# Fortified-Edge 4.0: A ML-Based Error Correction Framework for Secure Authentication in Collaborative Edge Computing

Presenter: Venkata P. Yanambaka

Seema G. Aarella[1], Venkata P. Yanambaka[2], Saraju P.Mohanty[3], Elias Kougianos[4]

University of North Texas, Denton, TX 76203, USA.[1,3,4]

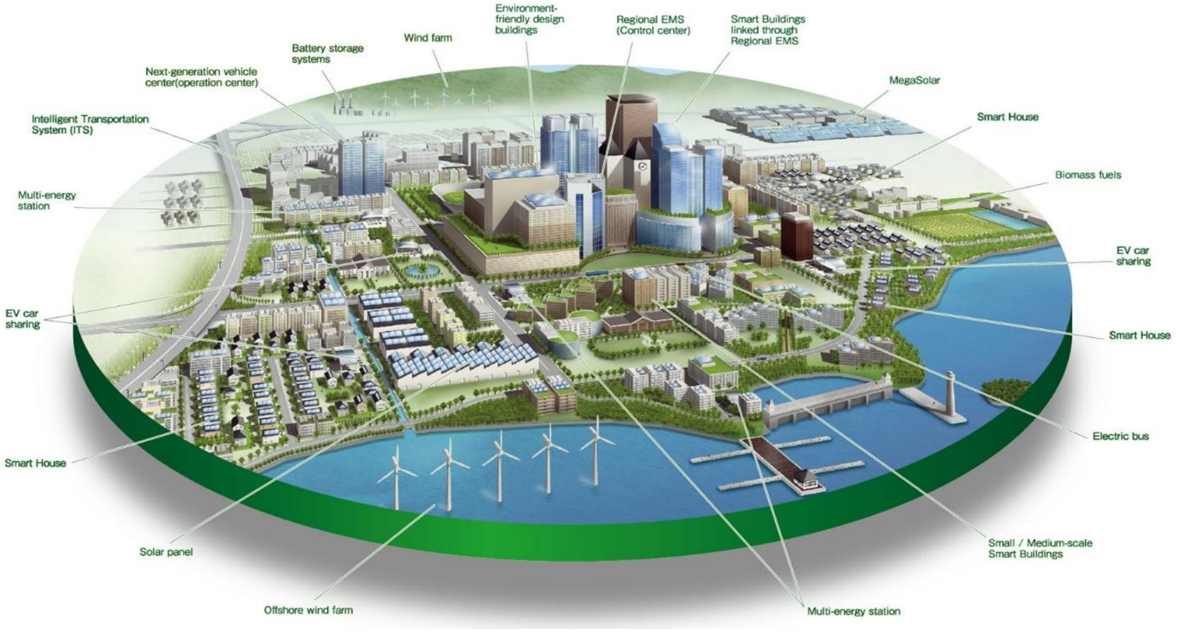
Texas Woman's University, Denton, TX, USA.[2]

Email: Seema.Aarella@unt.edu[1], vyanambaka@twu.edu[2], Saraju.Mohanty@unt.edu[3] and Elias.Kougianos@unt.edu[4]

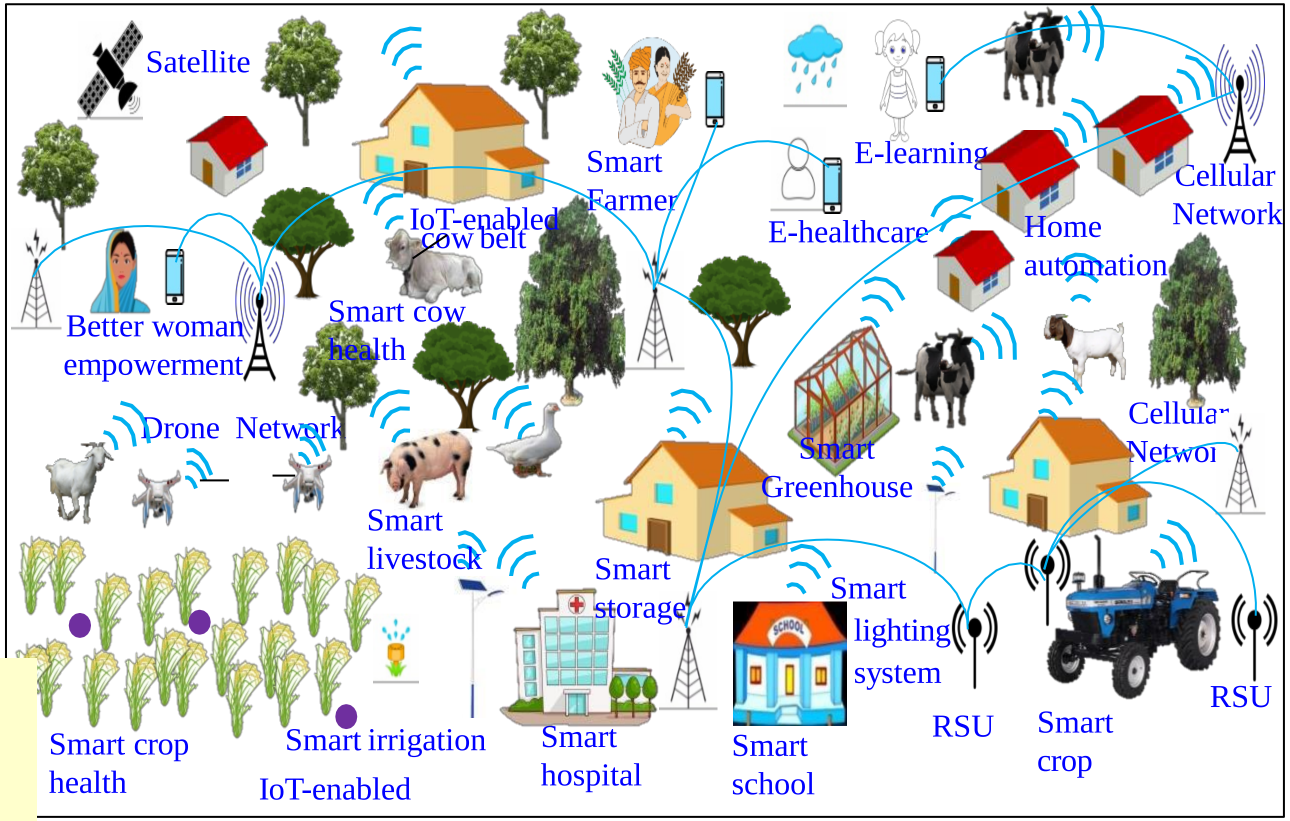
# Outline of the Talk

- Introduction
- Smart Cities and Smart Villages
- Need for Security-by-Design
- Novel Contributions
- Fortified-Edge Ecosystem
- Proposed Fortified-edge 4.0
- Experimental Setup
- Results and Analysis
- Conclusions

# Smart Cities Vs Smart Villages




Source: http://edwingarcia.info/2014/04/26/principal/




Source; P. Chanak and I. Banerjee, "Internet of Things-enabled Smart Villages: Recent Advances and Challenges," *IEEE Consumer Electronics Magazine*, DOI: 10.1109/MCE.2020.3013244.

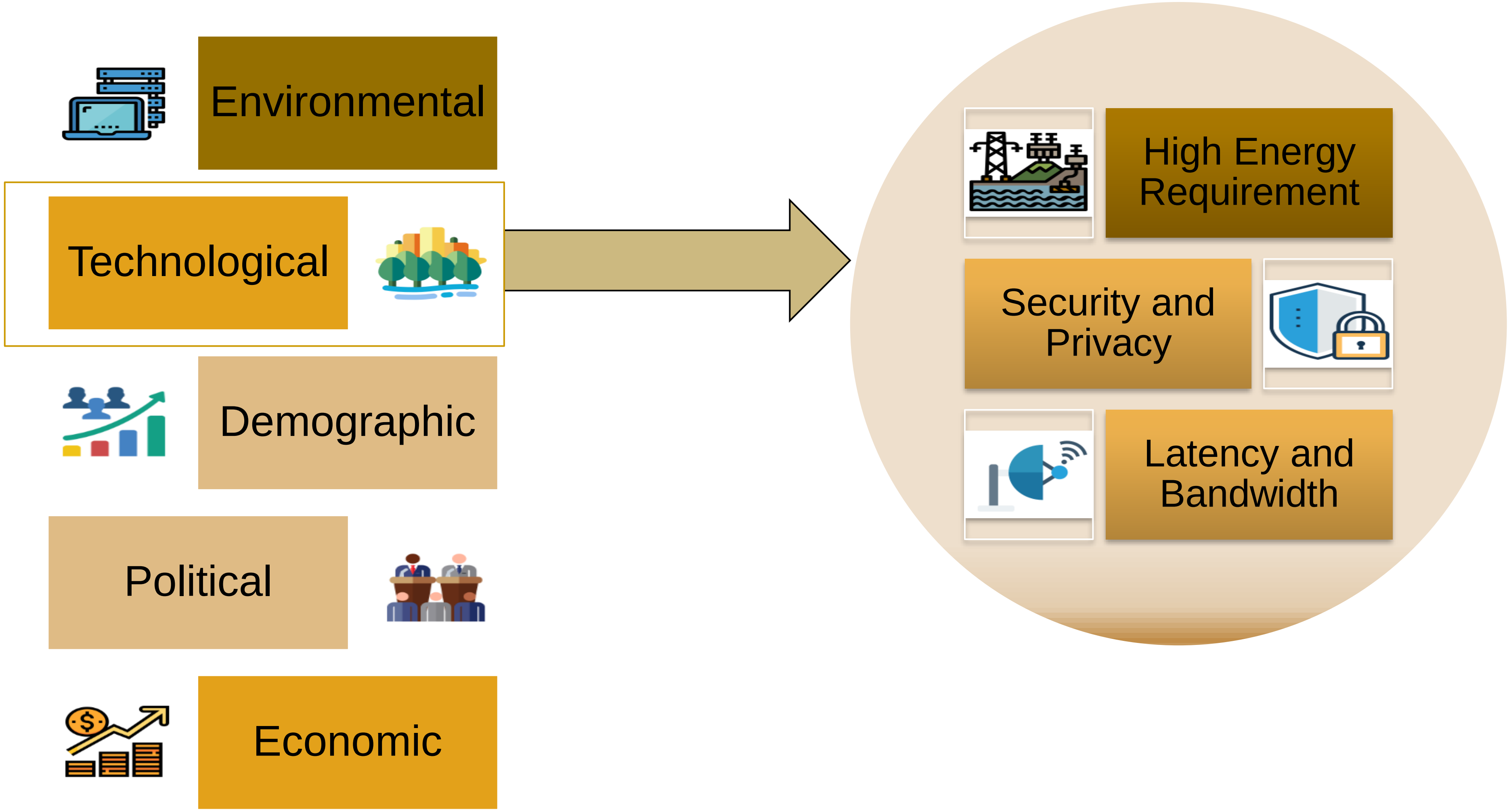
## Smart Villages

CPS Types - Less
Design Cost - Low
Operation Cost – Low
Energy Requirement - Low

## Smart Cities

CPS Types - More
Design Cost - High
Operation Cost – High
Energy Requirement - High

# Challenges of Smart Village



# Security-by-Design (SbD)

- Integration of the cybersecurity early in the design phase, not retrofitted

- Device, circuit, and system-level cybersecurity solutions for robust CPS and smart component design

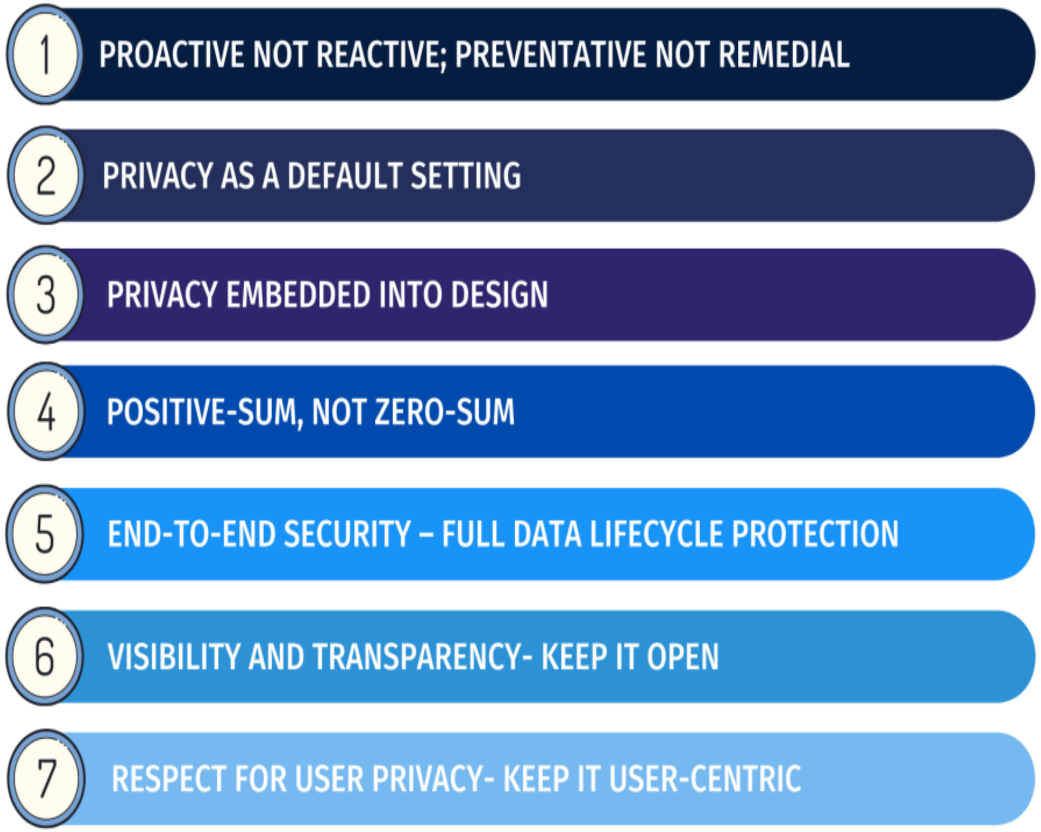
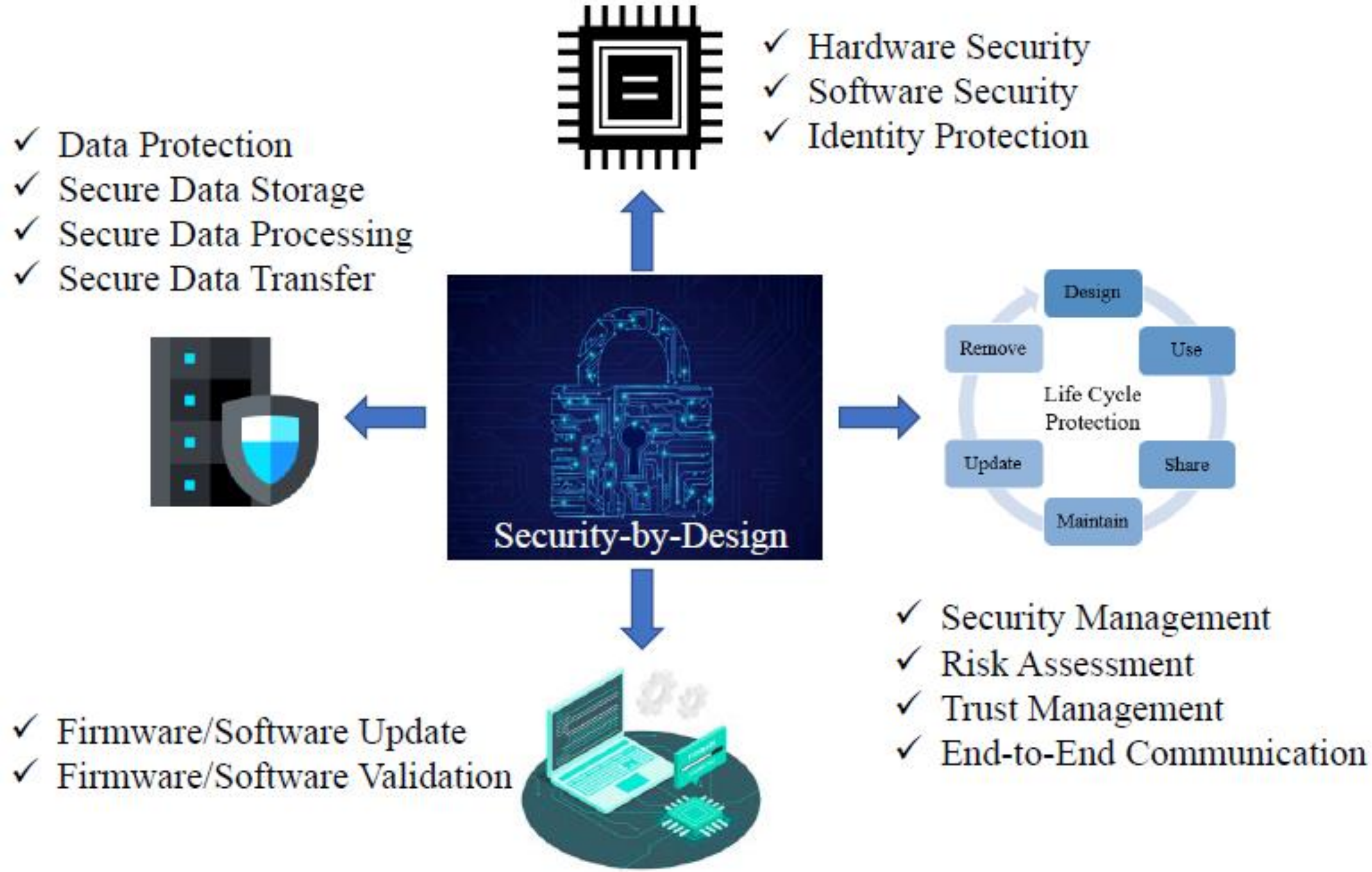
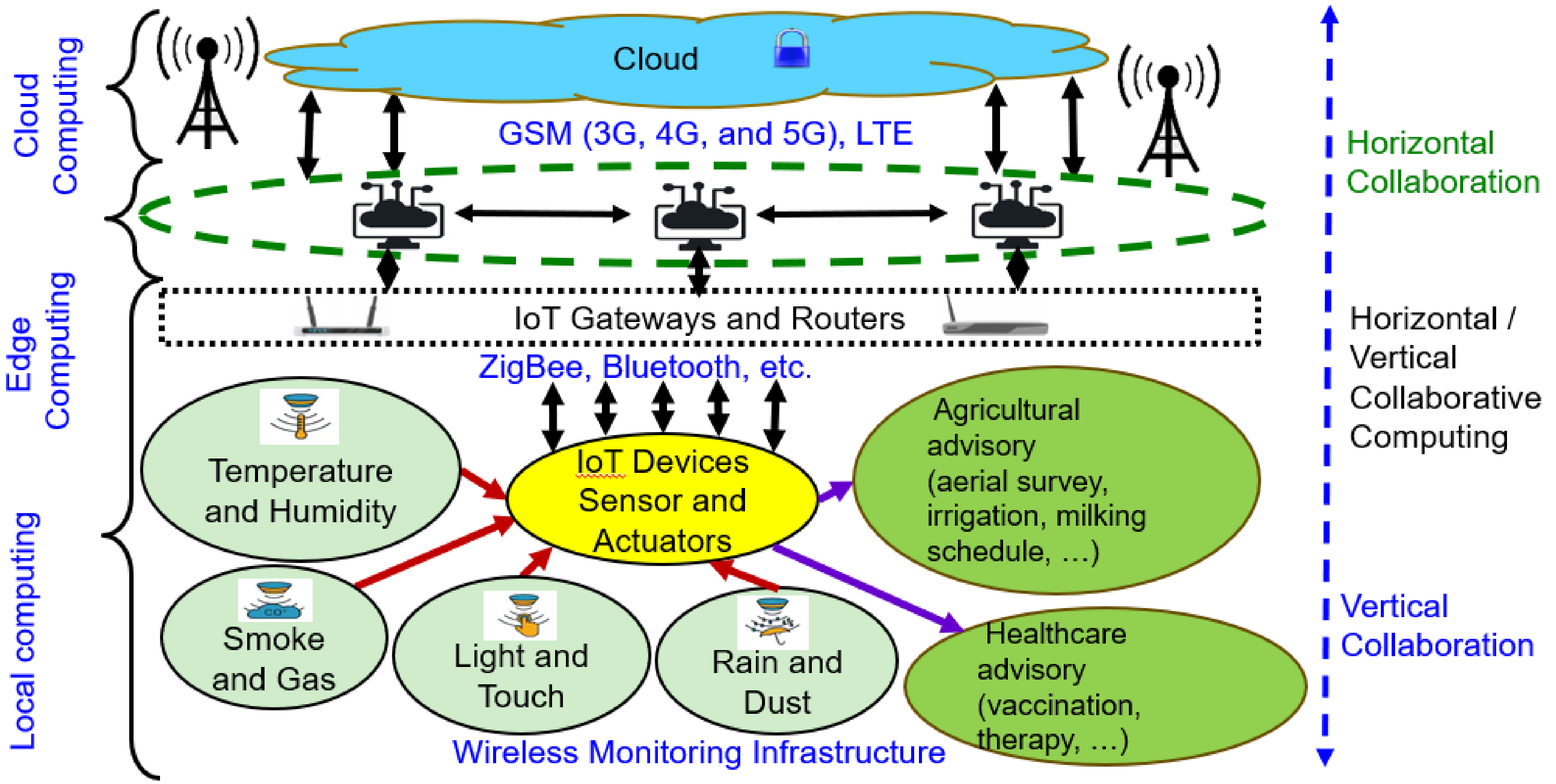



Image Source: https://dataprivacymanager.net/seve-principles-of-privacy-by-design-and-default-what-is-data-protection-by-design-and-default/

# Security-by-Design (SbD)

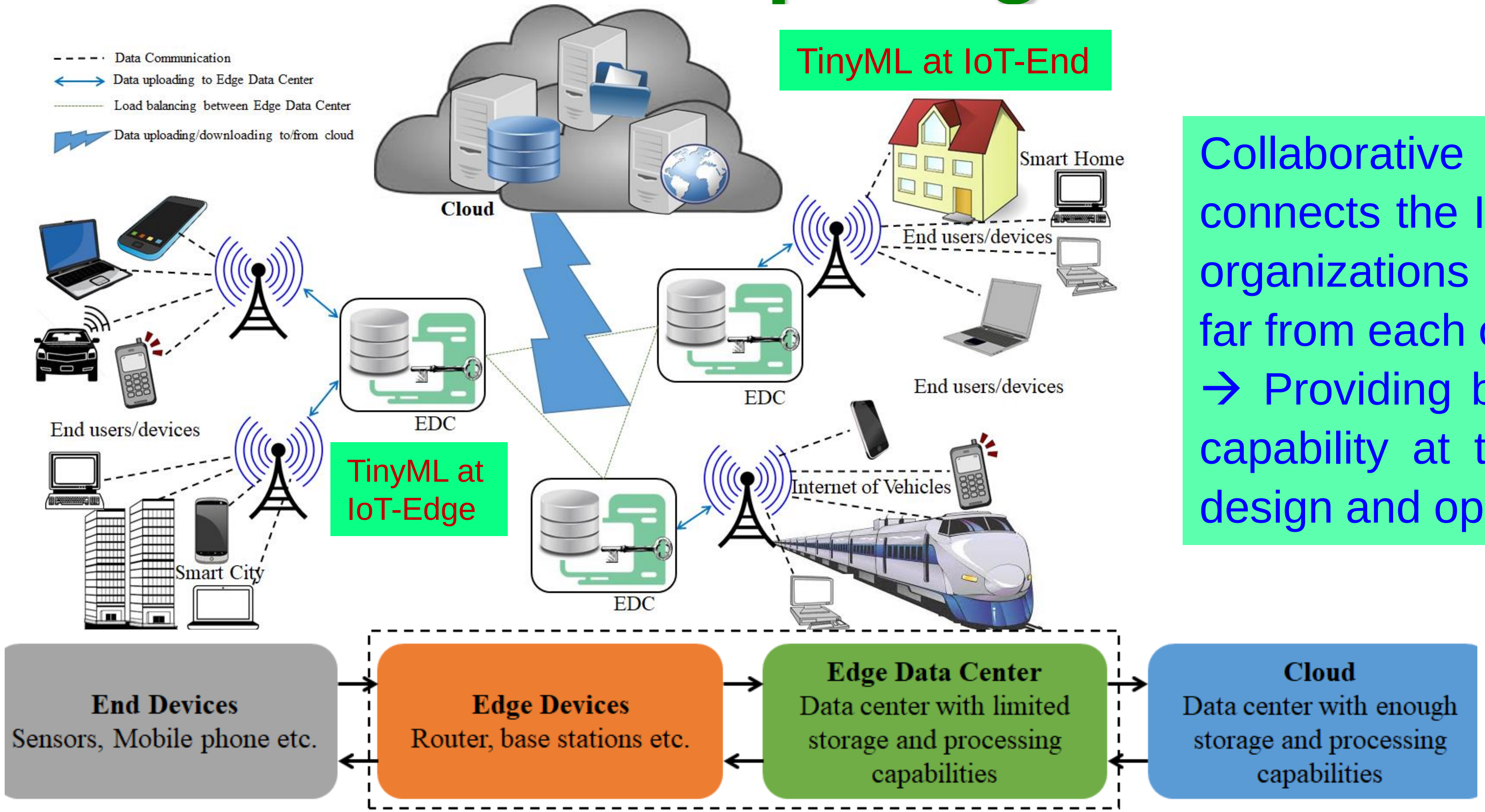


# Collaborative Edge Computing (CEC)




Source: D. Puthal, S. P. Mohanty, S. Wilson and U. Choppali, "Collaborative Edge Computing for Smart Villages", *IEEE Consumer Electronics Magazine (MCE)*, Vol. 10, No. 03, May 2021, pp. 68-71.

# Collaborative Edge Computing is Cost Effective Sustainable Computing for Smart Villages

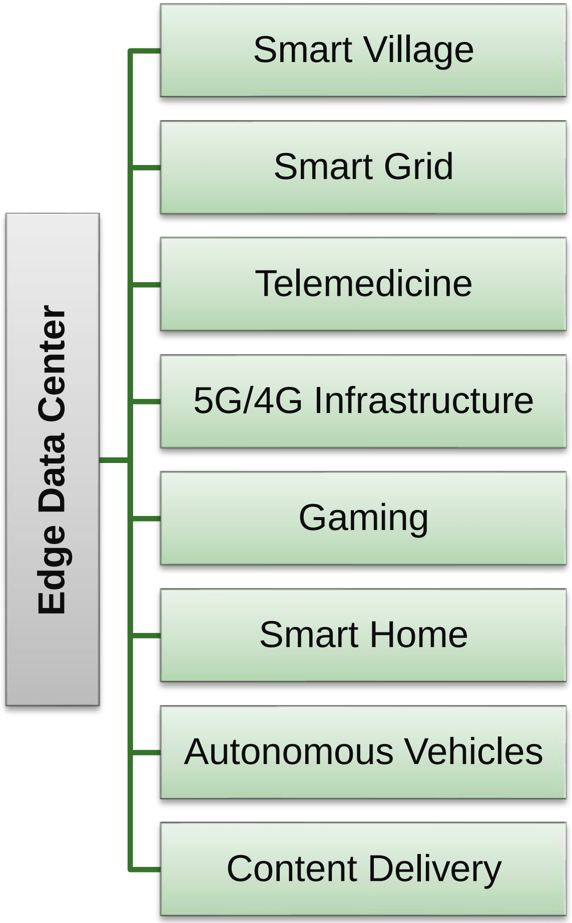



Collaborative edge computing connects the IoT-edges of multiple organizations that can be near or far from each other
→ Providing bigger computational capability at the edge with lower design and operation cost.

Source: D. Puthal, M. S. Obaidat, P. Nanda, M. Prasad, S. P. Mohanty, and A. Y. Zomaya, “Secure and Sustainable Load Balancing of Edge Data Centers in Fog Computing”, *IEEE Communications Mag*, Vol. 56, No 5, May 2018, pp. 60--65.

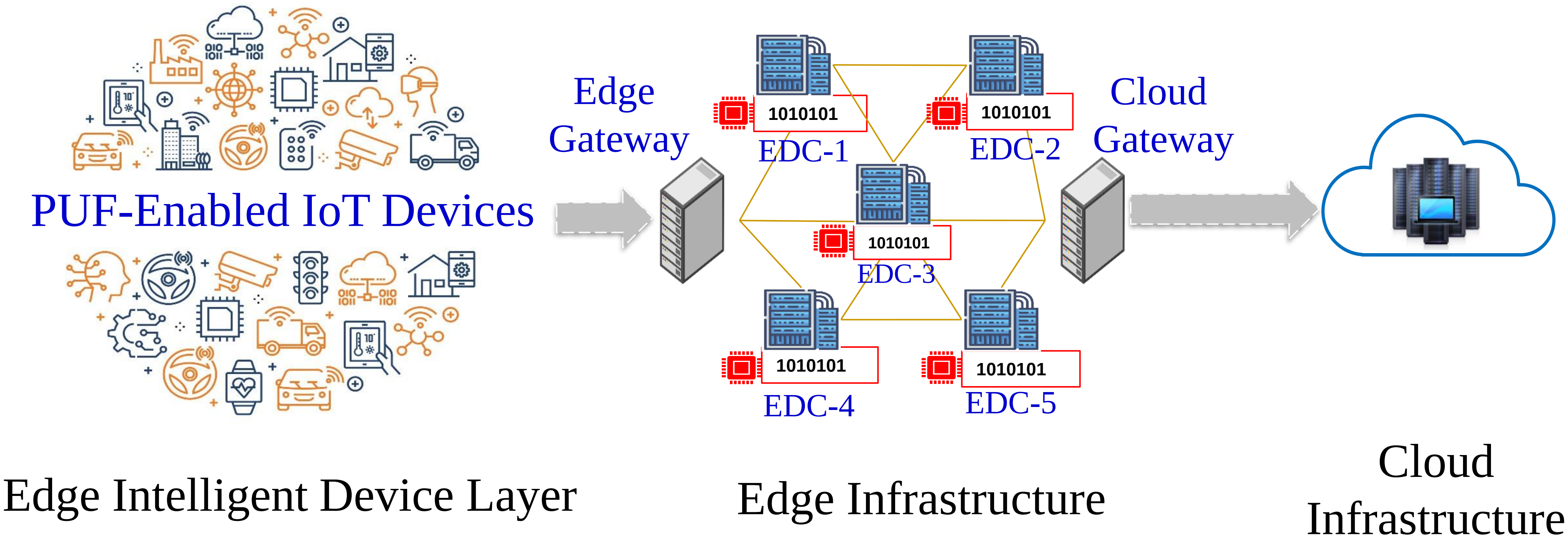
# Collaborative Edge Computing (CEC)

- Collaborative Edge Computing is a distributed processing environment
- CEC is a collaboration of distributed edge
- Smart control of heterogenous network
- Reduced Bandwidth and Transmission costs
- CEC enables seamless processing through load balancing



# Secure Authentication of EDC in CEC

Load Balancing in Collaborative Edge Computing



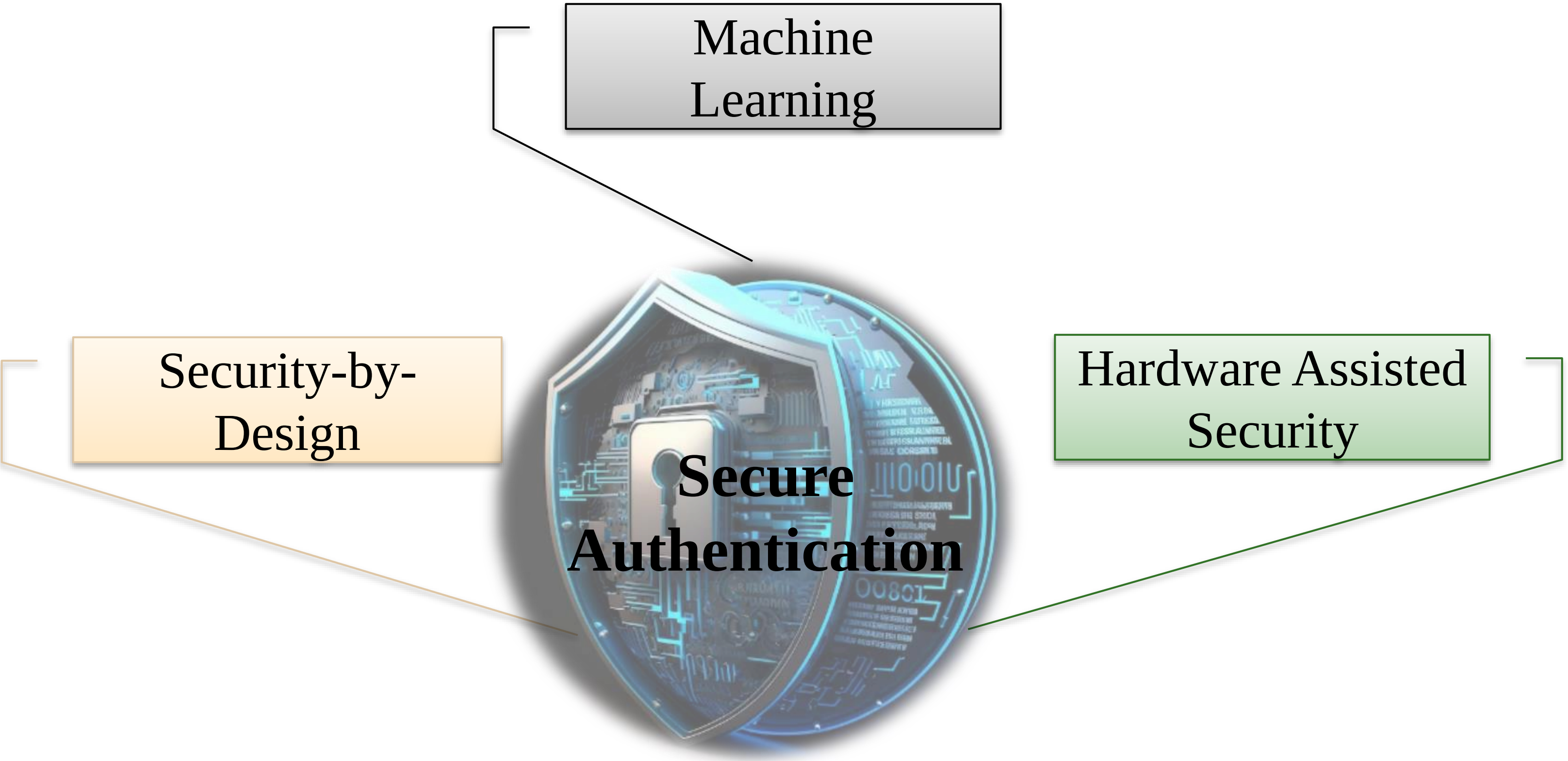
# Long-term Vision

Cybersecurity for smart villages based on the SbD principles for secure resource sharing in the CEC environment

AI/ML for Cybersecurity in Smart Villages

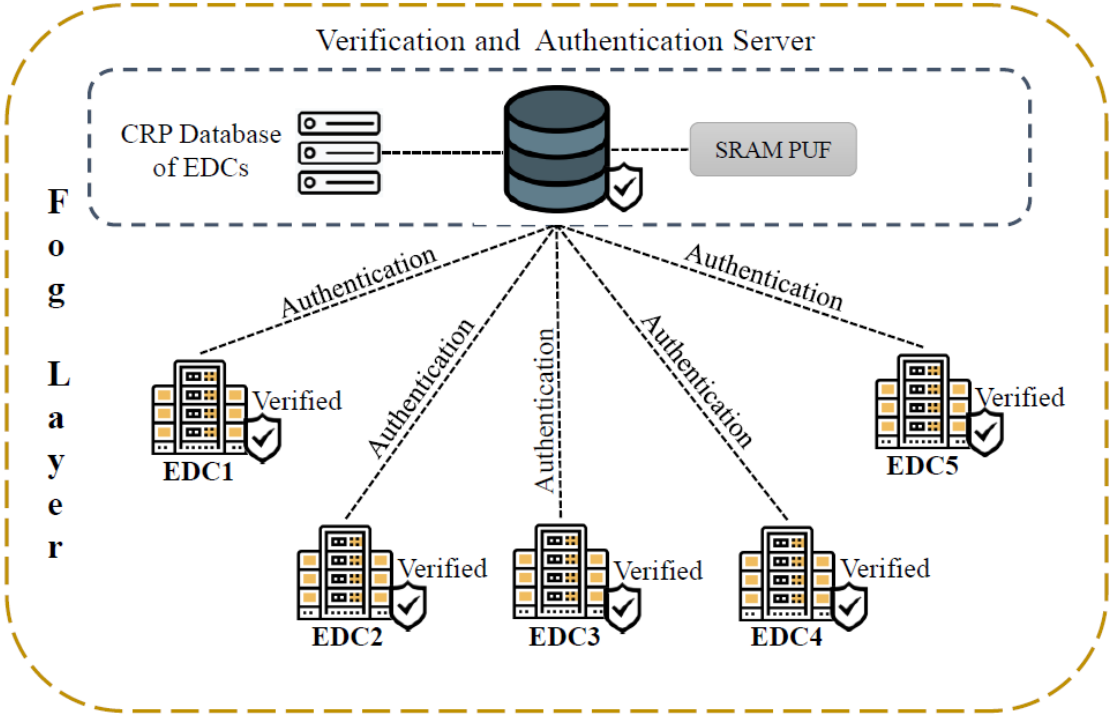
# Our Fortified-Edge: The Key Idea

- A lightweight and Secure Authentication scheme for EDCs during load balancing in the CEC environment of smart villages



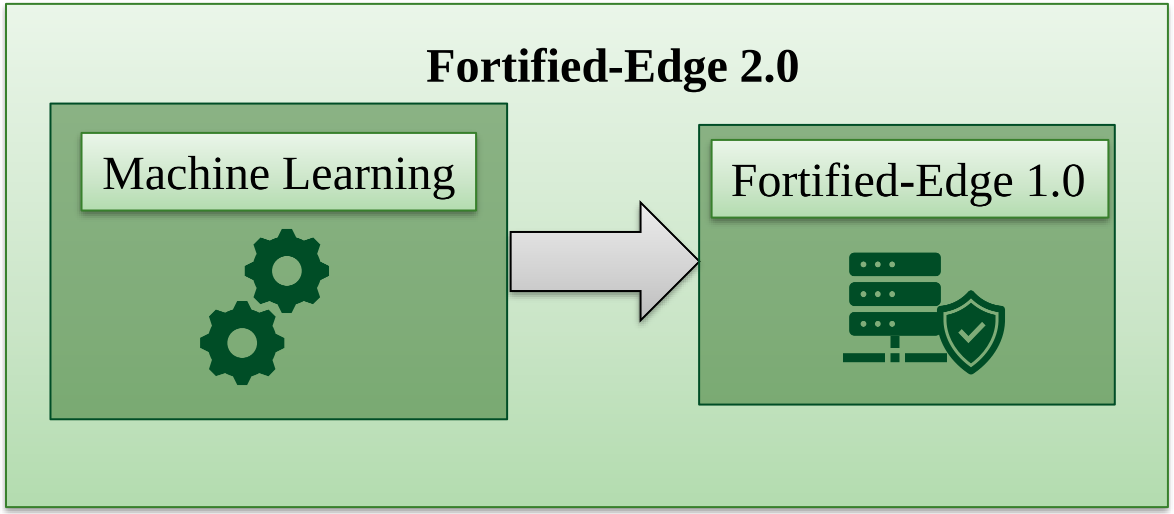
# Fortified-Edge 1.0 - The Idea

- CEC enables applications in smart villages through load balancing
- To develop a secure authentication protocol for Load balancing
- Suitable for a smart village environment
- Incorporate Security-by-Design for smart and sustainable security




Source: S. G. Aarella, **S. P. Mohanty**, E. Kougianos, and D. Puthal, "Fortified-Edge: Secure PUF Certificate Authentication Mechanism for Edge Data Centers in Collaborative Edge Computing", in *Proceedings of the ACM Great Lakes Symposium on VLSI (GLSVLSI)*, 2023, pp. 249--254, DOI: https://doi.org/10.1145/3583781.3590249

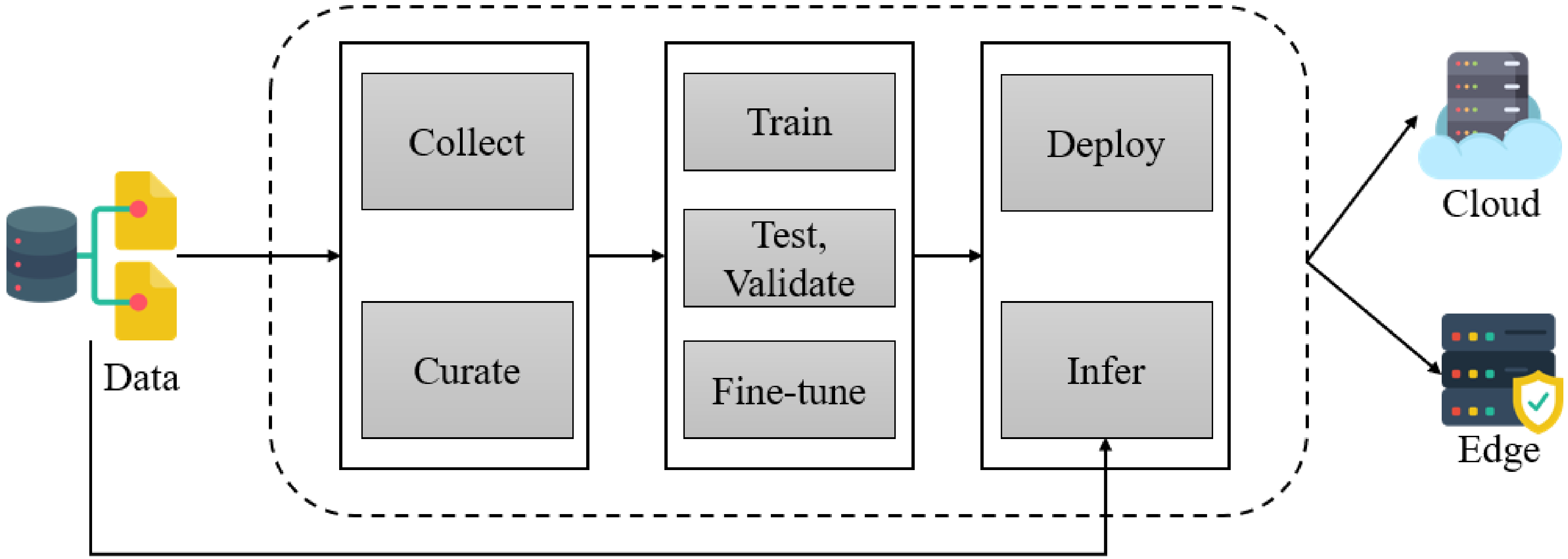
# Fortified-Edge 2.0 - The Idea




Features

- Secure, Low Latency Authentication
- Device identification
- Intrusion detection
- Attack Prevention
- EDC Monitoring
- Resilient against malicious Requests
- ML model suitable for a smaller dataset

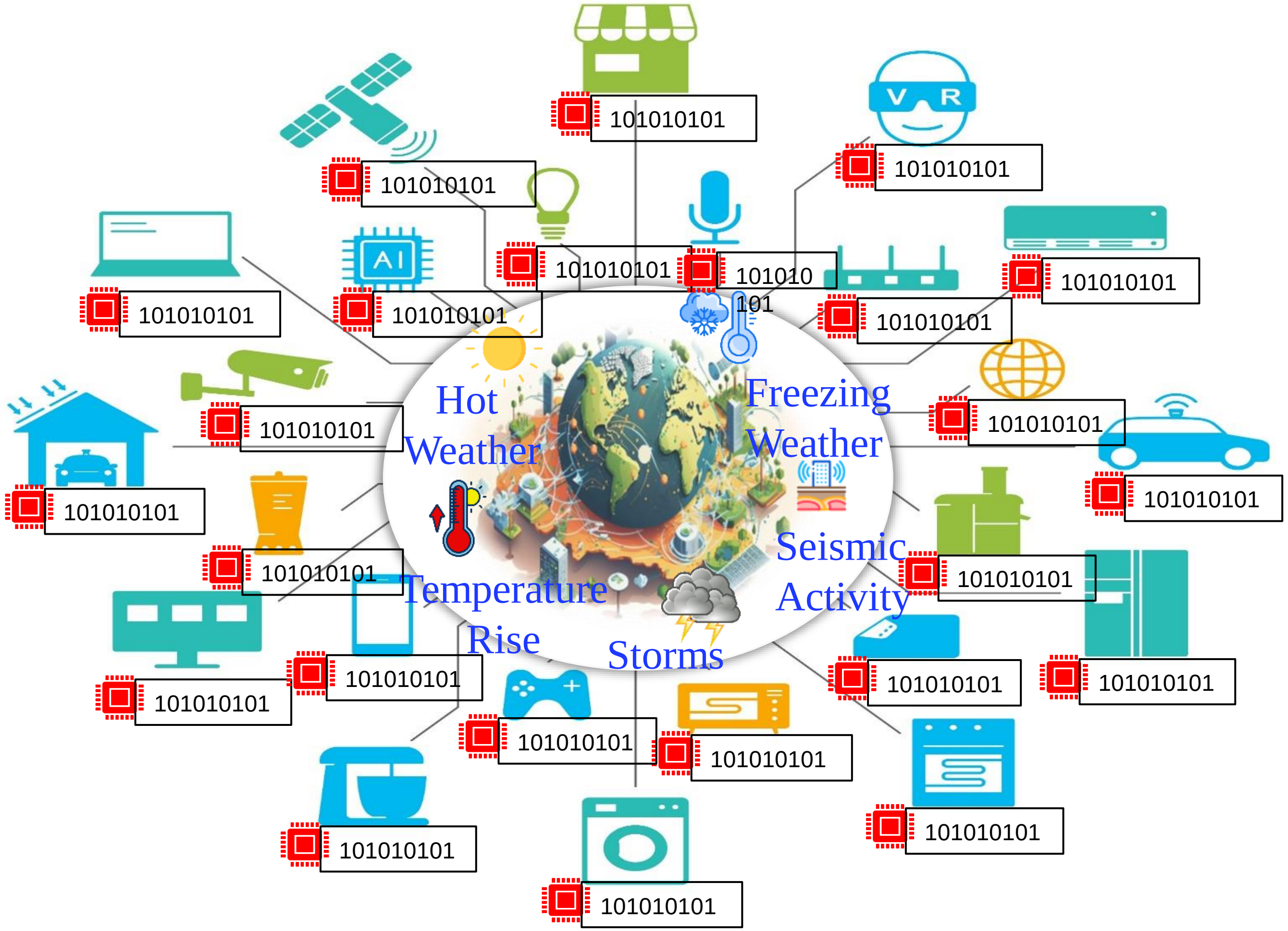
# Fortified Edge 3.0 Machine Learning for Edge



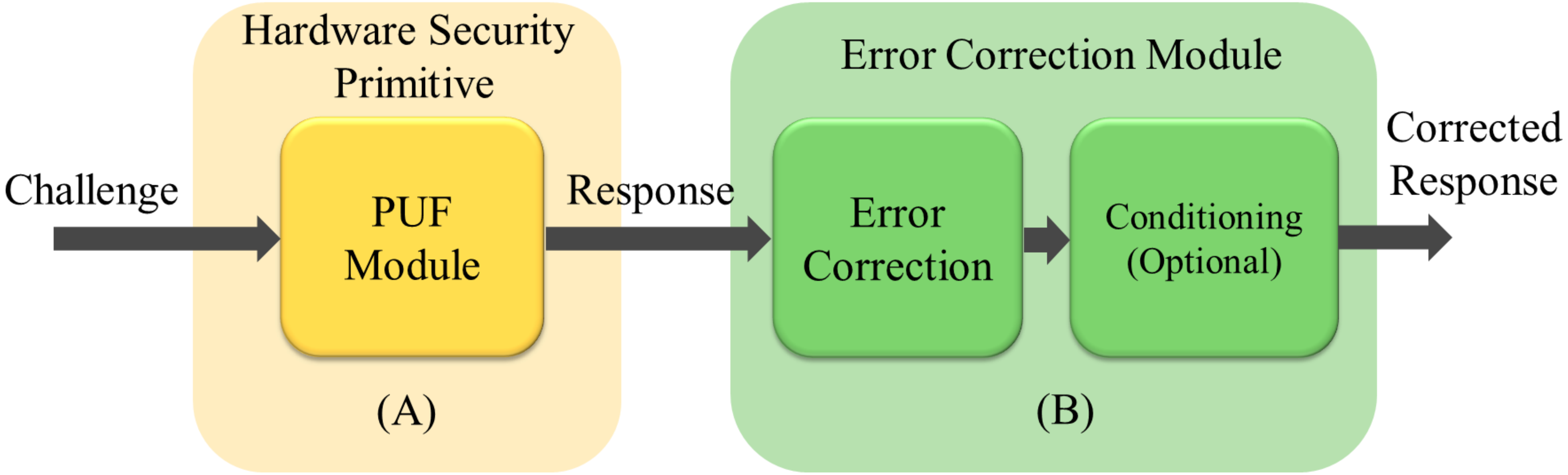

- **Develop**
  - Security System
- **Train**
  - ML Models
- **Deploy**
  - Cloud or Edge

# Fortified- Edge 4.0 - Motivation

- Environmental effects on PUF
- Bit flipping in PUF response affects the performance
- Bit errors reduce the reliability
- Security applications need reliable PUF



# PUF Security Model for Bit Error Correction



# Related Prior Research

| Research | Year | ML Algorithm | Application |
|---|---|---|---|
| Upadhyaya et. al. [20] | 2019 | Natural Redundancy decoders based on Machine Learning | Error Correction |
| Suragani et. al. [19] | 2022 | Proof-of-Concept using CNN | Classification of corrupted PUF responses |
| Chatterjee et. al. [5] | 2020 | Random Forest based PUF Calibration scheme | Validate sensor data |
| Najafi et. al. [14] | 2021 | Deep CNN | Recognize PUF responses under error conditions |
| Wen et. al. [21] | 2017 | Fuzzy Extractor | PUF reliability |
| **Current Research Fortified-Edge 4.0** | **2024** | **K-mer Sequence** | **PUF bit error correction** |

# Novel Contributions of Current Research

- Novel machine learning method for bit error detection and correction
- Data preprocessing done through visualization, data cleaning
- Sequencing methods used in DNA sequencing and Natural Language Processing (NLP)
- Vectorization of the sequences
- MultinomialNB  for classification
- A deployable working model that can predict the correct response from the response with an error

# Problems Addressed and Solutions Proposed

## Problems Addressed

- Area overhead added by the bit error correction module
- Computational overhead
- Extensive error correction schemes do not suit the lightweight aspect of the security system
- Data leakage issues related to helper data in schemes that use helper bits
- Secure storage of the helper data, an added feature making the design complex

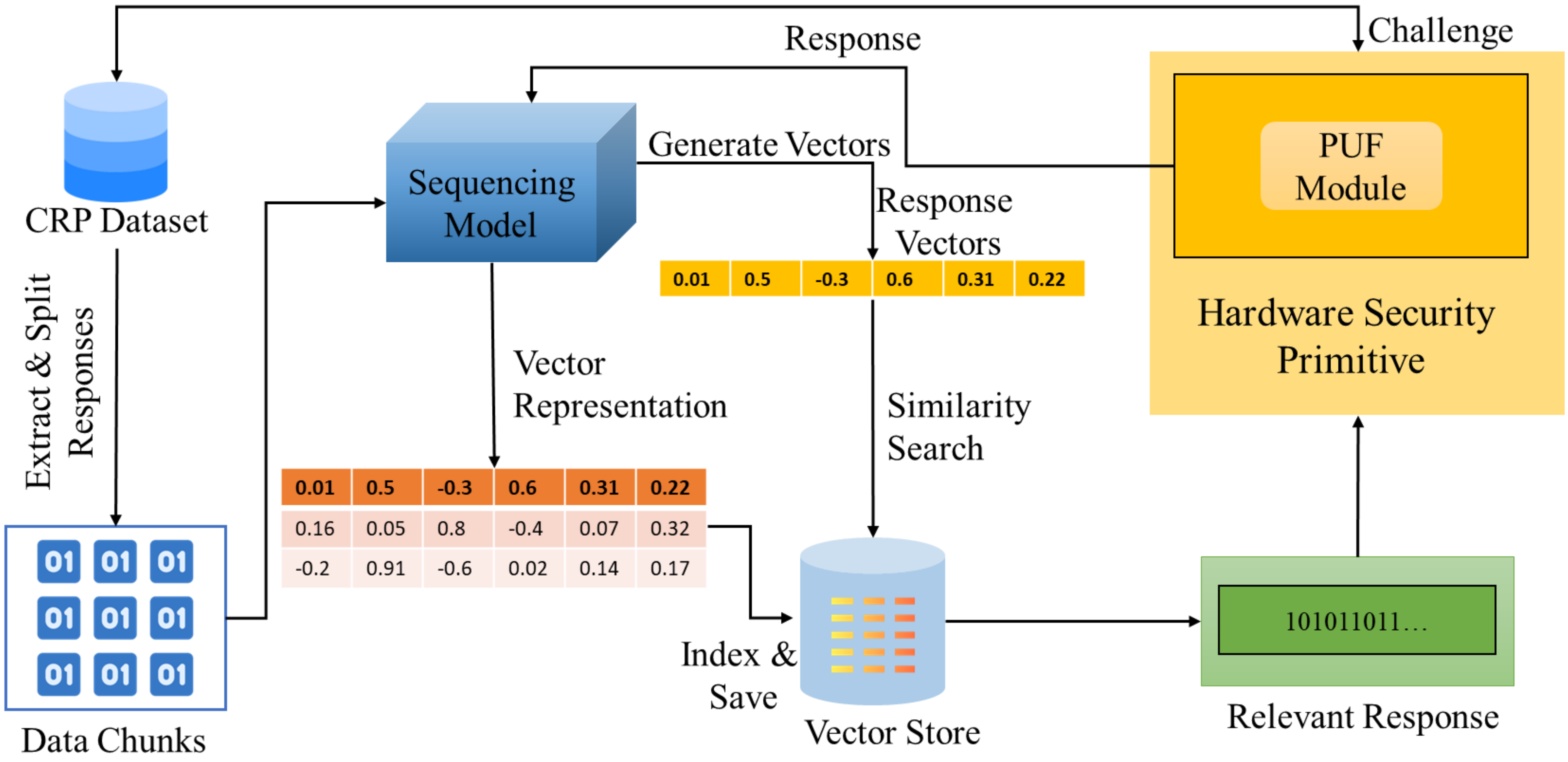
## Solutions Proposed

- ✓ The area overhead and computational overhead are low as the trained model is used at the device end
- ✓ There is no need to store the helper data
- ✓ Helper data leakage is not an issue as the trained model is deployed at the device end
- ✓ The ML model is highly accurate in correcting the erroneous response bits

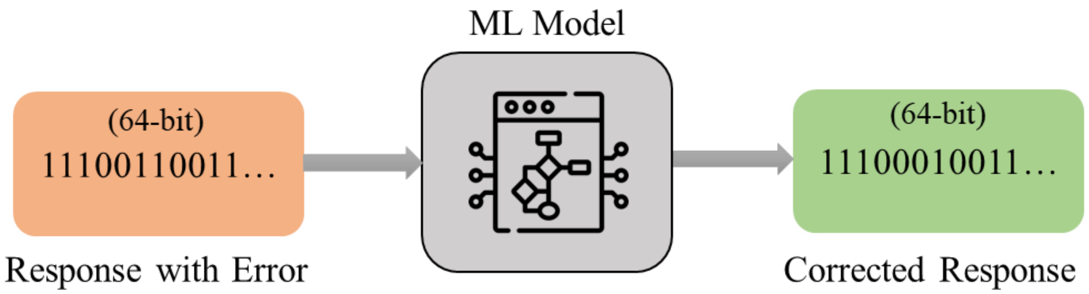
# Fortified-Edge Research

| Research | Algorithm | Application | Accuracy |
|---|---|---|---|
| Fortified-Edge 1.0 [4] | SRAM PUF-based Certificate | EDC Authentication | NA |
| Fortified-Edge 2.0[3] | SVM | ML-based Authentication & Monitoring | 100.0 |
| Fortified-Edge 3.0[1] | Lightweight ML models | Anomaly & Intrusion detection | 99.33 |
| **Current Research Fortified-Edge 4.0** | **K-mer Sequence** | **PUF Response Bit Error Correction** | **99.74** |

# PROPOSED FORTIFIED-EDGE 4.0



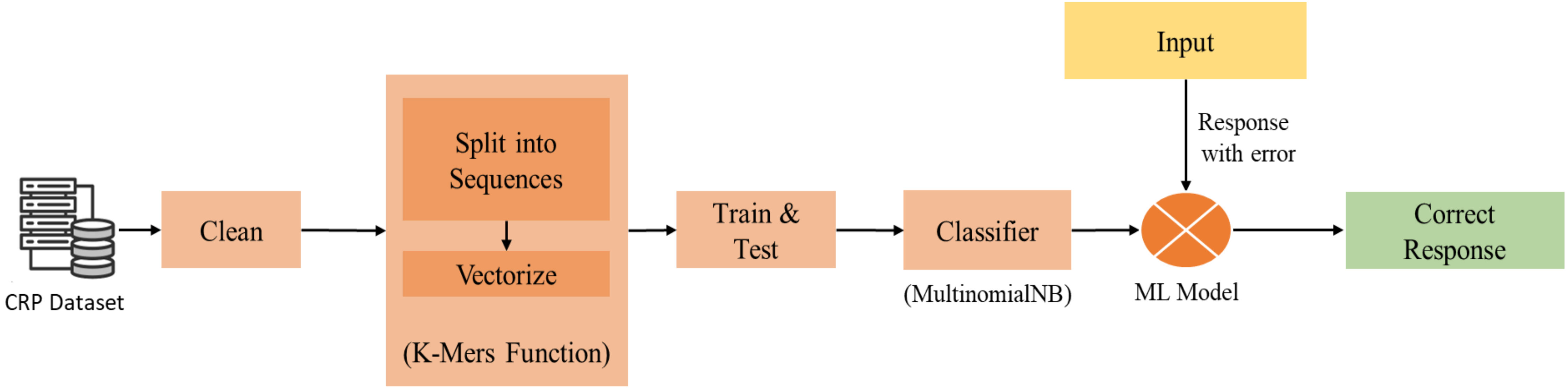
# Experimental Setup




- This research uses the 64-bit Arbiter PUF architecture
- PUFs. PYNQ™ Z2 FPGA which is based on Xilinx Zynq C7Z020 SoC used for PUF implementation
- Xilinx BASYS3 FPGA used to build PUF
- Raspberry Pi 4 to test the trained model
- Uniqueness, Randomness, and Hamming Distance are used to measure the performance of PUF
- Precision, Recall, Accuracy, and F-1 Score are the metrics used for the performance of the ML model

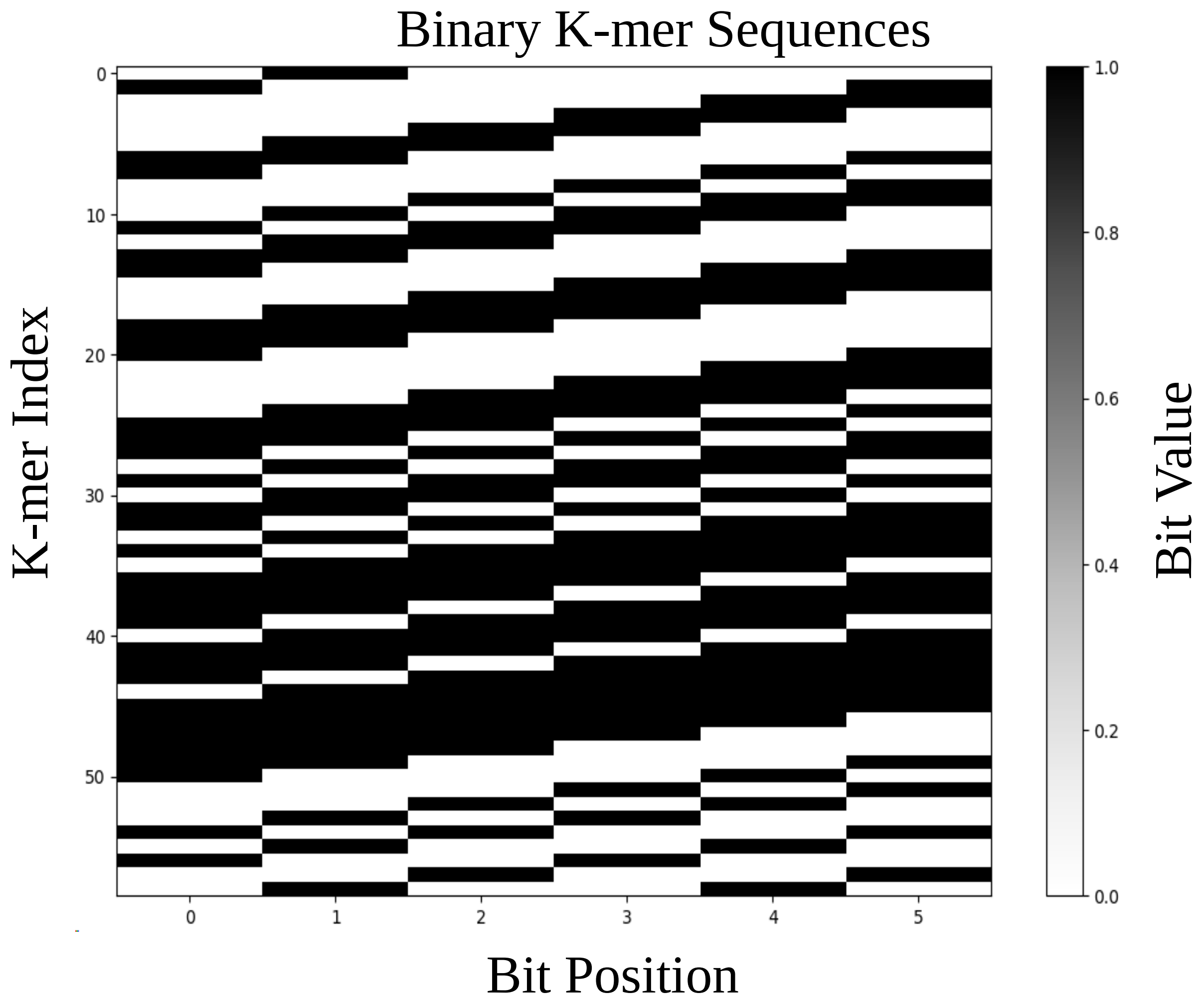
# Process Flow

- The performance of the arbiter PUF - 49.52% Uniqueness, 86.85% Randomness, and 45.67% inter-HD
- Dataset – 100K Responses, 1000 Challenges, 100 Responses for each Challenge
- Dataset includes responses with error, 80% training set, 20% testing set

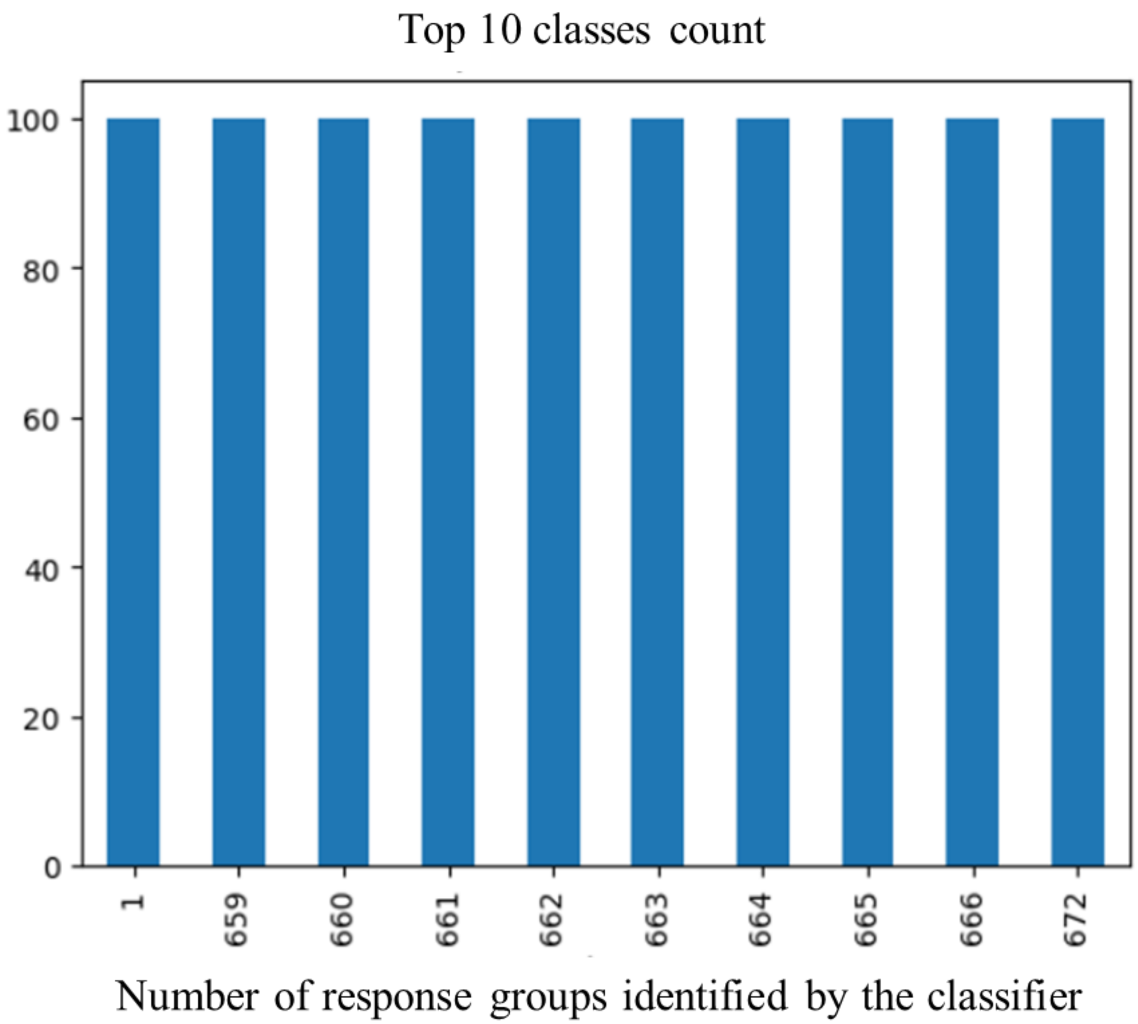


# Experimental Results

- K-mers size of 6  applied on string data generates 51 unique sequences
- 511 features are generated from 100K data
- The visual representation of the K-mer sequences helps to study the pattern distribution
- Examine bit positions for diversity, consistency, and other structural information



# Experimental Results




- 100K data initialized as string
- The algorithm identifies 100 classes
- Group the responses into unique classes

[1000 rows x 1000 columns]

| Actual | 554 | 958 | 47 | 164 | 309 | 397 | 498 | 883 | 24 | 118 |
|---|---|---|---|---|---|---|---|---|---|---|
| Actual | | | | | | | | | | |
| 554 | 33 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 958 | 0 | 32 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 47 | 0 | 0 | 31 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 164 | 0 | 0 | 0 | 31 | 0 | 0 | 0 | 0 | 0 | 0 |
| 309 | 0 | 0 | 0 | 0 | 31 | 0 | 0 | 0 | 0 | 0 |
| 397 | 0 | 0 | 0 | 0 | 0 | 31 | 0 | 0 | 0 | 0 |
| 498 | 0 | 0 | 0 | 0 | 0 | 0 | 31 | 0 | 0 | 0 |
| 883 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 31 | 0 | 0 |
| 24 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 30 | 0 |
| 118 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 30 |

# Experimental Results

```
Input Response:      110010011000001010111100001110100011011111000011100010
Predicted Class: 293
***************************************************************
Actual Class:  293
Corrected Response:  1100100110000010101111000011101000110111110000111100010
```

The result of the algorithm shows the input response and corrected response

```
Predicted ID:  293
```

| | ID | Challenge | Corresponding_Response |
|---|---|---|---|
| 0 | 293 | 1001000111011101011111100001111111010010011100... | 1100100110000010101111000011101000110111110000... |

The result shows the corresponding challenge to the corrected response

# Analysis of Results

- The trained model is deployed on Raspberry Pi 4 and both training and prediction analysis are done
- The algorithm is evaluated for accuracy, precision, recall, and F1-score, for each of the metrics it gives 100% results
- Coverage Rate - gives insights into the percentage of unique K-mers in the dataset, increasing the size of the K-mers it was observed that the coverage rate increases significantly
- Test for overfitting - KFold cross-validation was done
- The cross-validation scores - 99.78%, 99.74%, 99.70%, 99.75%, 99.73%, with a mean accuracy of 99.74%

# Analysis of Results

- The process of training and prediction of new data is analyzed for time and power consumed
- Raspberry Pi 4 used as an edge device for training and prediction
- Processing speed – 13.15 sequences per second
- Processing Power – 0.28 sequences per character
- Idle Power of Raspberry Pi 4 – 3.4-3.6 Watts

| | Time(s) | Power (W) |
|---|---|---|
| Training | 30.63 | 4.6-4.7 |
| Prediction | 0.08 | 4.1 |

# Comparative Table for State-of-the-Art Literature

| Research | Year | Algorithm | Accuracy |
|---|---|---|---|
| Upadhyaya et. al. [20] | 2019 | Natural Redundancy decoders based on Machine Learning | NA |
| Suragani et. al. [19] | 2022 | Proof-of-Concept using CNN | 97.34 |
| Chatterjee et. al. [5] | 2020 | Random Forest based PUF Calibration scheme | 90.00 |
| Najafi et. al. [14] | 2021 | Deep CNN | 94.90 |
| Wen et. al. [21] | 2017 | Fuzzy Extractor | 98.00 |
| **Current Research Fortified-Edge 4.0** | **2024** | **K-mer Sequence** | **99.74** |

# Conclusion

- This research proposes a novel K-mer sequence-based bit error detection and correction algorithm for correcting the PUF responses
- The stability of the PUF response increases the reliability of the PUF when employing it in security and cryptographic applications
- The power and time analysis proves that the ML model is low power consuming, and faster in processing
- Suitable for EDC Authentication in resource-constrained environments at the edge
- The multinomialNB classifier used is fast and computationally efficient

# Future Research

- For future research, we are considering using this reliable PUF architecture for deepfake detection or prevention
- This PUF module can be used as a device authenticator if installed in the camera module to identify the device
- The machine learning model can be used as a verifier for the images generated from the authorized device

# Thank you!