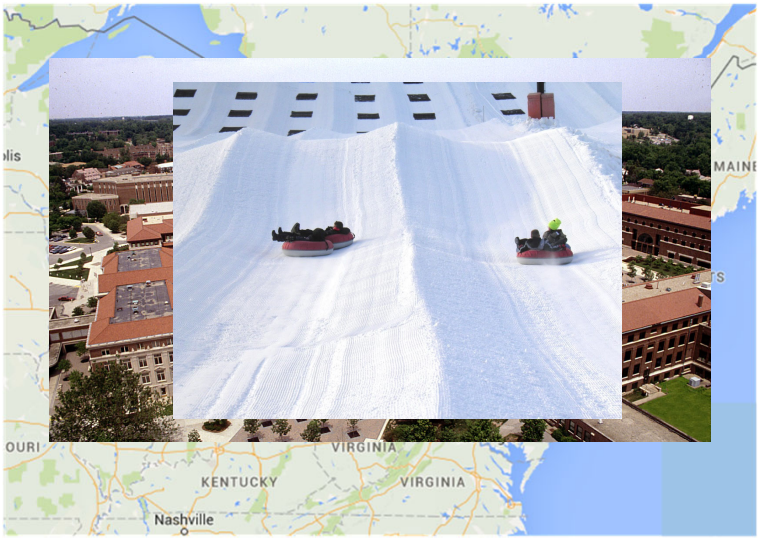
Dependability and Machine Learning: A Grand Challenge of the Times

Saurabh Bagchi
Purdue University
School of Electrical and Computer Engineering
Department of Computer Science
Director, Center for Resilient Infrastructures, Systems and Processes (CRISP)
Director, Army Assured Autonomy Innovation Institute (A2I2)

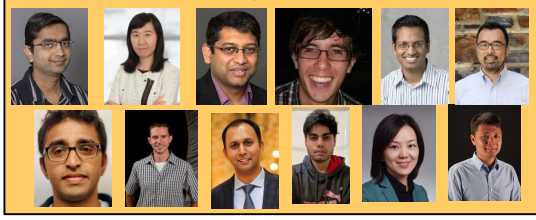
PURDUE UNIVERSITY

Greetings come to you from …



CRISP

Cyber



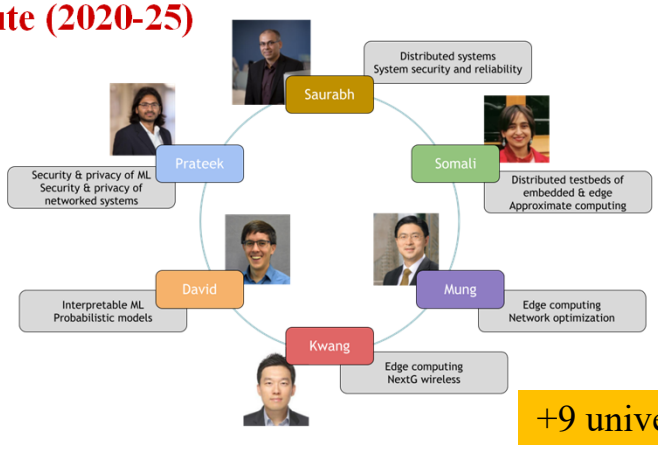
35 faculty
ECE: 9, CS: 6,
Other engineering departments: 20

Cyber-Physical

Socio-Technical

U.S.ARMY DEVCOM ARMY RESEARCH LABORATORY

- **Army Research Lab: Assured Autonomy Innovation Institute (2020-25)**



+9 universities

Where It All Started

Roadmap

Dependability across the stack

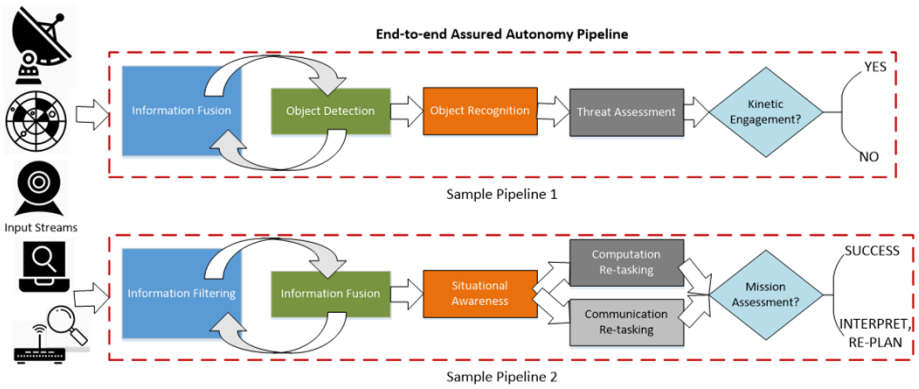
- Security in Federated Learning: [AsiaCCS-23, DSN-23]
- Privacy in Federated Learning: [CVPR-23, S&P-24]
- Takeaways and Open Problems

Fun tech quiz:
https://bit.ly/sbquiz23

Problem Context

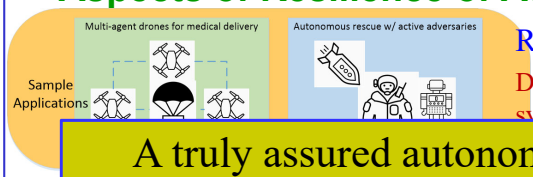
- Multi-domain operations in defense will involve **autonomous operations** among multiple cyber, physical, and kinetic assets, together with interactions with humans
- Such autonomous operation will rely on:
  - A **pipeline** of machine learning (ML) algorithms
  - Executing in **real-time** on
  - A **distributed** set of **heterogeneous platforms**
- Conditions will be adversarial and operation must be guaranteed to be secure while **maintaining timeliness guarantees**
  - Security guarantees need to be carefully analyzed and proven, under well-quantified adversary models
  - Move away from one-off solution for specific attack type
- Different degrees of autonomy
  - Some require humans in the loop; in such cases, the cognitive load of any software solution must be analyzed for feasibility
  - Some require humans on the loop
  - Some are fully autonomous agents

Sample Autonomy Pipelines



- Multiple ML algorithms in a pipeline
  - Different resource requirements, different input-output patterns
  - Can be required to execute on vastly different platforms

Aspects of Resilience of Autonomous Systems



Resilience by design:

Designs & develops autonomous systems so that they are resilient to a

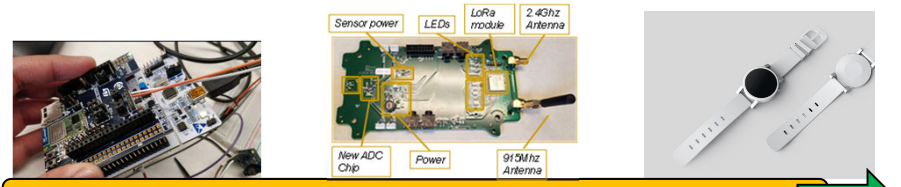
A truly assured autonomous system has:

1. *Close interactions* during design and execution of the two aspects
2. Resilience by reaction principles are *learned* and become part of resilience by design

Perturbations

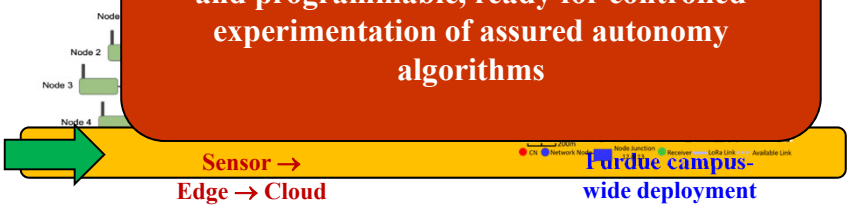
S. Bagchi, V. Aggarwal, S. Chaterji, F. Douglis, A. Gamal, J. Han, B.J. Henz, H. Hoffmann, S. Jana, M. Kulkarni, F.X. Lin, K. Marais, P. Mittal, S. Mou, X. Qiu, and G. Scutari, "*Vision Paper: Grand Challenges in Resilience: Autonomous System Resilience through Design and Runtime Measures*," in IEEE Open Journal of the Computer Society (OJCS), pp. 1-15, 2020.

Testbed for Evaluation of Assured Autonomy Protocols



Low-
embe
devi

The integrated testbed is network accessible and programmable, ready for controlled experimentation of assured autonomy algorithms



Sensor → Edge → Cloud

Purdue campus-wide deployment of mesh network

Requirements for Assured Autonomy

1. ML algorithms must be capable of
   - Executing on **a distributed set of execution platforms** (mobile nodes, ground-based or aerial sensors, edge computing nodes, private cloud nodes, etc.)
   - Trained **both offline** and **in the field**
   - Tolerating **varying amounts of noise** either due to naturally occurring causes or due to maliciously injected errors
2. Autonomous algorithms must interface well with humans who may need to act on their decisions
   - Interpretable and explainable at **the tactical level** in real time
   - Interpretable and explainable at **the strategic level** so that a leader may make modifications for future missions

Requirements

3. Probabilistic guarantees
   - On **accuracy** *and* **latency**
   - Guarantees must hold under adversarial actions
   - Guarantees must hold under batch mode *and* incremental training
4. Algorithms must be able to ingest heterogeneous sources of data
   - Data sources will vary in their **fidelity**, **rate**, and **characteristic**
   - These data sources will be **intermixed** coming from white, blue, and red networks
   - In the process of inferencing, the algorithms also **tag data sources with their trust level**, so that future decision making becomes more accurate

Roadmap

Dependability across the stack

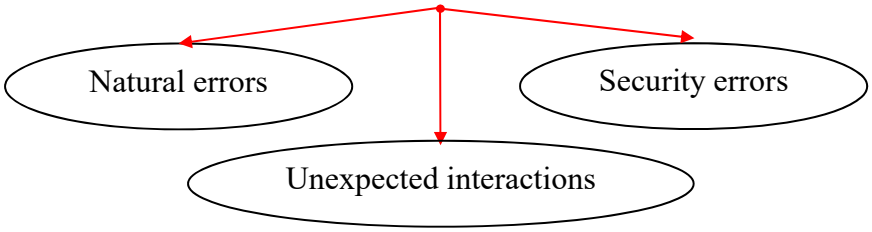
- Networks of wireless embedded devices
- Wearables and mobiles
- Clusters of edge devices
- Autonomous systems

1. Problem context
2. Domain-specific requirements
3. Application of data analytics

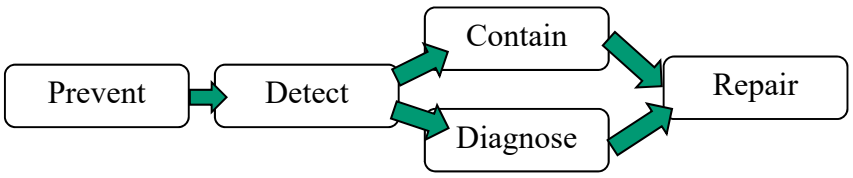
- Security in Federated Learning: [AsiaCCS-23, DSN-23]
- Privacy in Federated Learning: [CVPR-23, S&P-24]
- Takeaways and Open Problems

Dependability Basic

System provides intended functionality despite



Workflow for achieving dependability (Simplistic)



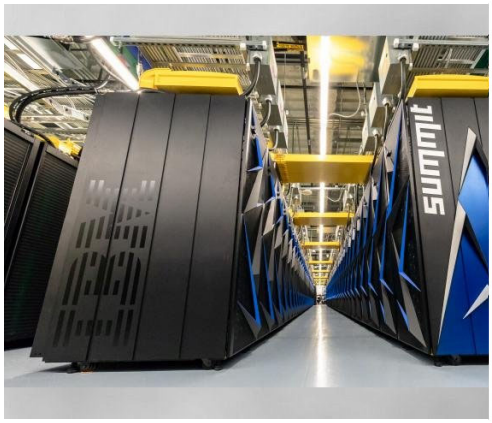
Dependability Trends

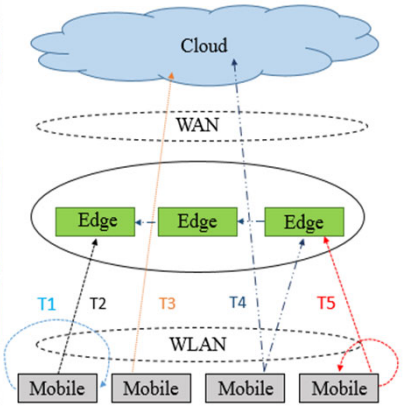
- Dependability driven by 2 emerging trends

1. Execution environments are changing
   - FROM: All components have perfect visibility and controllability and run on homogeneous nodes
   - TO: Systems are made up of third-party software components and run on heterogeneous nodes and networks
2. Solutions are dynamic and adaptable
   - FROM: Rule-based, always on, not adaptive to dynamic workloads or system conditions
   - TO: Data-driven solutions that can change with workload, system conditions, and architecture

Dependability Trends: Examples

1. Execution environments are changing

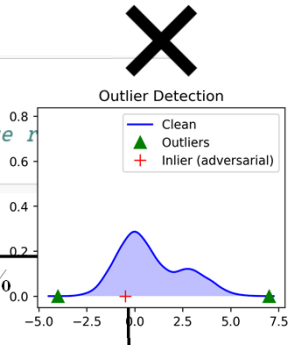


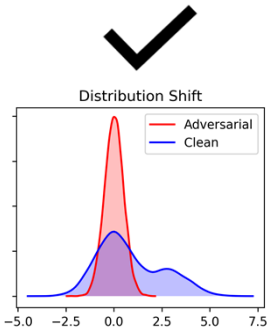


Dependability Trends: Examples

2. Static to dynamic

```
def square_root(x):
    assert x >= 0
    ...   # compute square r
    assert y * y == x
```

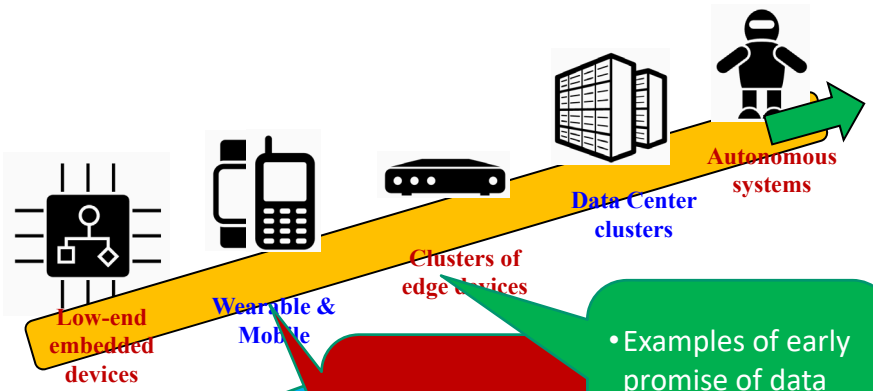




CPU load > 80%
OR
Memory occupancy > 90%
⇒ Migrate VM to different machine

[NeurIPS-20]

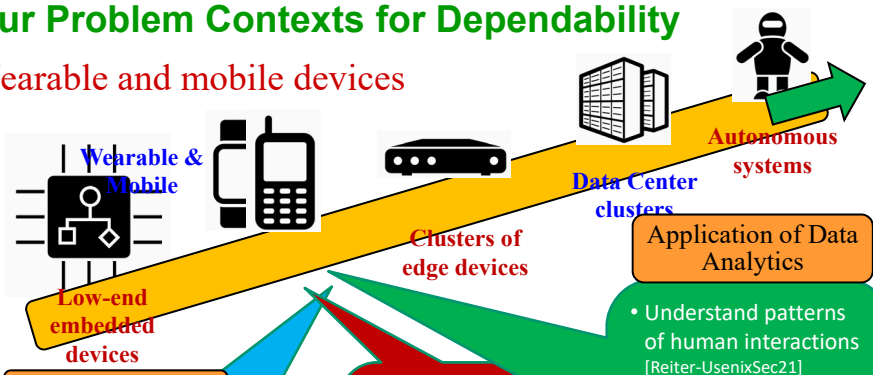
Framing of Dependability across the Stack



- What is the unique problem context for each level?
- What are the domain-specific requirements?
- Examples of early promise of data analytics for dependability

Our Problem Contexts for Dependability

Wearable and mobile devices



Problem Context

- New failure modes for wearables
- New software design patterns for wearable software
- Crowdsensing of information through mobile sensors

Requirements

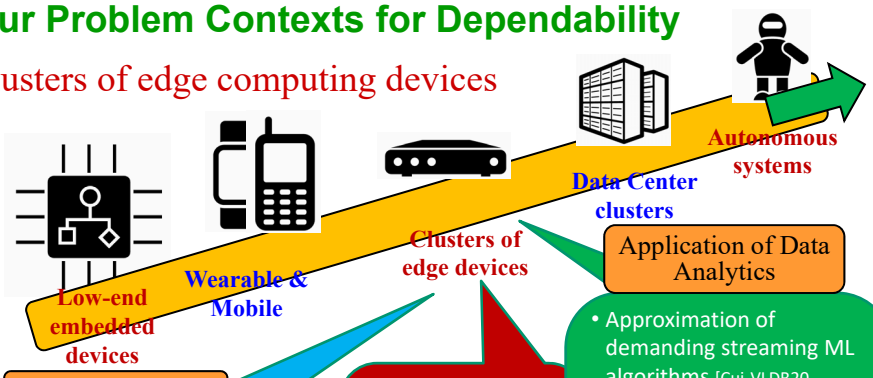
- Reliability from unreliable components
- Unobtrusive to human users

Application of Data Analytics

- Understand patterns of human interactions [Reiter-UsenixSec21]
- Prevent privacy leak [Provost-AAAI20, Wang-ICDE19]
- Reliable inference through unreliable crowdsensed data [Abdelzaher-IoTJ22, Bagchi-EWSN20]

Our Problem Contexts for Dependability

Clusters of edge computing devices



Problem Context

- Edge capacity is not fully elastic
- Individual edge device availability is highly variable
- Large geographic scale
- Large numbers of heterogeneous client devices

Requirements

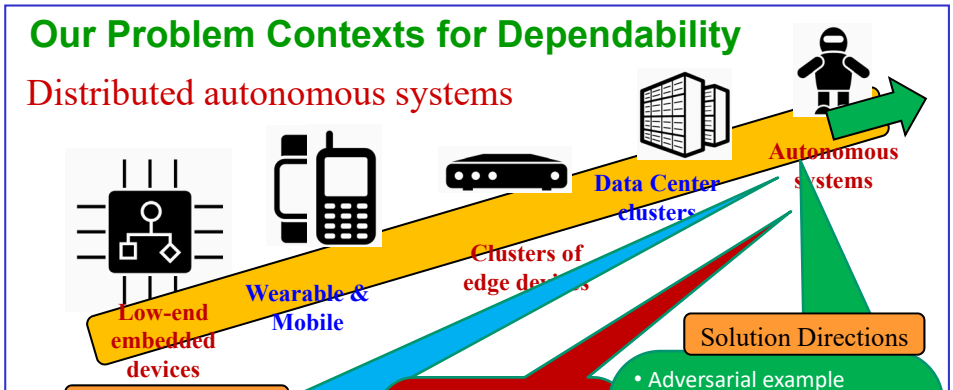
- Low latency
- Both unmanaged and managed edge devices

Application of Data Analytics

- Approximation of demanding streaming ML algorithms [Cui-VLDB20, E2train-NeurIPS19]
- Hybrid offloading between cloud, edge, and client for reliability [Bagchi-CACM20, Bagchi-SRDS22]
- Streaming inference under adversaries [Yogev-JACM22, Zhou-NeurIPS21]

Our Problem Contexts for Dependability

Distributed autonomous systems



Problem Context

- Secure distributed execution of ML codes on heterogeneous platforms
- Tolerating adversarial samples (video, social networks, …)
- Ensemble of models of varying trust

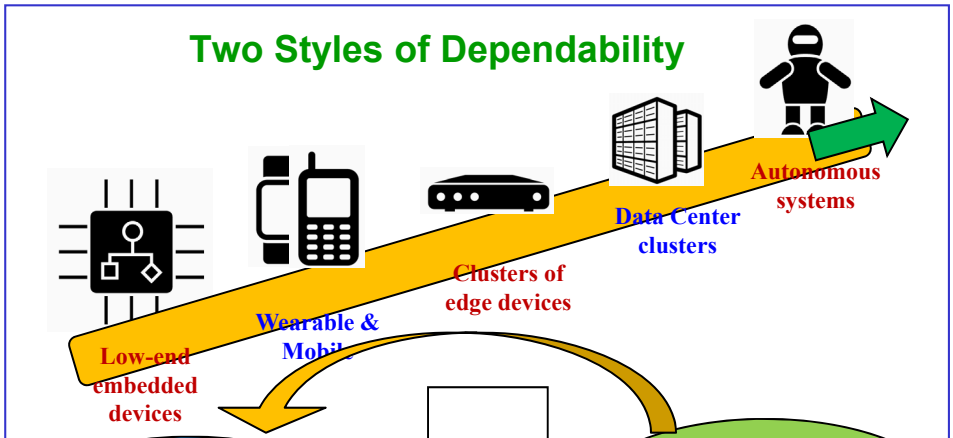
Requirements

- Many models are black box
- Noisy data sources
- Unreliable execution platforms

Solution Directions

- Adversarial example detection [Urtasun-CVPR20, Liu-NDSS19]
- Distributed situational awareness [Popa-UsenixSec21, Bagchi-AsiaCC23]
- Protecting against model and data poisoning [Gong-UsenixSec21, Houmansadr-S&P22]

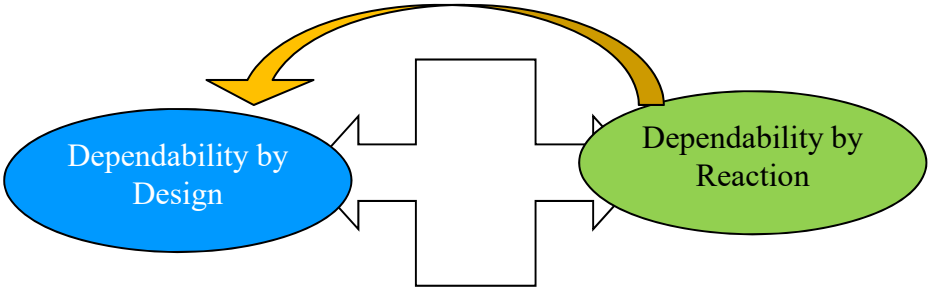
Two Styles of Dependability



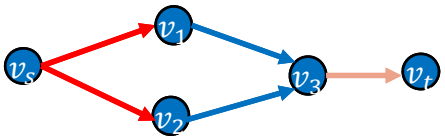
Dependability by Design

Dependability by Reaction

Two Styles of Dependability



Where to put security protections in a large-scale critical infrastructure system? *[Bagchi-S&P22, Bagchi-WSJ22]*



1. Learn the min-cut edge set in the attack graph
2. Invest in security protections on *all* edges in that set
3. Learn the appropriate level of investment on each edge

Roadmap

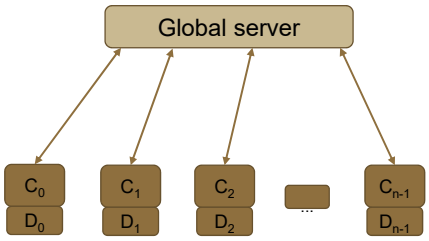
- Dependability across the stack
- **Security in Federated Learning:** [AsiaCCS-23, DSN-23]
- Privacy in Federated Learning: [CVPR-23, S&P-24]
- Takeaways and Open Problems

Fun tech quiz:
https://bit.ly/sbquiz23

Federated Learning: Background

- Client nodes communicate only with the server and not with each other.
- Clients only share their local model updates while the data remains private.
- Server aggregates the local model updates to update the global model, which is sent back to the clients.

$x(t) = A(x(t-1), local\_grads(t))$

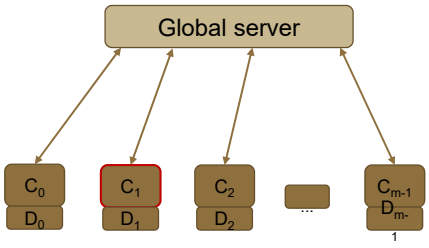


$local\_gradsi(t+1) = \nabla fi(x(t), Di)$

PURDUE UNIVERSITY

FL Vulnerabilities

- A malicious client can send faulty gradients to the server to throw it off from converging at the optima.
- The server is trusted to perform Byzantine-robust aggregation.



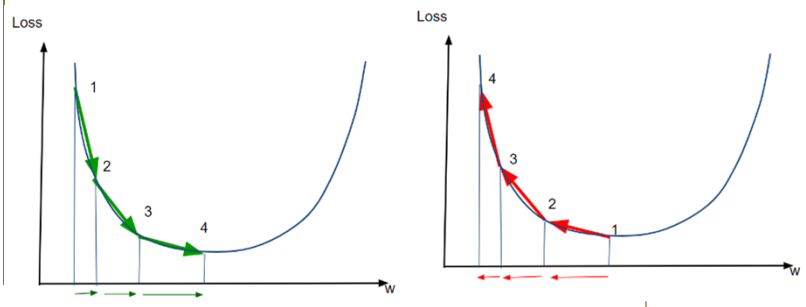
Data Poisoning vs Model Poisoning: Model poisoning attacks are more potent than data poisoning attacks.

Targeted vs Untargeted Attacks: Untargeted attacks affect all data samples and can cause more extensive damage compared to targeted attacks, which focus on specific data samples.

PURDUE UNIVERSITY

Directed Deviation Attack: SOTA Attack in FL

1. **Gradient Manipulation**: Attackers send gradient updates deviating from the local optima to throw off the system from learning an accurate global model.
2. **Optima Estimation**: Attackers estimate local optima direction using benign gradients, the accuracy of which depends on the threat model.
3. **Threat Model Dependence**: Precision of optima estimation varies with threat model - higher in white-box scenarios, lower in black-box scenarios.

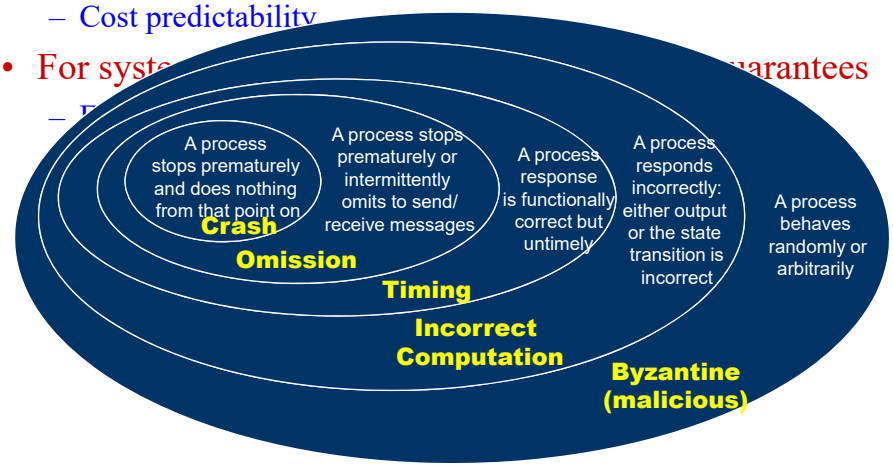


FL attack: Threat model

1. **Compromised Nodes**: Assume c out of m nodes are compromised by an adversary, enabling them to send malicious gradient updates.
2. **Access to Benign Gradients**: The adversary has access to the benign gradients of the compromised nodes in each learning round.
3. **Knowledge of Aggregation**: The adversary is aware of the aggregation technique used by the server.

What Does Dependability Have To Do With This?

- Predictability is an attribute of dependability
  - Performance predictability
  - Cost predictability
- For syste… …arantees
  - R…



FL Directed Deviation Attack

Fang Attack [USENIX Sec '20]: Computes a direction vector along the inverse of the average benign direction.

- Krum Attack: This attack ensures all malicious models are close to each other with small mutual distance, fooling the Krum aggregator to choose the poisoned model.

- Trim Attack: This attack samples model updates per parameter in a way that skews the distribution toward the malicious direction.

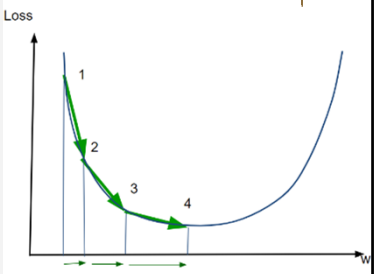
Shejwalkar Attack [NDSS '21]: Computes a perturbation (inverse unit) vector and scales it up before adding to the benign updates. The scaling factor is tuned depending on the dataset and the model used.

How Bad Does It Get With the Attack?

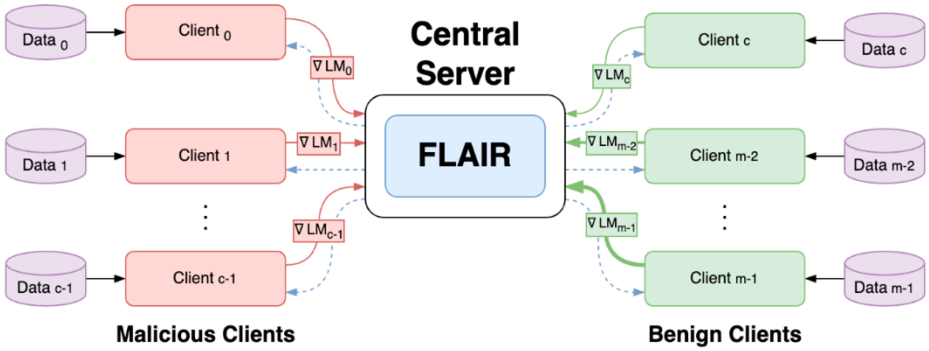
Dataset (Model)	AGR	No attack (A_θ)	Gradients of benign devices are known					Gradients of benign devices are unknown				
			AGR tailored (agr-updates)		AGR agnostic (updates-only)			AGR tailored (agr-only)		AGR agnostic (agnostic)		
						Our attacks					Our attacks	
			Fang [17]	Ours	LIE [4]	Min-Max	Min-Sum	Fang [17]	Ours	LIE [4]	Min-Max	Min-Sum
CIFAR10 (Alexnet)	Krum	53.5	21.8	**43.6**	9.9	17.4	**30.1**	19.8	**43.1**	18.1	13.7	**30.2**
	MKrum	67.6	12.6	**36.8**	20.5	27.8	**30.8**	11.2	**35.3**	19.7	**31.7**	30.4
	Bulyan	66.9	12.3	**45.6**	33.8	35.5	**44.5**	11.8	**34.6**	30.0	40.6	**41.1**
	TrMean	67.7	15.8	**45.8**	22.8	**41.6**	33.5	12.9	**45.0**	19.4	**38.7**	27.9
	Median	65.5	12.9	**40.9**	20.7	34.7	**39.6**	12.6	**39.1**	19.7	34.1	**39.5**
	AFA	66.8	7.0	**47.0**	5.9	**31.5**	16.9	6.1	**46.8**	5.5	**22.2**	16.0
	FangTrmean	66.8	8.9	**56.3**	6.5	**42.9**	21.5	8.5	**56.0**	6.3	**42.1**	19.9

FLAIR: Key defense idea

1. **Smooth Loss Landscape**: FLAIR assumes a learning task with a smooth loss landscape around the current state of the model.
2. **Small Learning Rate**: A well-chosen small learning rate should ensure that a large number of parameter gradients do not flip their direction with large magnitudes in a benign setting.
3. **Gradient Inertia**: The model maintains some degree of inertia in the parameter gradients, meaning that large changes in direction are unusual.
4. **Detecting Attacks**: Large collective flips in some gradient vectors are indicative of an attack.

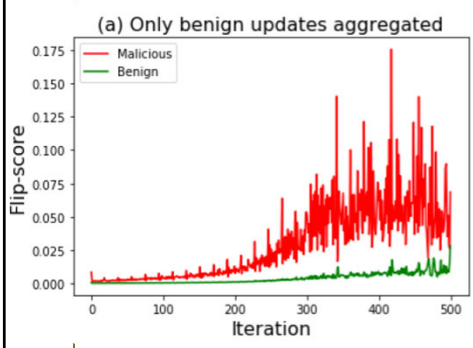


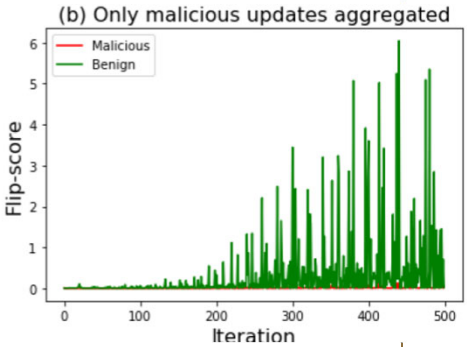
FL with FLAIR Aggregation



FLAIR: Key Defense Idea

Detecting malicious updates from the flip-score





Macro Results

Table 1: Impact of Directed Deviation Model Poisoning Attacks: This table presents the test accuracy for directed deviation model poisoning attacks (Full-Krum; Full-Trim) on various datasets with a c/m ratio of 0.2. For the Shakespeare dataset, we report the test loss instead. The results highlight the damaging impact of Full-Trim attacks on mean-like aggregations (FedSGD, Trimmed mean, Median) and Full-Krum attacks on Krum-like aggregations (Krum, Bulyan). While existing defenses such as FABA, FoolsGold, and FLTrust show mixed results, our proposed method, FLAIR, consistently outperforms in all cases.

Attack	Defense	Metrics			
		MNIST+ DNN	CIFAR-10+ ResNet-18	Shakespeare+ GRU	FEMNIST+ DNN
None	FedSGD	92.45	**71.17**	**1.62**	83.60
	FLAIR	**92.52**	66.92	1.64	83.58
	FABA	91.77	69.94	1.76	82.69
	FoolsGold	91.20	70.71	1.63	**83.80**
	FLTrust	87.70	68.08	**1.62**	82.72
Full-Krum	FedSGD	82.97	39.68	1.62	29.87
	Krum	8.92	9.81	11.98	5.62
	Bulyan	10.14	13.24	9.23	9.91
	FLAIR	**87.73**	61.26	1.64	**80.19**
	FABA	86.99	55.96	1.75	55.61
	FoolsGold	47.12	42.28	**1.63**	0.07
	FLTrust	82.50	**65.25**	1.67	79.53
Full-Trim	FedSGD	65.25	47.32	1.74	32.34
	Trim	36.36	55.25	3.28	13.03
	Median	28.37	50.54	3.30	45.6
	FLAIR	90.55	67.65	1.66	82.51
	FABA	**91.84**	67.31	**1.64**	79.66
	FoolsGold	91.61	**69.24**	1.66	**83.09**
	FLTrust	34.20	64.23	1.68	79.28

FLAIR is among the top-2 performers across datasets and attack types

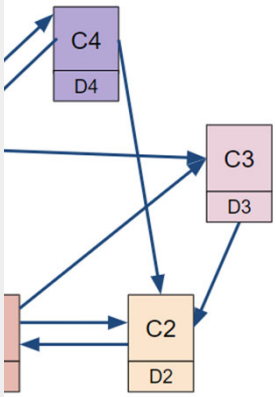
Macro Results

Table 3: Comparison of test accuracies for FLAIR under SHEJWALKAR attack, with and without the knowledge of the aggregator. The performance of FLAIR is compared against the baseline FedSGD model on two datasets: MNIST and CIFAR-10. The table illustrates FLAIR's robustness in the face of attacks and its ability to maintain high accuracy rates.

AGR	AGR-knowledge	MNIST	CIFAR-10
FedSGD	No	10.10	10.00
	Yes	10.09	10.00
FLAIR	No	92.25	69.35
	Yes	92.83	69.98

Open Problem – Security in P2PL

1. **Decentralized Learning**: Clients collaborate among themselves without relying on a single server.
2. **Dissensus Problem**: Without central coordination of the learning process, there can be disagreements among nodes about state of the global model slowing down learning
3. **Difficult Detection of Malicious Activity**: Each node only has a limited view of the network, making it harder to identify malicious activity.
4. **Increased Vulnerability to Attacks**: An adversary could potentially compromise multiple nodes and use them to introduce malicious updates.



Takeaways

- FLAIR is a secure parameter server for federated learning, robust to *any* untargeted model poisoning attack.
- We define malicious behavior using a metric, flip-score, which when too high or too low, captures attacks that try to divert the global model away from convergence.
- FLAIR uses a stateful algorithm to allocate reputation scores to the participating clients to lower the contribution of maliciously behaving clients.
- We evaluate the benefits of FLAIR compared to the fundamental FL aggregation FedSGD and state-of-the-art defenses, Krum, Bulyan, FABA, FoolsGold, and FLTrust.
- FLAIR is robust even against a full knowledge untargeted model poisoning adversary that were recently found to be most damaging against FL.

Roadmap

- Dependability across the stack
- **Security in Federated Learning:** [AsiaCCS-23, DSN-23]
- **Privacy in Federated Learning:** [CVPR-23, S&P-24]
- **Takeaways and Open Problems**

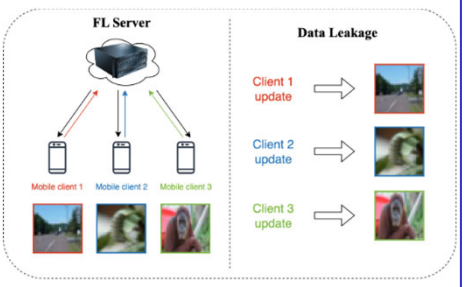
Fun tech quiz:
https://bit.ly/sbquiz23

Federated learning (FL) for privacy preservation

- Federated learning was introduced as a privacy-preserving technology to allow decentralized training of models on client data
  - Mobile devices can train models now (computational power)
    - Model training updates can be computed by clients
  - Only the updates (gradients) need to be sent to the server

FL privacy attack - data leakage

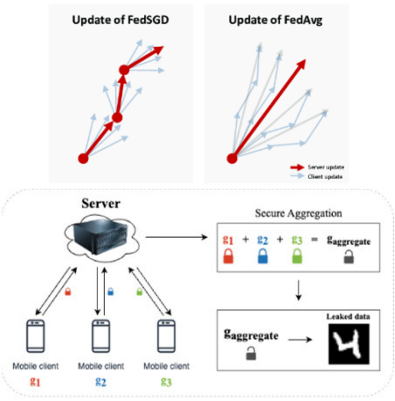
- "Privacy is enhanced by the ephemeral and focused nature of the [Federated Learning] updates." [1]
- Privacy-preservation premise of FL holds only if the *client updates cannot be used to recover information about the data*.
  - Strongest FL privacy attacks are data leakage/reconstruction attacks



FedAVG and Secure aggregation (SA): Difficult to Attack, but are they Privacy Preserving?

- FedSGD vs. FedAVG
  - FedSGD: clients have a single training iteration before sending the update to server
  - FedAVG: clients have multiple local training iterations/epochs before sending the update to server
    - Server cannot see intermediate model updates, only the initial and final state
- Secure aggregation (SA)
  - Server can only see the aggregated model update for all clients

The **combination of FedAVG + SA** is particularly difficult to attack.

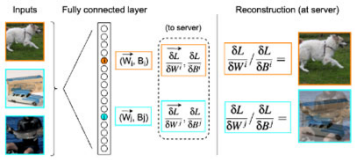


Basic Idea of Linear Layer Leakage for Data Reconstruction

- Inputs to a fully-connected (linear) layer can be directly leaked through the gradients of the layer. This can be the input images if layer placed at the start.

$$x = \frac{\delta L}{\delta W^i} / \frac{\delta L}{\delta B^i}$$

- This requires a single input image to activate a neuron in the layer
  - If multiple images activate a single neuron, the reconstructions fail
- Robbing the Fed[1] (RtF) built upon this idea and proposed a more efficient linear layer leakage approach
  - Higher leakage rate
  - Better scalability (batch size/secure aggregation)



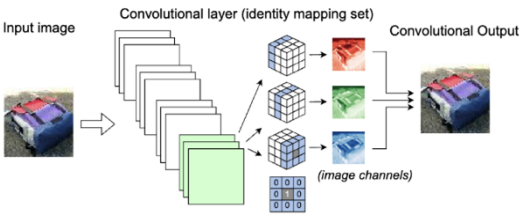
[1]Fowl et. al., "*Robbing the Fed: Directly Obtaining Private Data in Federated Learning with Modified Models*". ICLR, 2022.

Problems with Linear Layer Leakage Scalability

- As the total number of input images increases, the size of the FC layer must increase to accommodate this, as multiple image activations per neuron results in reconstruction failure.
  - Batch size 64, 1 client, 256 FC layer size. On average 4 neurons per image avoids collision.
  - With 100 clients instead, there is on average 25 images per neuron.
- Model size overhead becomes very large as the FC layer increases in size.
- Weight scaling in FedAVG causes precision problems with large FC layer.
  - Leakage rate decreases and reconstructed images become much lower quality

LOKI: Convolutional Layer Prior to FC Layers to Split Scaling

- Convolutional layer placed in front of FC leakage layers.
- Use "identity mapping sets" to push the input image to the FC layer.
  - First three kernels of the convolutional layer propagate image forward exactly as it is.
- Clients are sent *different* identity mapping sets



- Split scaling:
  - Increase the number of convolutional kernels based on number of clients
  - Increase FC layer size based on local dataset size/batch size

LOKI: Scalable Leakage for FedAVG and SA

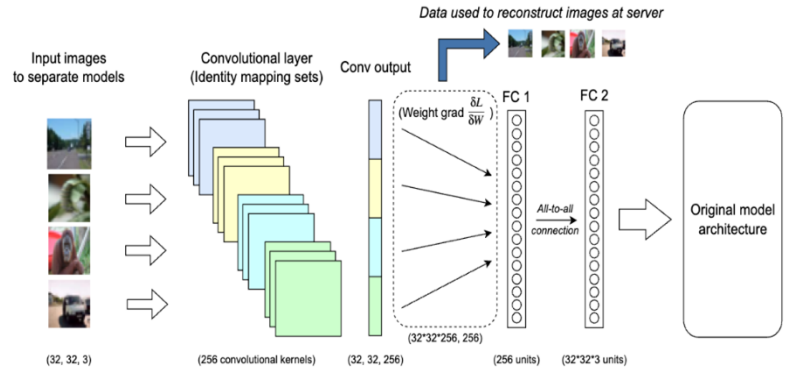
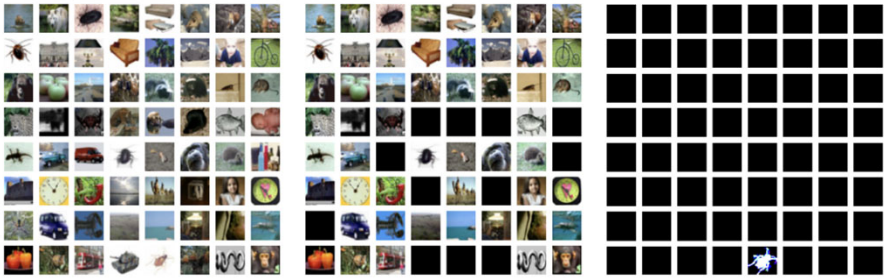


Figure 2. The inserted attack module of a convolutional layer and two FC layers. Images are leaked using the gradients of the weight parameters connecting the convolutional output to the first FC layer. Weight gradient de-aggregation is done through separate identity mapping sets, indicated by the different color values in the convolutional kernels. In the aggregate update, the weight gradients of different clients are separated and used to leak the images.

Data Reconstruction Examples for LOKI



(a) Ground truth local dataset (b) LOKI reconstruction (c) Robbing the Fed [17] reconstruction

Figure 5. LOKI and Robbing the Fed (RtF) reconstructions on CIFAR-100 for 1 client out of 100 in FedAVG aggregation. Clients use 5 epochs with 8 local iterations and a local mini-batch size 8. Empty boxes indicate images were not reconstructed because multiple images activated a neuron. In the case of RtF, nearly all empty boxes occur due to precision problems causing the computed gradient to be zero rather than multiple activations.

Leakage Rate and Scalability in FedAVG

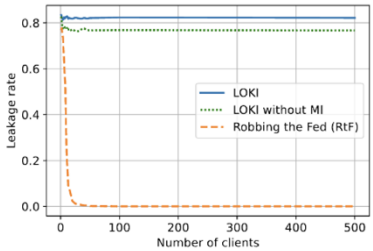
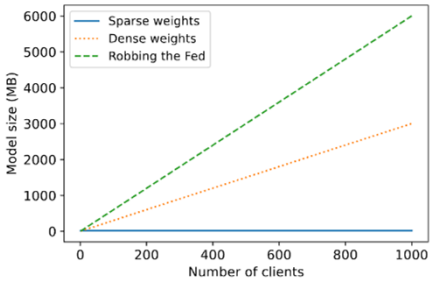


Figure 6. Leakage rate of LOKI with and without model inconsist (MI) and Robbing the Fed (RtF) for a number of clients between 1-50 CIFAR-100. Clients train with 8 local iterations of mini-batch size 8.

Dataset	Metrics	LOKI	RtF [17] + MI [20]
CIFAR-100	Leaked imgs	5290	50
	Total imgs	6400	6400
	Leakage rate	**82.66%**	**0.78%**
Tiny ImageNet	Leaked imgs	5202	49
	Total imgs	6400	6400
	Leakage rate	**81.28%**	**0.77%**
MNIST	Leaked imgs	4907	49
	Total imgs	6400	6400
	Leakage rate	**76.67%**	**0.77%**

TABLE 2. LEAKAGE RATE FOR FEDAVG AGGREGATED UPDATE WITH 100 PARTICIPATING CLIENTS. $\alpha = 1e-4$, $CSF = 100$ AND 5 LOCAL EPOCHS OF 8 ITERATIONS OF MINI BATCH SIZE 8 USED.

Model Size Reduction through Sparsity



	Model size (MB)	Sparse attack	Robbing the Fed
MobileNet v3 (L)	20.9161	87.65%	28690.76%
ResNet-18	44.5919	41.11%	13457.57%
ResNet-50	97.4923	18.80%	6155.35%
Inception v3	103.6120	17.69%	5791.79%
VGG-11	506.8334	3.62%	1184.02%

Table 1. Model size overhead added from the attacks with 1000 clients and a batch size of 64 on Tiny ImageNet compared to vision models. The overhead added by the sparse representation attack (18.33MB) is significantly smaller than Robbing the Fed (6000.99MB) and achieves the same leakage rate.

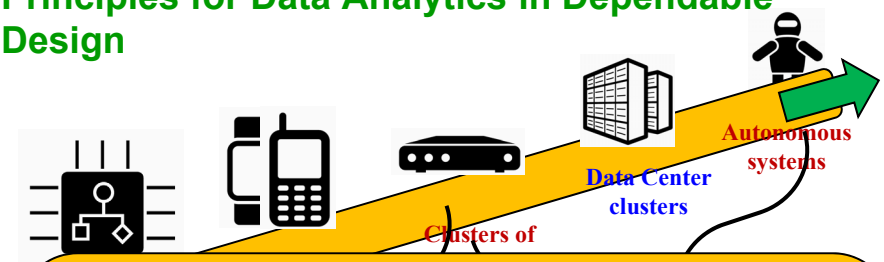
Takeaways

- In Federated Learning, even secure aggregation in FedAVG cannot preserve privacy of client data.
- Our attack LOKI: Malicious server sends customized models to clients.
- Key design idea: Send customized convolutional kernels to each client, an identity mapping set, that separates the weight gradients from the clients despite the use of secure aggregation.
- The server then uses these weight gradients to reconstruct the original data points.
- We are the first to achieve a privacy attack in FL with FedAVG that scales well with the size of the local dataset and the number of clients.

Open Problems for Security in ML

- How to handle data dependent behavior
  - Control flow may be affected by content of request
  - What is benign and what is malicious may change based on this data dependence
- How to refresh models for detection or identification
  - When workload changes or for an adaptive attacker
  - Model update incrementally through online learning
  - How to leverage sparse human input
- How to secure peer-to-peer learning
  - Particularly vulnerable to well-placed adversary nodes
  - Can we take advantage of "wisdom of the crowds"

Principles for Data Analytics in Dependable Design



Data Analytics can help

1. Only in some cases
2. Model must be simple
3. Model inference and search overheads must be considered
4. White box nature is often a desired element
5. Model must be adaptable to dynamic conditions

How You Can Get Involved?

1. Visit us at Purdue's ***Assured Autonomy Innovation Institute (A2I2)***
   - BTech/MTech students as 2-month interns (remote)
   - PhD students, Post-doctoral scholars, Faculty
   - Anywhere from 1-9 months possible
   - Immigration paperwork and local arrangements handled by our Institute staff
2. Jointly pursue funding with NSF and DST
   - For organizing workshops
   - For researcher visits
3. Co-author perspective articles that will go on to define topics within ML and dependability

It Takes a Village

- **Purdue**: [*Faculty*] Qiang Qiu, Murat Kocaoglu, Somali Chaterji [*Students*] Atul Sharma, Joshua Zhao, Wei Chen
- **Wisconsin Madison**: Yin Li
- **EPFL:** Mathias Payer
- **USC**: Ramesh Govindan, Salman Avestimehr
- **Illinois:** Tarek Abdelzaher
- **ETH:** **Ana Klimovic**
- **Microsoft Research:** Sameh Elnikety
- **Army Research Lab:** Noah Weston, Brian Henz, Priya Narayanan, Dan Cassenti
- **Adobe Labs:** Vishy Swaminathan Haoliang Wang, Subrata Mitra
- **Google** (Infrastructure Team): Rama Govindaraju, Liqun Cheng

Thanks To

CPS, SaTC, CCRI, SPX

NIAID (R01)

Army Research Lab
A2I2 Institute

Faculty Award

Backup

Our Problem Contexts for Dependability