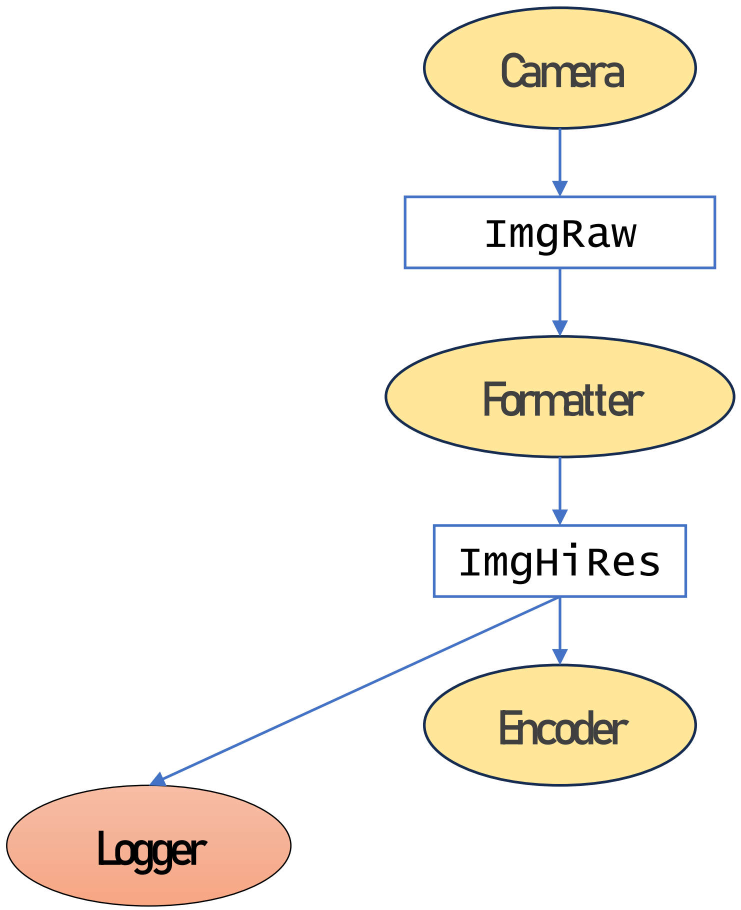
- User space libraries that facilitate communication between apps.
- “Nodes” publish and subscribe to “topics”



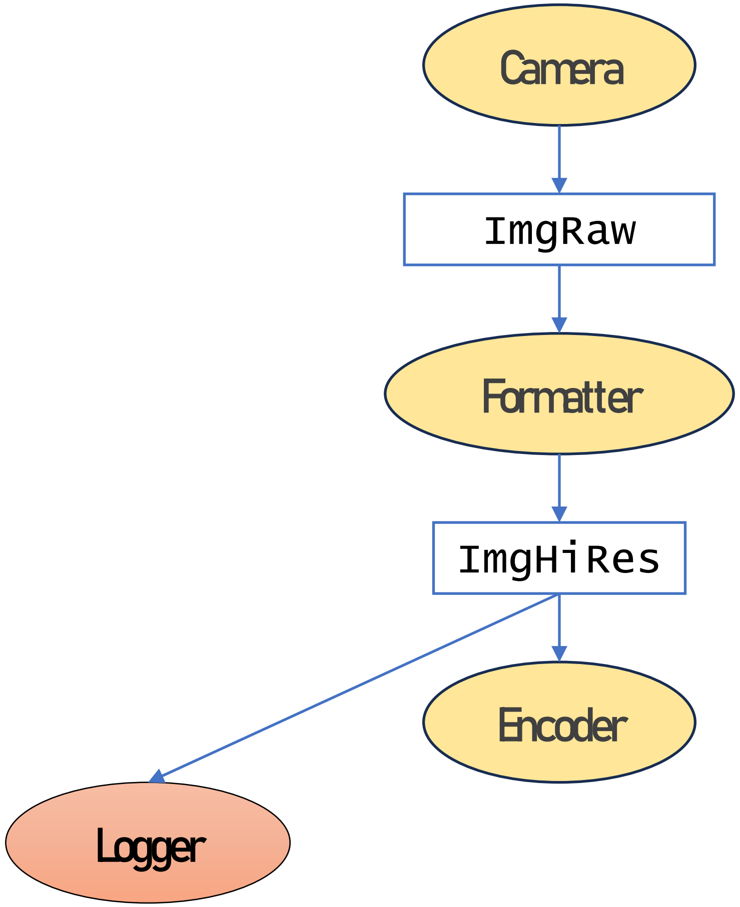
<u>Supports encryption and authentication, but not sufficient</u>

Problem of Downstream Control

Problem of Downstream Control



Problem of Downstream Control



Camera can’t control what happens to its data beyond Formatter!

Decentralized Information Flow Control Systems

Decentralized Information Flow Control Systems

- A DIFC System[1] tracks path of data using sets of tags attached to messages and nodes

[1] - A Decentralized Model for Information Flow Control" by Andrew C. Myers and Barbara Liskov. In ACM Symposium on Operating Systems Principles (SOSP), (Saint Malo, France), Oct. 1997, pp. 129-142

Decentralized Information Flow Control Systems

- A DIFC System[1] tracks path of data using sets of tags attached to messages and nodes
- It enforces the following –

[1] - A Decentralized Model for Information Flow Control" by Andrew C. Myers and Barbara Liskov. In ACM Symposium on Operating Systems Principles (SOSP), (Saint Malo, France), Oct. 1997, pp. 129-142

Decentralized Information Flow Control Systems

- A DIFC System[1] tracks path of data using sets of tags attached to messages and nodes
- It enforces the following –
  1. A node can read a message only if it has all the tags carried by that message

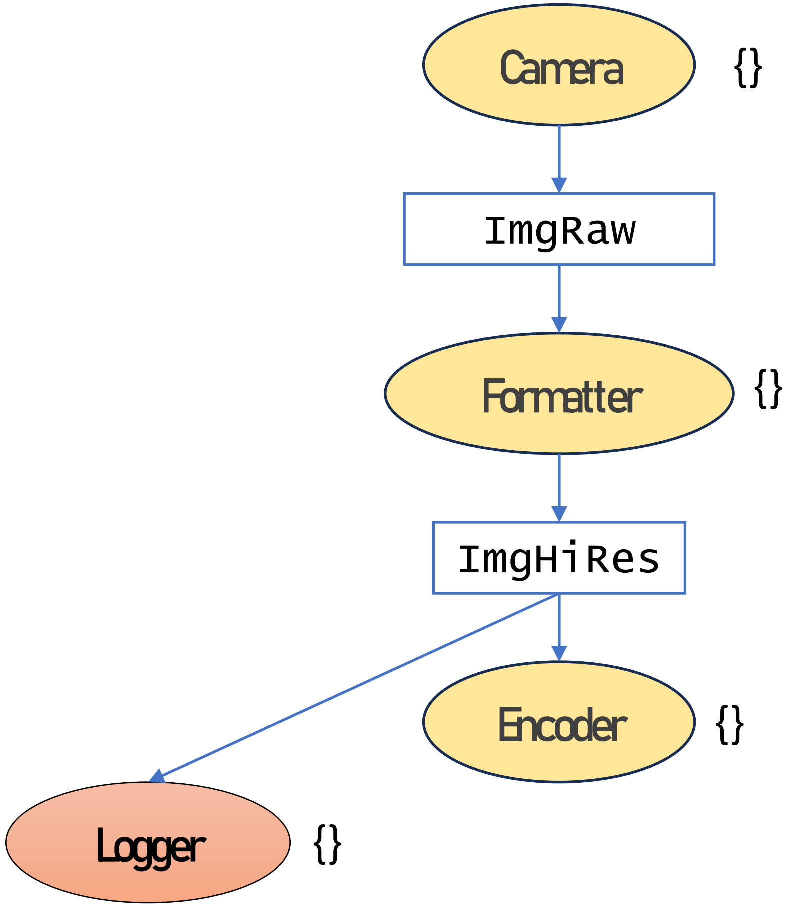
[1] - A Decentralized Model for Information Flow Control" by Andrew C. Myers and Barbara Liskov. In ACM Symposium on Operating Systems Principles (SOSP), (Saint Malo, France), Oct. 1997, pp. 129-142

Decentralized Information Flow Control Systems

- A DIFC System[1] tracks path of data using sets of tags attached to messages and nodes
- It enforces the following –
  1. A node can read a message only if it has all the tags carried by that message
  2. Outgoing messages from any node inherit the tags of its sender

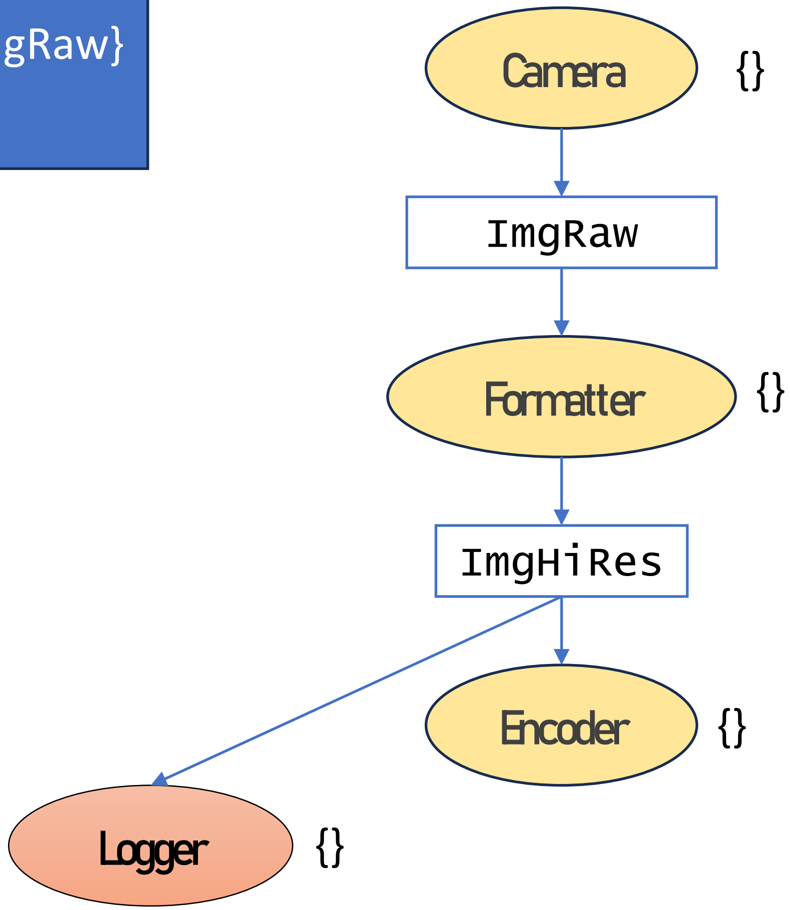
[1] - A Decentralized Model for Information Flow Control" by Andrew C. Myers and Barbara Liskov. In ACM Symposium on Operating Systems Principles (SOSP), (Saint Malo, France), Oct. 1997, pp. 129-142

Decentralized Information Flow Control



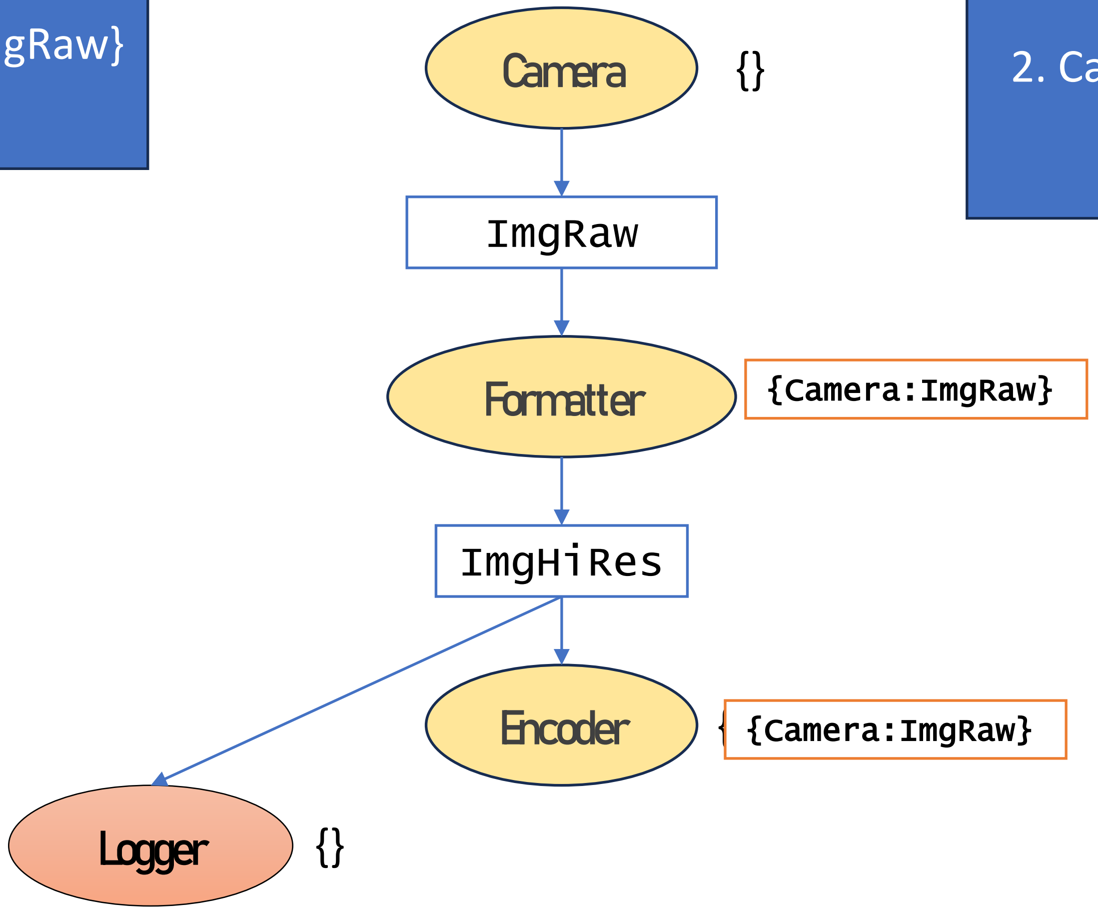
Decentralized Information Flow Control

1. Camera adds tag T = {Camera:ImgRaw} to its outgoing messages



Decentralized Information Flow Control

1. Camera adds tag T = {Camera:ImgRaw} to its outgoing messages



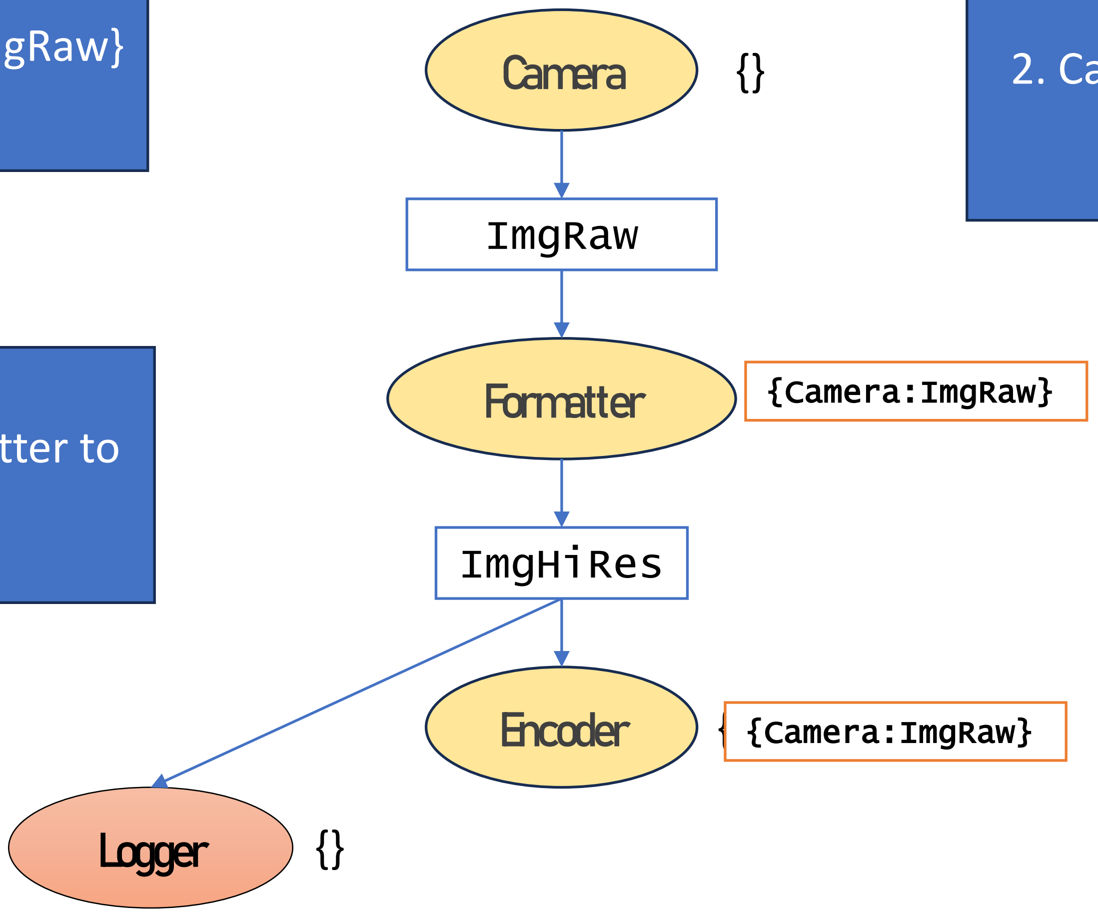
2. Camera grants tag T to Formatter & Encoder (but not Logger)

Decentralized Information Flow Control

1. Camera adds tag T = {Camera:ImgRaw} to its outgoing messages

2. Camera grants tag T to Formatter & Encoder (but not Logger)

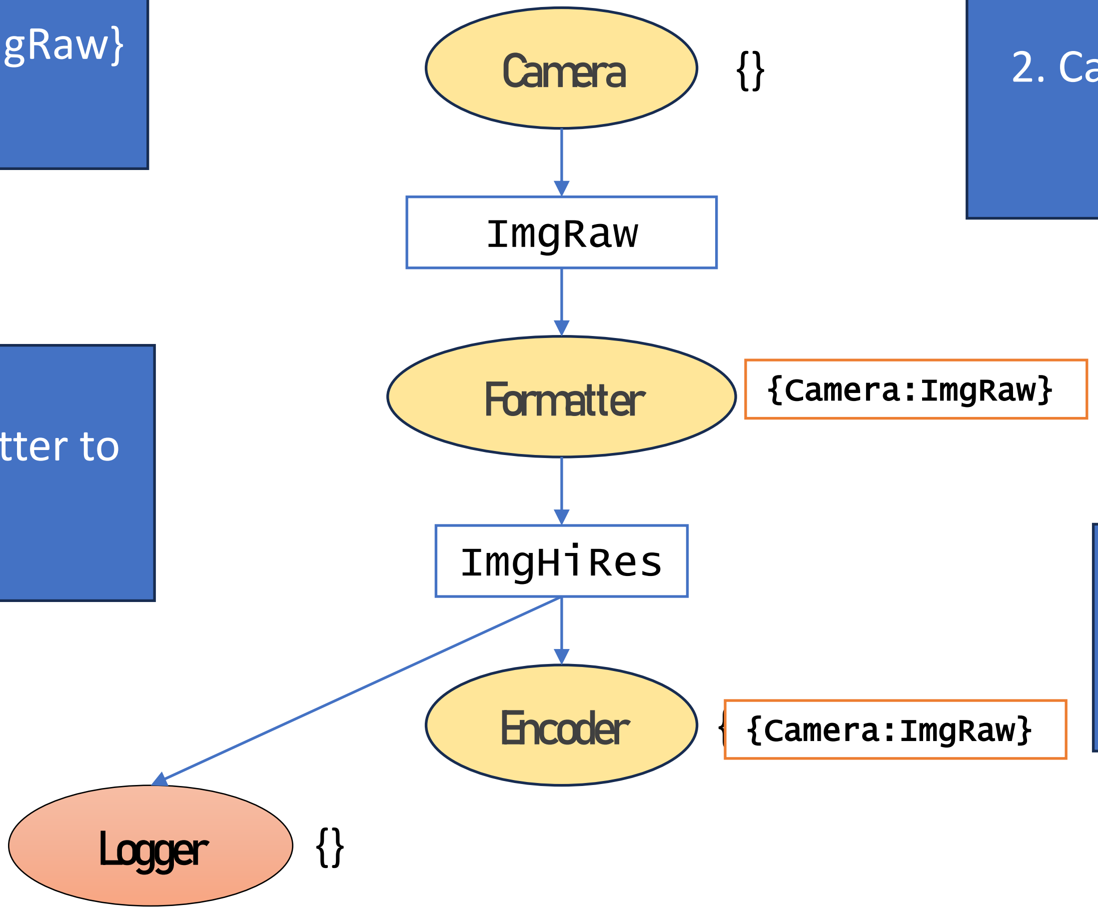
3. The DIFC system allows Formatter to read the message



Decentralized Information Flow Control

1. Camera adds tag T = {Camera:ImgRaw} to its outgoing messages

2. Camera grants tag T to Formatter & Encoder (but not Logger)



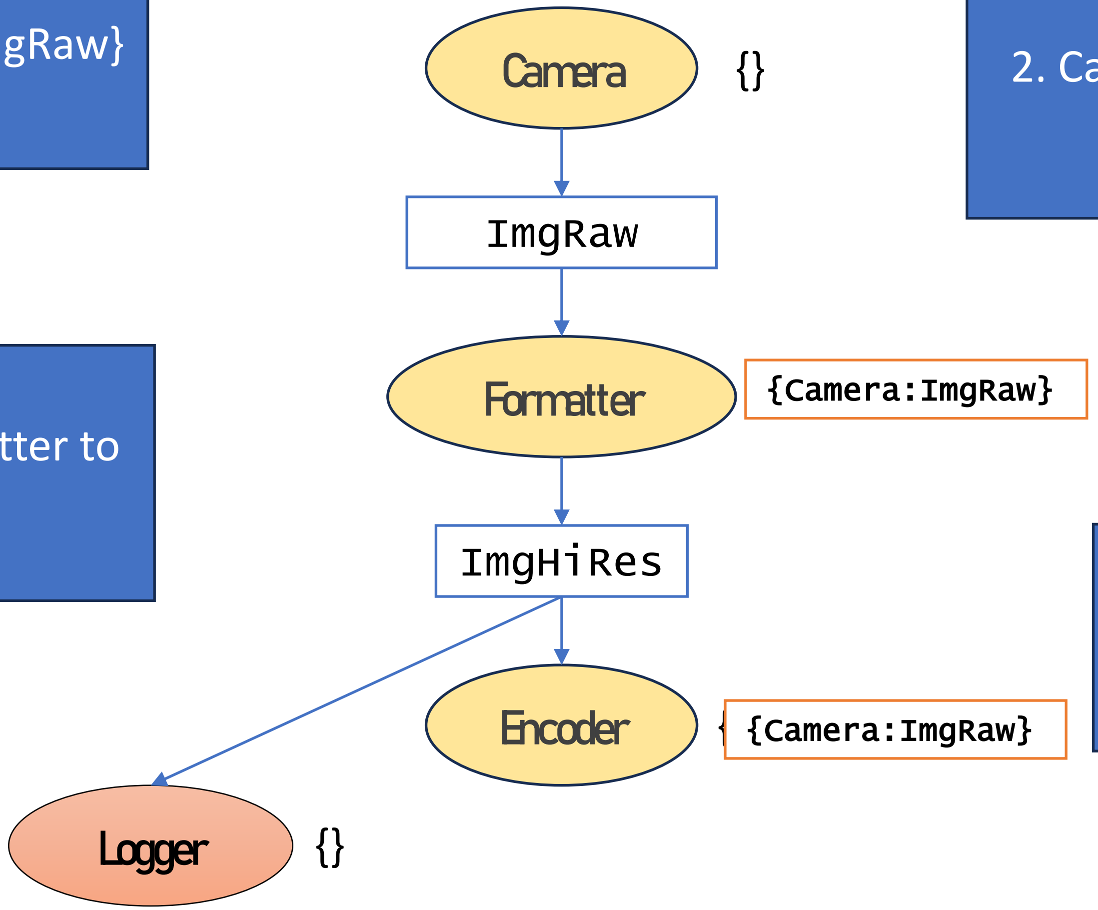
3. The DIFC system allows Formatter to read the message

4. The DIFC system ensures that all outgoing messages from Formatter now inherit tag T

Decentralized Information Flow Control

1. Camera adds tag T = {Camera:ImgRaw} to its outgoing messages

2. Camera grants tag T to Formatter & Encoder (but not Logger)

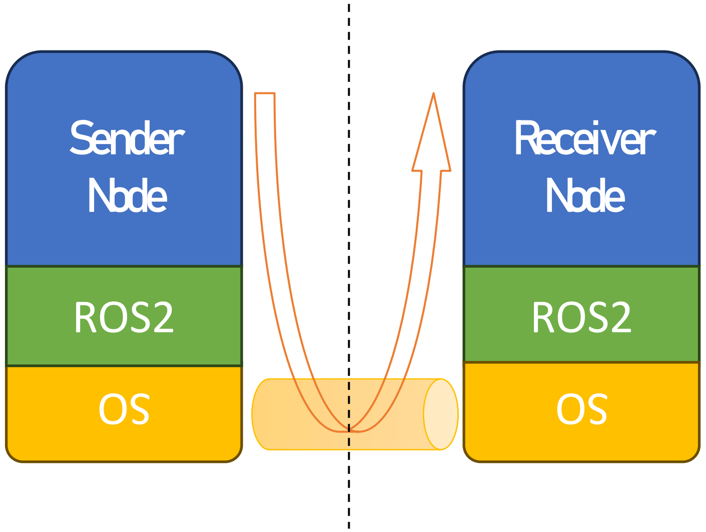


3. The DIFC system allows Formatter to read the message

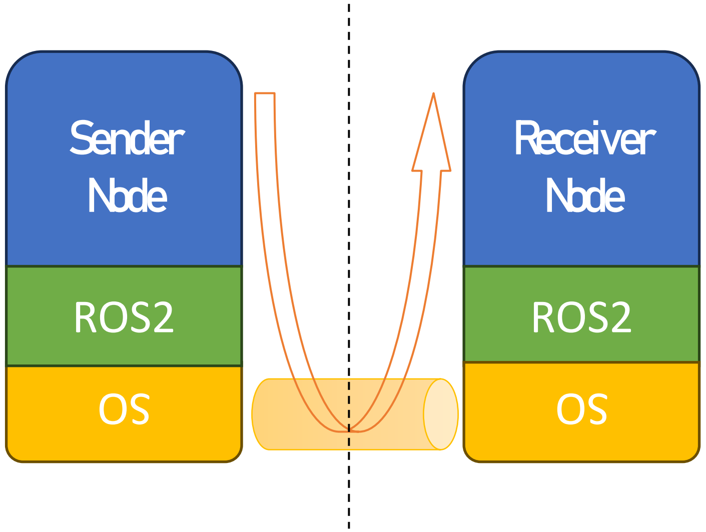
4. The DIFC system ensures that all outgoing messages from Formatter now inherit tag T

5. The DIFC system stops Logger from reading the messages

Challenges to enforcing DIFC

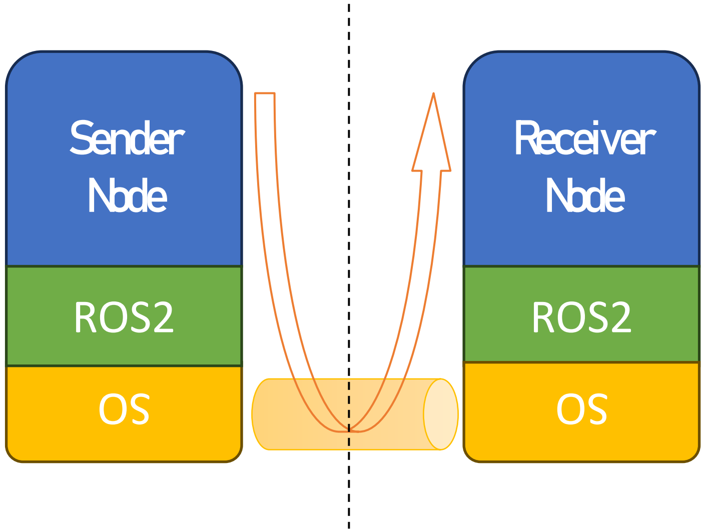


Challenges to enforcing DIFC



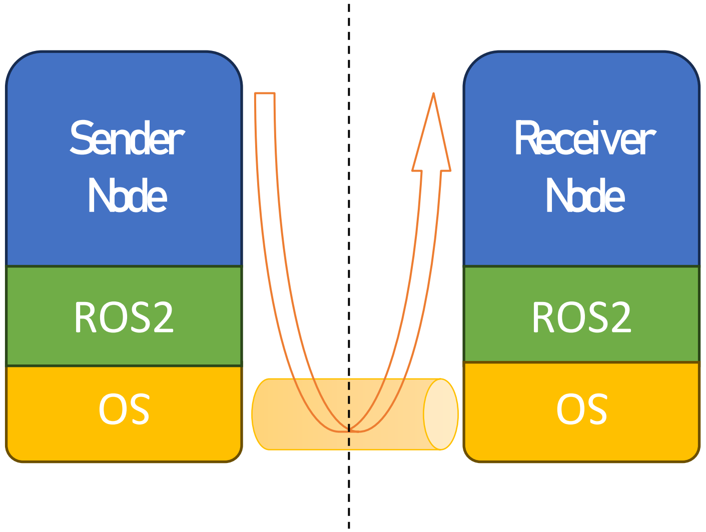
- ROS2 is *distributed* and *decentralized*

Challenges to enforcing DIFC



- ROS2 is *distributed* and *decentralized*
- Receiver side filtering in ROS2 too late!

Challenges to enforcing DIFC



- ROS2 is *distributed* and *decentralized*
- Receiver side filtering in ROS2 too late!
- Sender side filtering in ROS2 requires trusted, centralized, state management

Our Key Insight

DIFC on top of a decentralized ABE cryptosystem[1] addresses these challenges

[1] Lewko, A., Waters, B. (2011). Decentralizing Attribute-Based Encryption. In: Paterson, K.G. (eds) Advances in Cryptology – EUROCRYPT 2011. EUROCRYPT 2011

Attribute-Based Encryption

Public Key Encryption

Attribute-Based Encryption

Attribute-Based Encryption

Public Key Encryption

- Encrypt for a specific *user* A

Attribute-Based Encryption

Attribute-Based Encryption

Public Key Encryption

- Encrypt for a specific *user* A

Attribute-Based Encryption

- Encrypt for *attributes* “Doctor” and “On Duty”

Attribute-Based Encryption

Public Key Encryption

- Encrypt for a specific *user* A
- Encrypt with public key corresponding to the user A

Attribute-Based Encryption

- Encrypt for *attributes* “Doctor” and “On Duty”

Attribute-Based Encryption

Public Key Encryption

- Encrypt for a specific *user* A
- Encrypt with public key corresponding to the user A

Attribute-Based Encryption

- Encrypt for *attributes* “Doctor” and “On Duty”
- Encrypt with both the public keys corresponding to the attributes

Attribute-Based Encryption

Public Key Encryption

- Encrypt for a specific *user* A
- Encrypt with public key corresponding to the user A
- Only the user A can decrypt

Attribute-Based Encryption

- Encrypt for *attributes* “Doctor” and “On Duty”
- Encrypt with both the public keys corresponding to the attributes

Attribute-Based Encryption

Public Key Encryption

- Encrypt for a specific *user* A
- Encrypt with public key corresponding to the user A
- Only the user A can decrypt

Attribute-Based Encryption

- Encrypt for *attributes* “Doctor” and “On Duty”
- Encrypt with both the public keys corresponding to the attributes
- Only those who are both “Doctor” and “On Duty” can decrypt

DIFC using ABE

DIFC using ABE

- “Add DIFC tags” => Encrypting with more *attributes*

DIFC using ABE

- “Add DIFC tags” => Encrypting with more *attributes*
- “Giving read access” => Handing out *decryption keys*

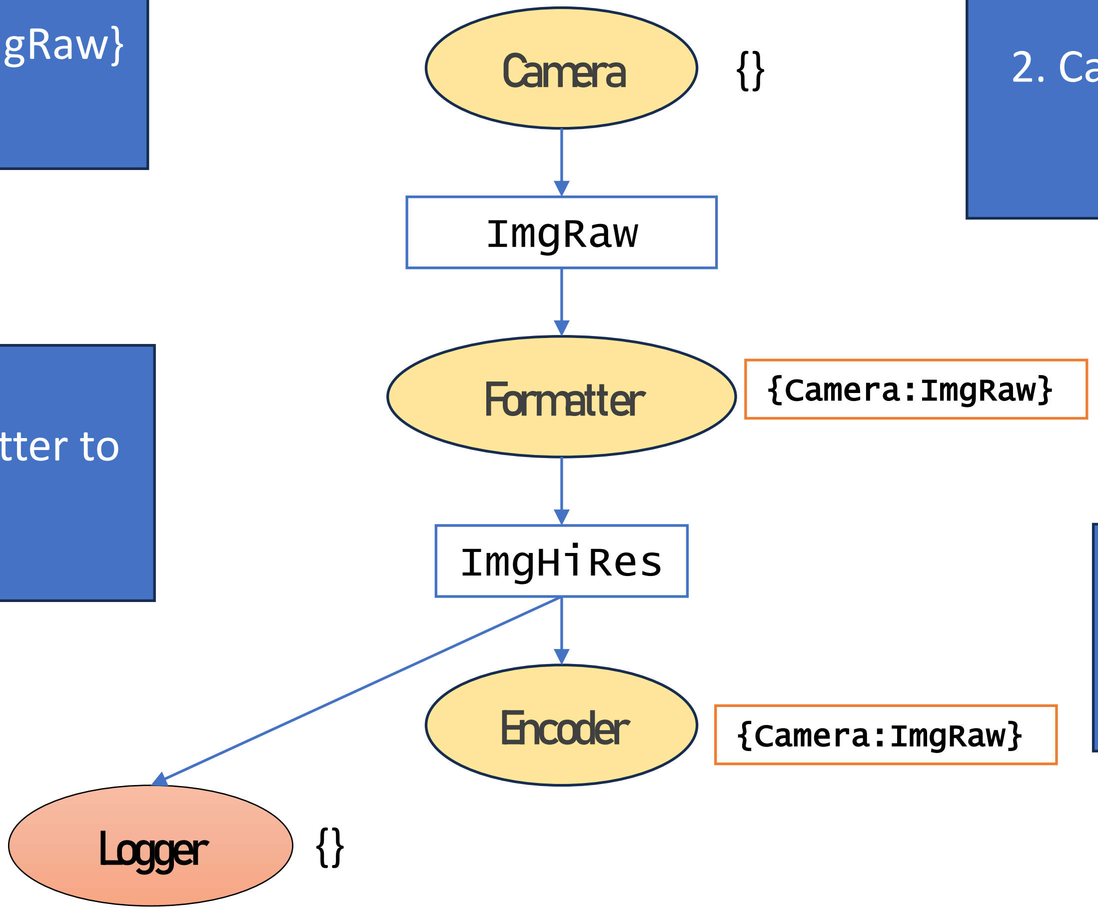
DIFC using ABE

1. Camera adds tag T = {Camera:ImgRaw} to its outgoing messages

2. Camera grants tag T to Formatter & Encoder (but not Logger)

3. The DIFC system allows Formatter to read the message

4. The DIFC system ensures that all outgoing messages from Formatter now inherit tag T



5. The DIFC system stops Logger from reading the messages

DIFC using ABE

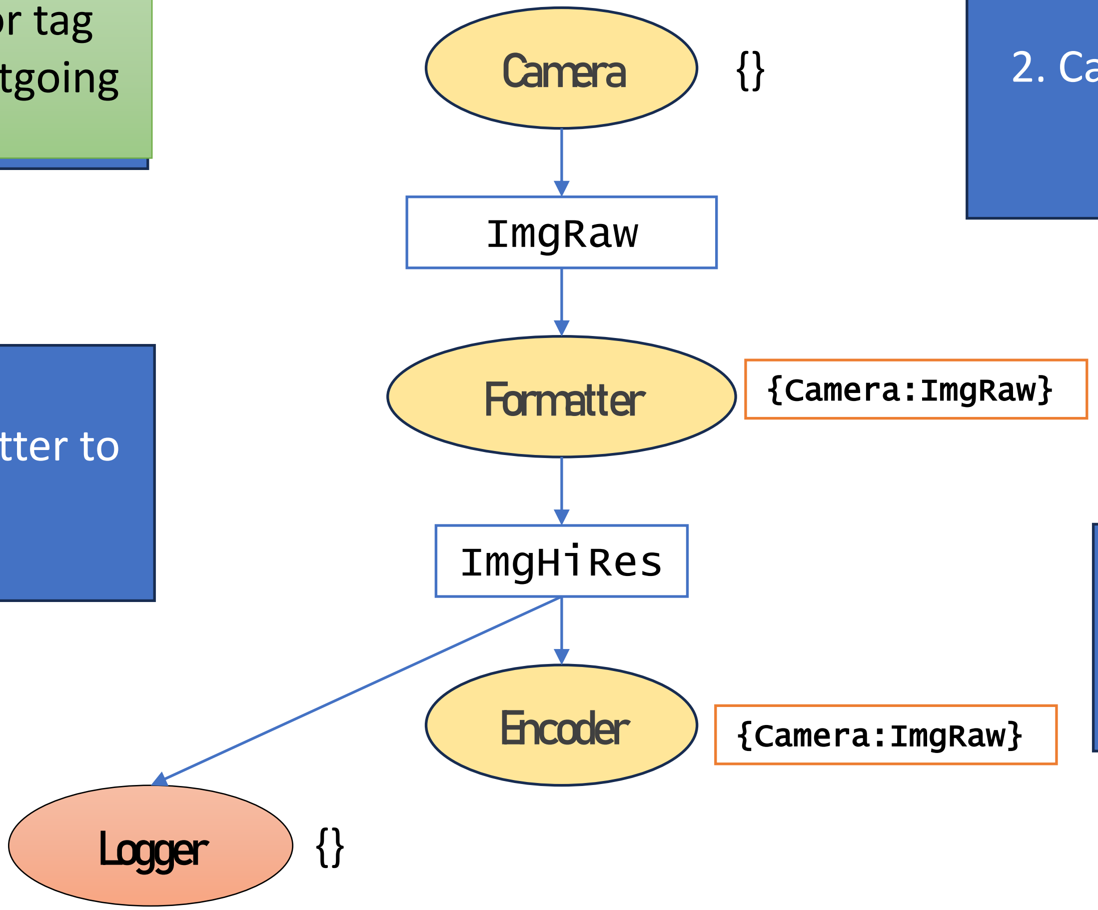
1. Camera generates public key for tag {Camera:ImgRaw} and encrypts outgoing messages

2. Camera grants tag T to Formatter & Encoder (but not Logger)

3. The DIFC system allows Formatter to read the message

4. The DIFC system ensures that all outgoing messages from Formatter now inherit tag T

5. The DIFC system stops Logger from reading the messages



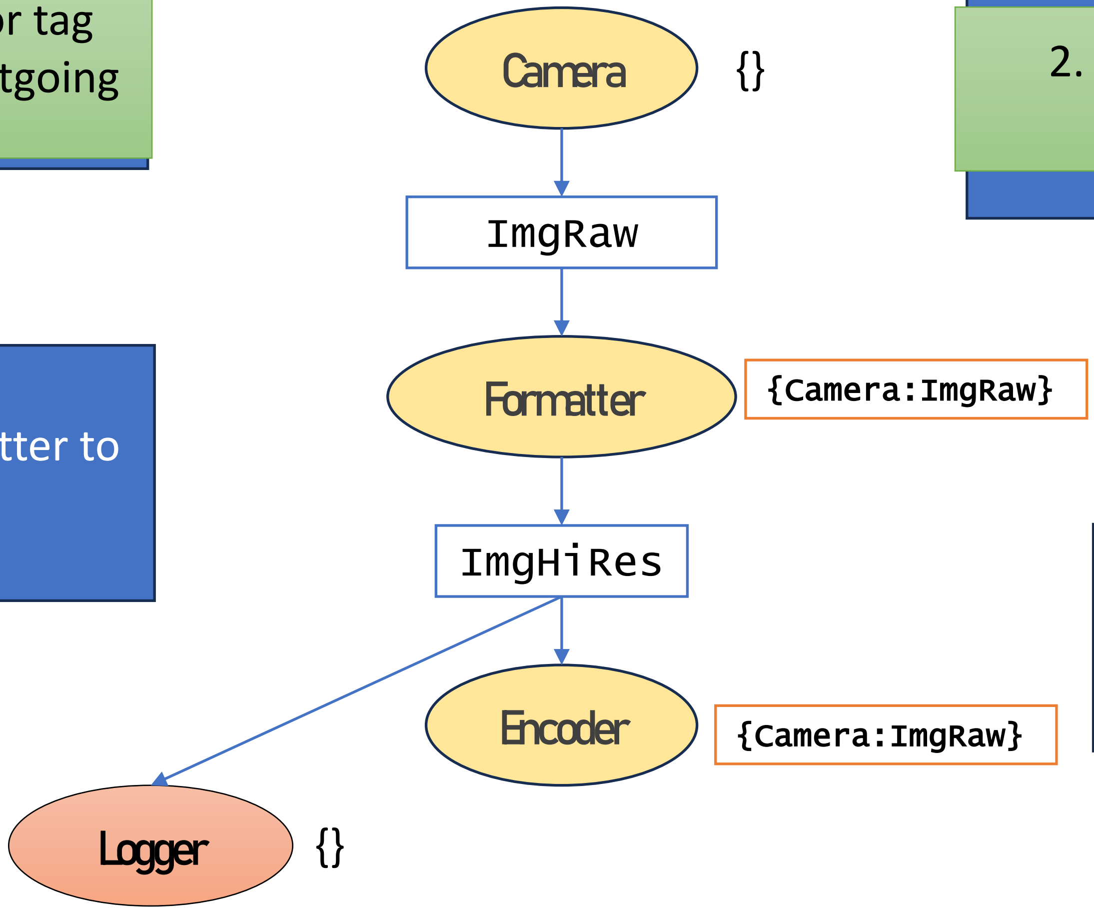
DIFC using ABE

1. Camera generates public key for tag {Camera:ImgRaw} and encrypts outgoing messages

2. Camera gives decryption keys to Formatter and Encoder

3. The DIFC system allows Formatter to read the message

4. The DIFC system ensures that all outgoing messages from Formatter now inherit tag T

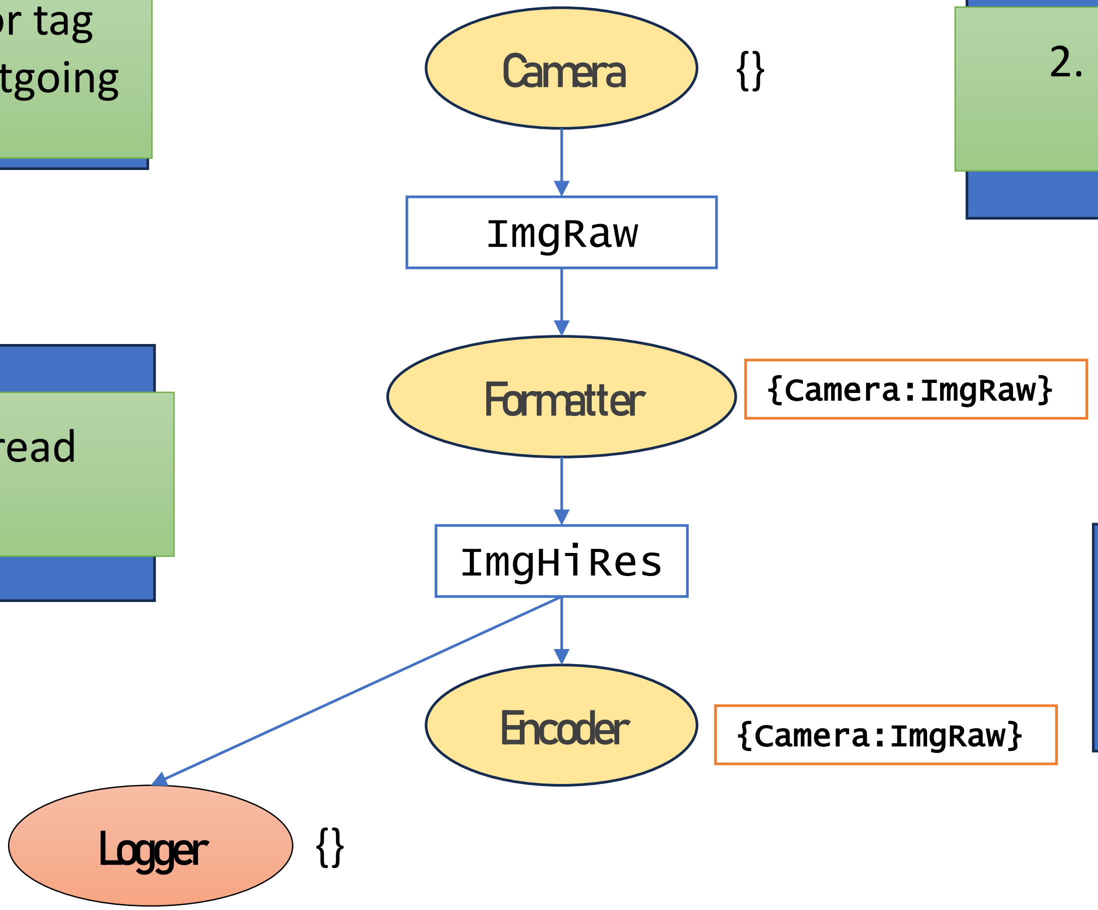


5. The DIFC system stops Logger from reading the messages

DIFC using ABE

1. Camera generates public key for tag {Camera:ImgRaw} and encrypts outgoing messages

2. Camera gives decryption keys to Formatter and Encoder



3. Formatter can decrypt and read messages

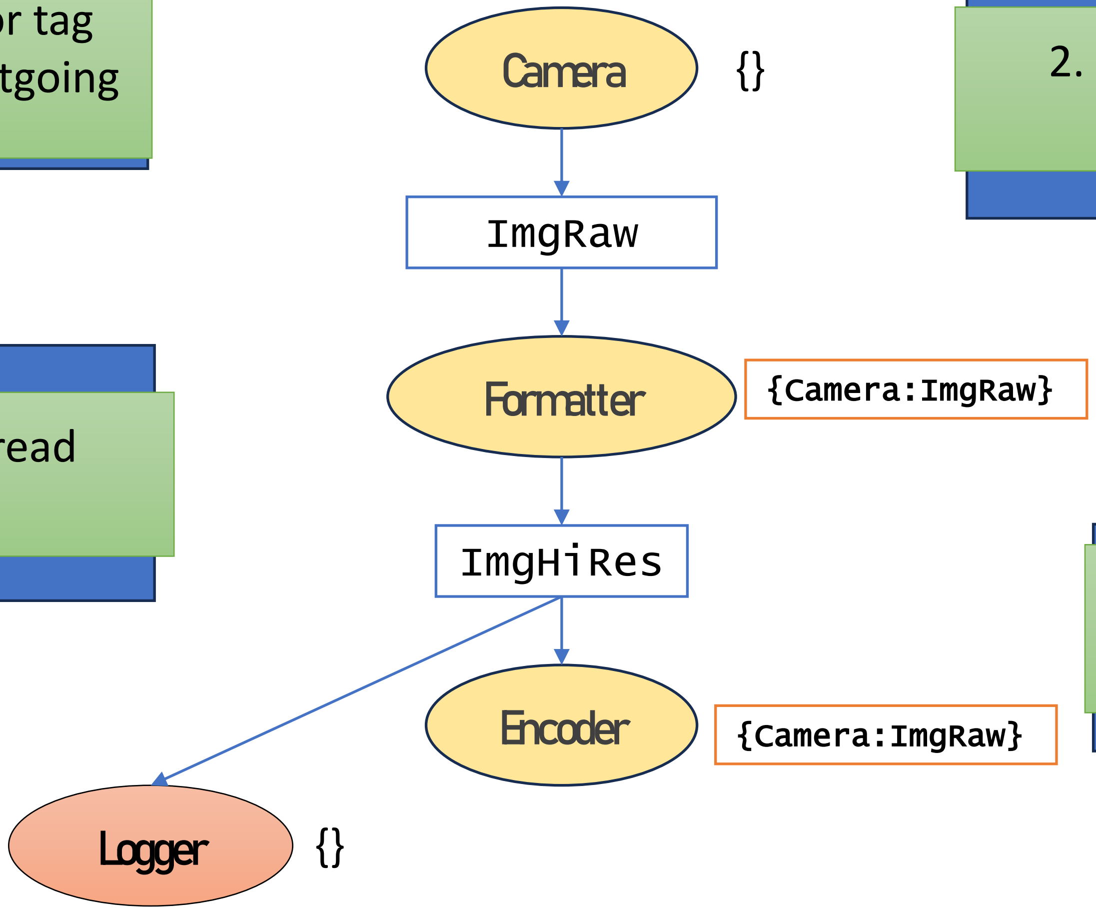
4. The DIFC system ensures that all outgoing messages from Formatter now inherit tag T

5. The DIFC system stops Logger from reading the messages

DIFC using ABE

1. Camera generates public key for tag {Camera:ImgRaw} and encrypts outgoing messages

2. Camera gives decryption keys to Formatter and Encoder



3. Formatter can decrypt and read messages

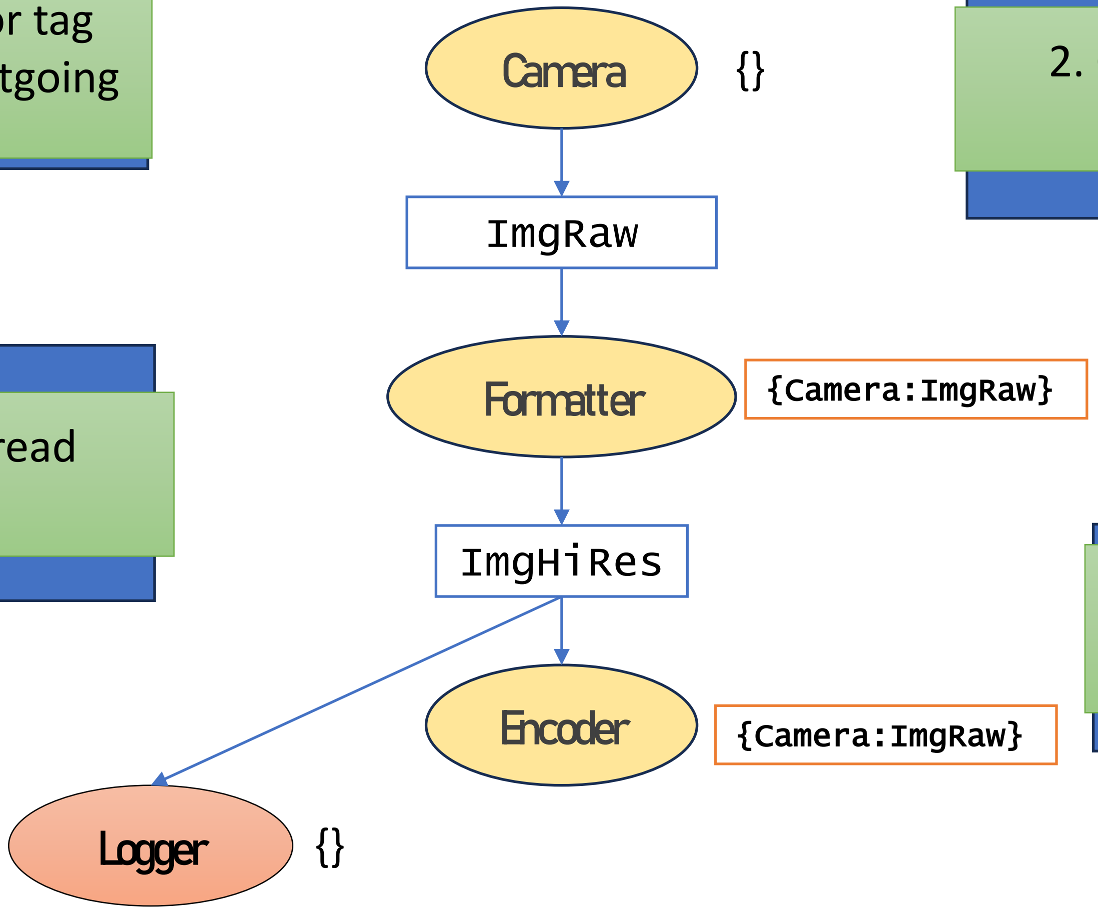
4. System ensures that outgoing messages are also encrypted with the public key

5. The DIFC system stops Logger from reading the messages

DIFC using ABE

1. Camera generates public key for tag {Camera:ImgRaw} and encrypts outgoing messages

2. Camera gives decryption keys to Formatter and Encoder



3. Formatter can decrypt and read messages

4. System ensures that outgoing messages are also encrypted with the public key

5. Logger can't decrypt messages

DIFC using ABE

DIFC using ABE

- Start with the decentralized scheme from [1]

DIFC using ABE

- Start with the decentralized scheme from [1]

[1] Lewko, A., Waters, B. (2011). Decentralizing Attribute-Based Encryption. In: Paterson, K.G. (eds) Advances in Cryptology – EUROCRYPT 2011. EUROCRYPT 2011

DIFC using ABE

- Start with the decentralized scheme from [1]
- Combine with existing ROS2 security primitives

[1] Lewko, A., Waters, B. (2011). Decentralizing Attribute-Based Encryption. In: Paterson, K.G. (eds) Advances in Cryptology – EUROCRYPT 2011. EUROCRYPT 2011

DIFC using ABE

- Start with the decentralized scheme from [1]
- Combine with existing ROS2 security primitives
- Add some OS protection

[1] Lewko, A., Waters, B. (2011). Decentralizing Attribute-Based Encryption. In: Paterson, K.G. (eds) Advances in Cryptology – EUROCRYPT 2011. EUROCRYPT 2011

DIFC using ABE

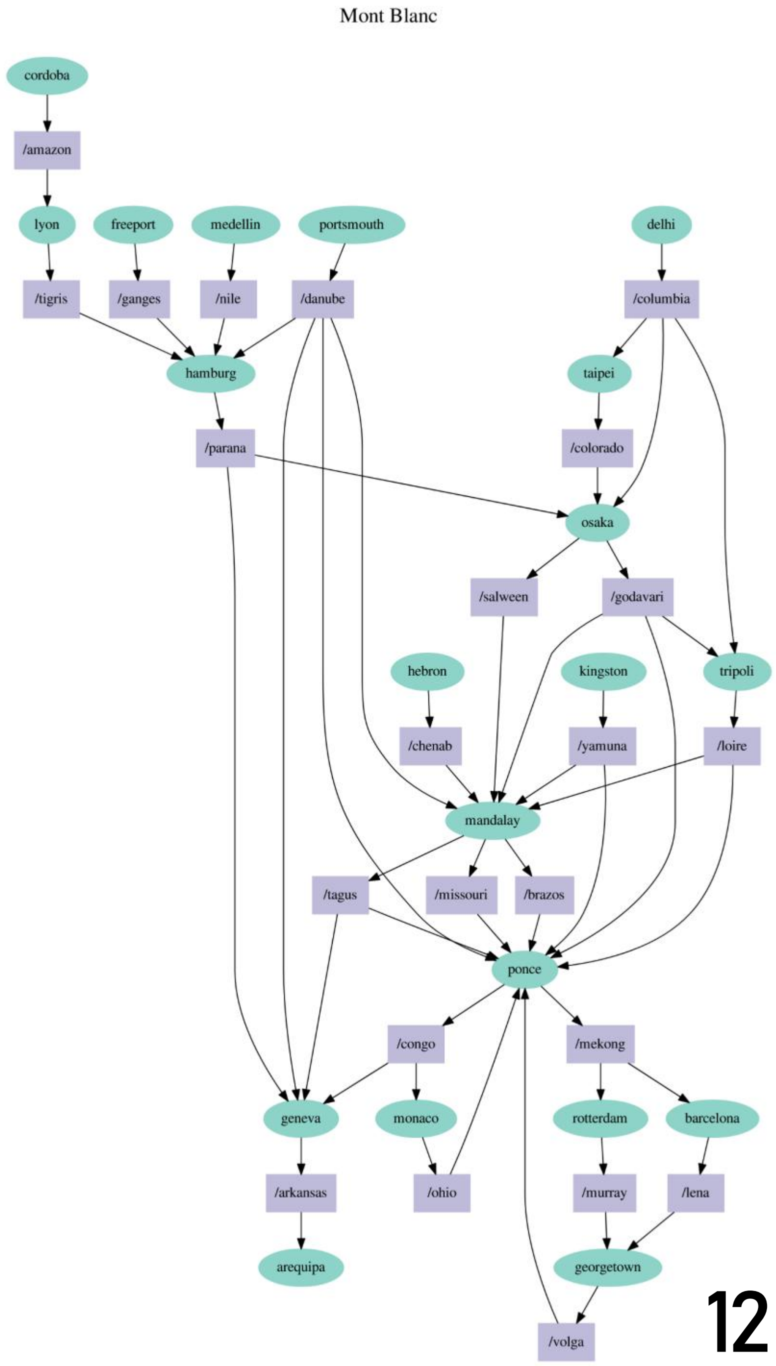
- Start with the decentralized scheme from [1]
- Combine with existing ROS2 security primitives
- Add some OS protection

Allows for decentralized enforcement of DIFC
Works in a distributed setting
No expensive setup phase requiring global coordination
Allows incremental deployment

[1] Lewko, A., Waters, B. (2011). Decentralizing Attribute-Based Encryption. In: Paterson, K.G. (eds) Advances in Cryptology – EUROCRYPT 2011. EUROCRYPT 2011

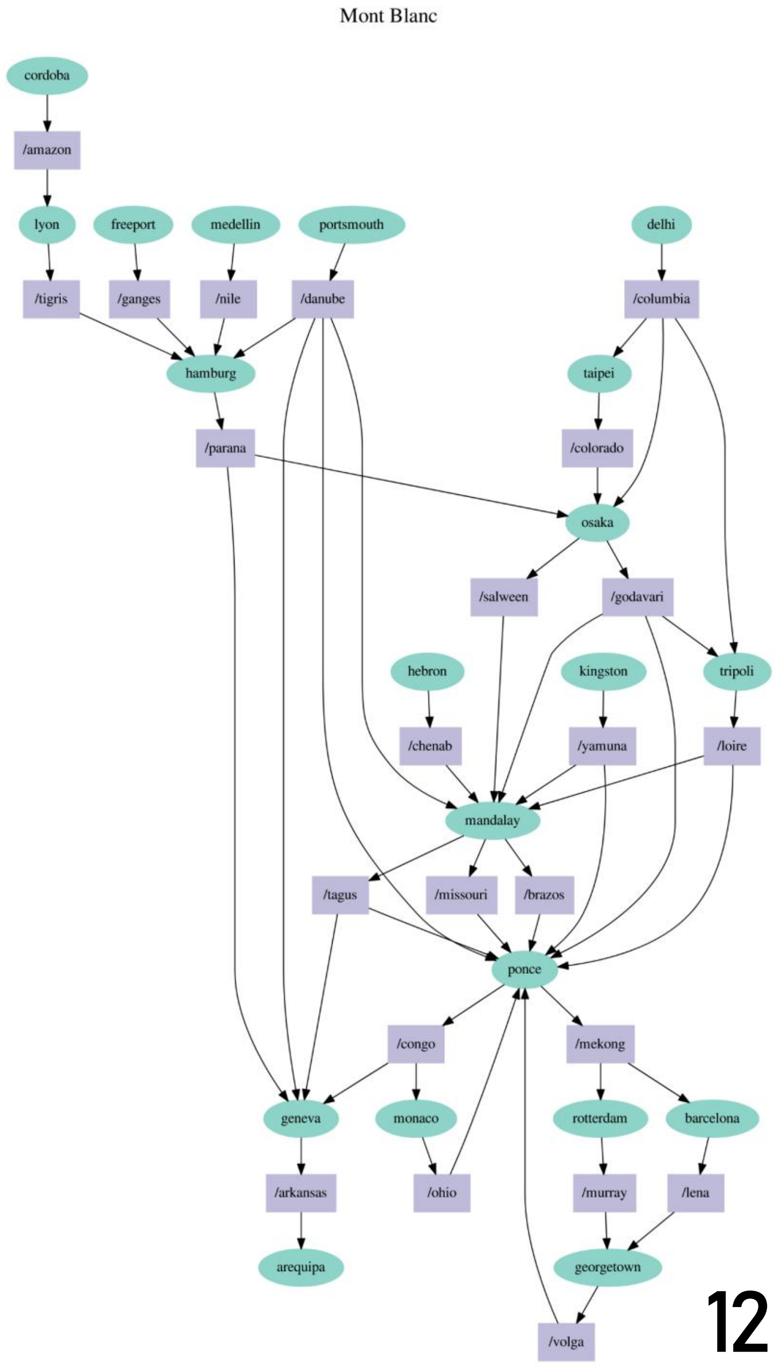
Performance – iRobot benchmarks

Performance – iRobot benchmarks



Performance – iRobot benchmarks

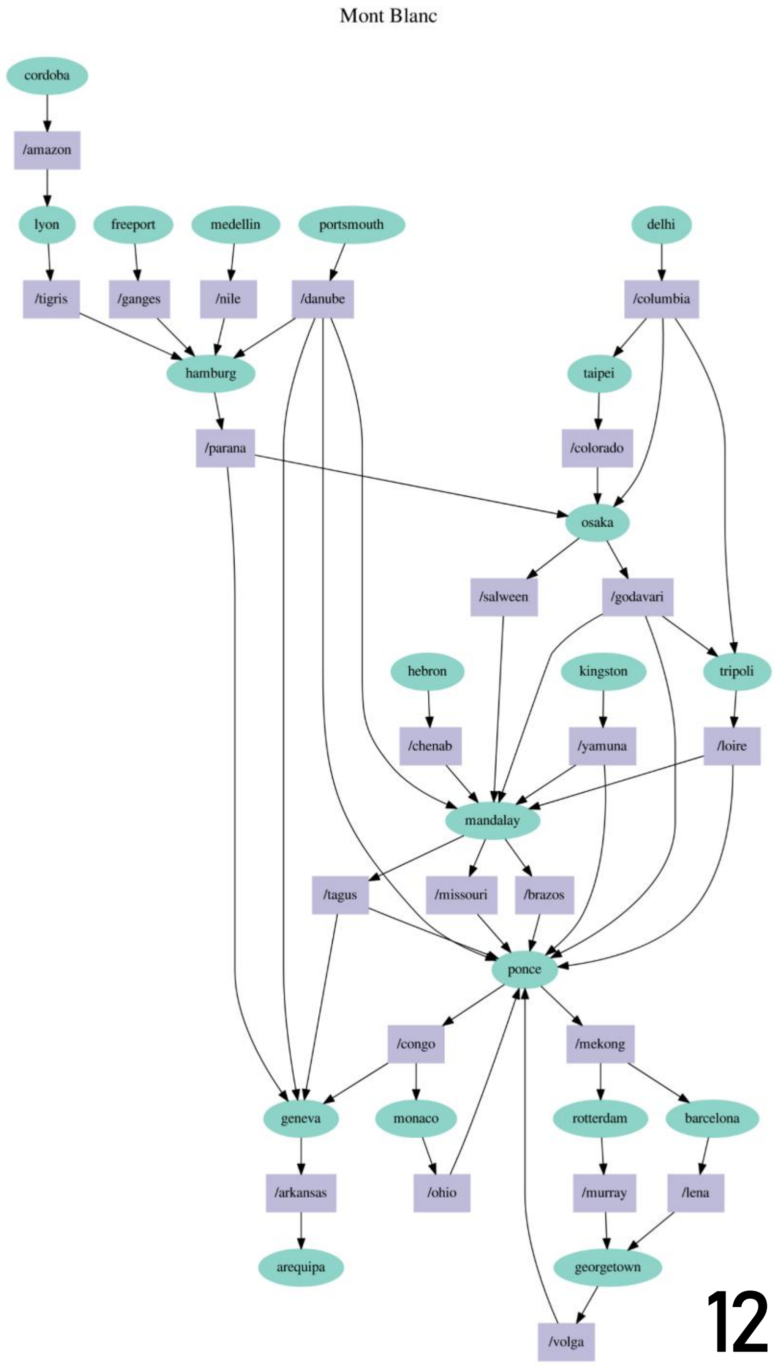
Topology	Path Length	SROS2 (ms)	Picaros (ms)
Cedar	3	0.85	10.4
Sierra Nevada	3	0.94	13.6
Mont Blanc - 1	5	1.34	61.3
Mont Blanc - 4	5	1.34	115.0
Mont Blanc - 7	5	1.34	316.9



Performance – iRobot benchmarks

Topology	Path Length	SROS2 (ms)	Picaros (ms)
Cedar	3	0.85	10.4
Sierra Nevada	3	0.94	13.6
Mont Blanc - 1	5	1.34	61.3
Mont Blanc - 4	5	1.34	115.0
Mont Blanc - 7	5	1.34	316.9

Topology	Memory Usage (MB)		Power Draw (mW)	
	SROS2	Picaros	SROS2	Picaros
Cedar	1690.1	2525.1	4896.7	5437.0
Sierra Nevada	2163.1	2442.5	4881.0	5393.0
Mont Blanc - 1	2529.3	4019.6	5056.1	5281.8
Mont Blanc - 4	2529.3	4068.2	5056.1	5295.7
Mont Blanc - 7	2529.3	4096.8	5056.1	5307.7



Performance – iRobot benchmarks

Performance – iRobot benchmarks

- ABE encryption and decryption are expensive operations.

Performance – iRobot benchmarks

- ABE encryption and decryption are expensive operations.
- Modular exponentiation takes up most time.

Performance – iRobot benchmarks

- ABE encryption and decryption are expensive operations.
- Modular exponentiation takes up most time.
- Since every node decrypts, computes, then encrypts again, latency grows significantly for longer paths.

Performance – iRobot benchmarks

- ABE encryption and decryption are expensive operations.
- Modular exponentiation takes up most time.
- Since every node decrypts, computes, then encrypts again, latency grows significantly for longer paths.
- More implementational optimizations might help.

Summary

Summary

- We address the problem of downstream control for ROS2 applications.

Summary

- We address the problem of downstream control for ROS2 applications.
- We cast the problem of DIFC into the framework of ABE

Summary

- We address the problem of downstream control for ROS2 applications.
- We cast the problem of DIFC into the framework of Decentralized ABE
- ABE based design allows for decentralized, distributed, dynamic enforcement which fits in line with ROS2 philosophy.

ROS2 implementational challenges

- The OS is unaware of ROS2 abstractions
- All messages between two nodes, irrespective of publisher and topic get sent via same port
- Thus, fine grained labelling not directly possible in the OS.

The ABE API

- `AuthSetup(Attribute) → (PrivKey, PubKey):` Every user wanting to add a DIFC tag to a message generates a public, private key pair and releases the public key
- `Encrypt(Message, {PubKey}) → Ciphertext:` Encryption of a message happens with respect to all the tags the message has to carry.
- `KeyGen(UserID, Attribute, PrivKey) → DecKey:` User Specific decryption keys for every attribute
- `Decrypt({DecKey}, Ciphertext) → Message:` Decryption requires decryption keys corresponding to all the tags the message carries.