THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

CCF 物联网专委会走进山东大学（青岛）青年论坛

Blockchain Made Wireless

—— Extend Blockchain to Wireless Networks

Minghui Xu（徐明辉）

Department of Computer Science, SEAS
The George Washington University
Email: mhxu@gwu.edu

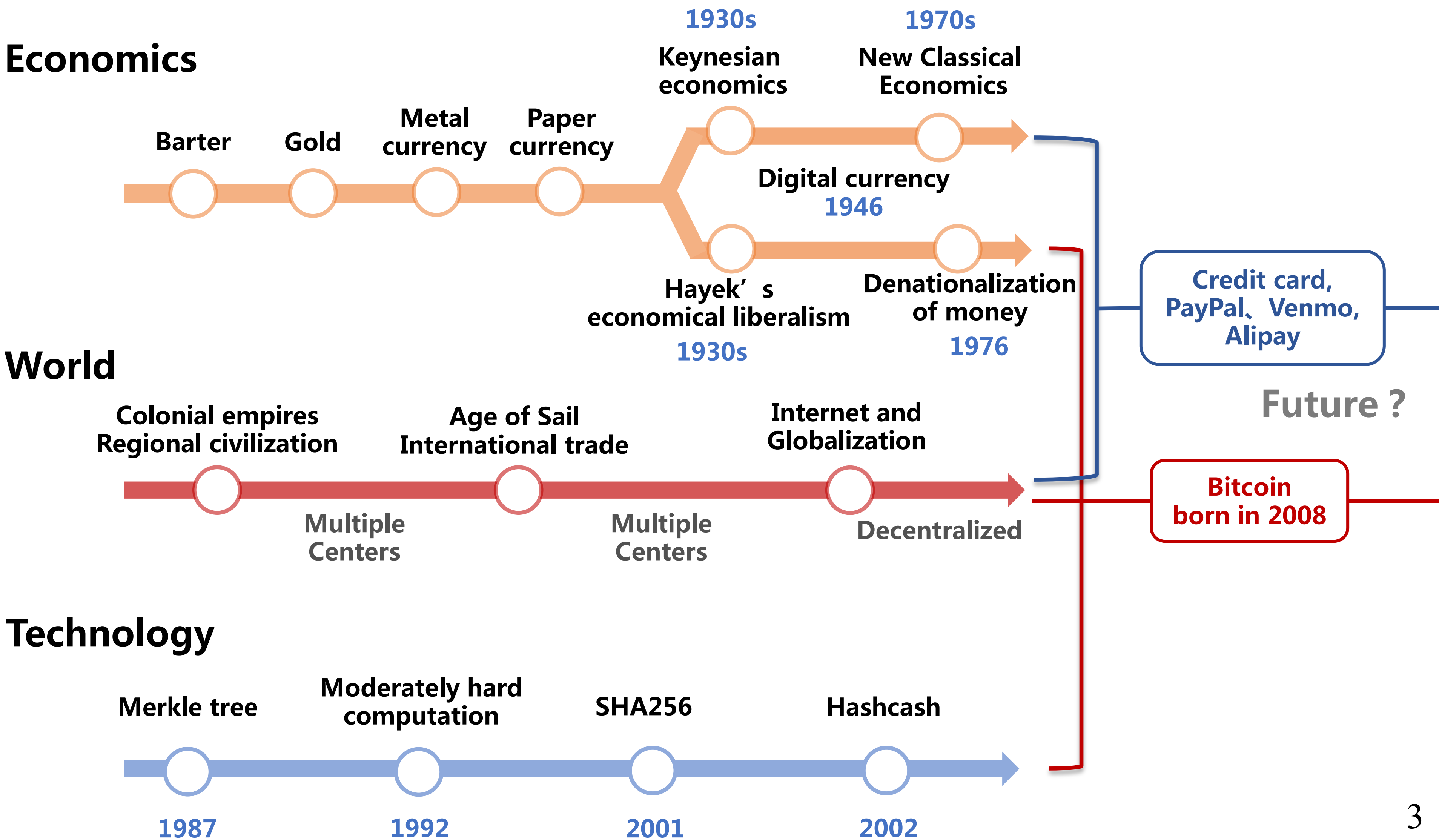
May 8, 2021

Outline

- **Brief Introduction to Blockchain**
  - Technical view
  - Historical view
- **Motivations of blockchain made wireless**
  - Wired and wireless networks tell different stories
  - Overview of related works
- **Blockchain Made Wireless**
  - BLOWN: A Blockchain Protocol for Wireless Networks under Adversarial SINR
  - wChain: A Fast Fault-Tolerant Blockchain Protocol for Multi-hop Wireless Networks
- **Conclusions and Future Directions**

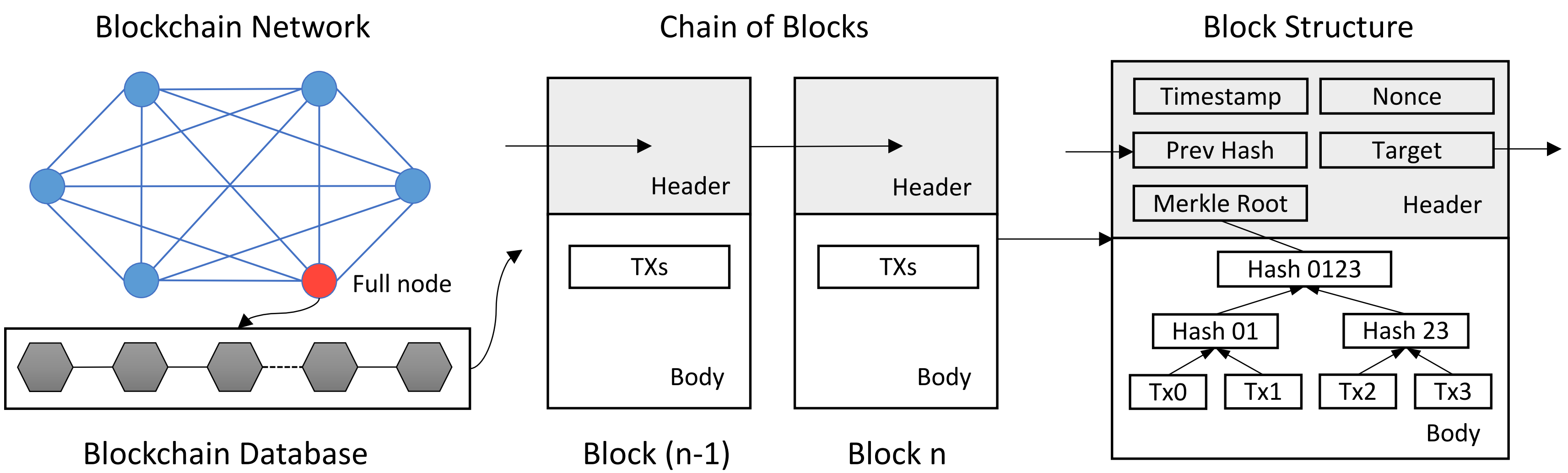
Historical view

Prehistorical Time of Blockchain



Technical view

Bitcoin as an example:



Hash function: digital fingerprint, heavily used in blockchain
Consensus algorithm, e.g., Proof-of-Work (PoW) in bitcoin:
crack hash puzzle "$H(Timestamp||Prev\ Hash||Merkle\ Root||Nonce) < Target$"

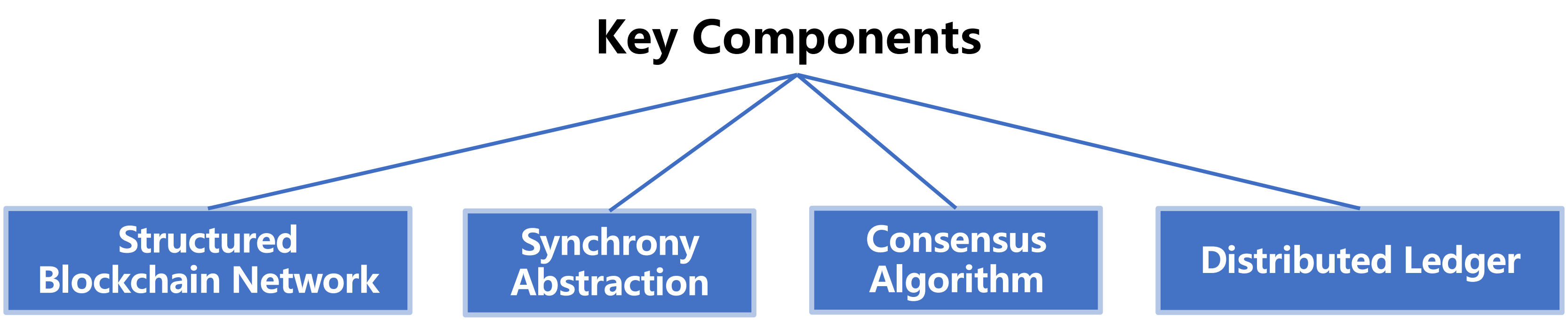
Salient properties of blockchain:

- Immutability
- Traceability
- Trustlessness

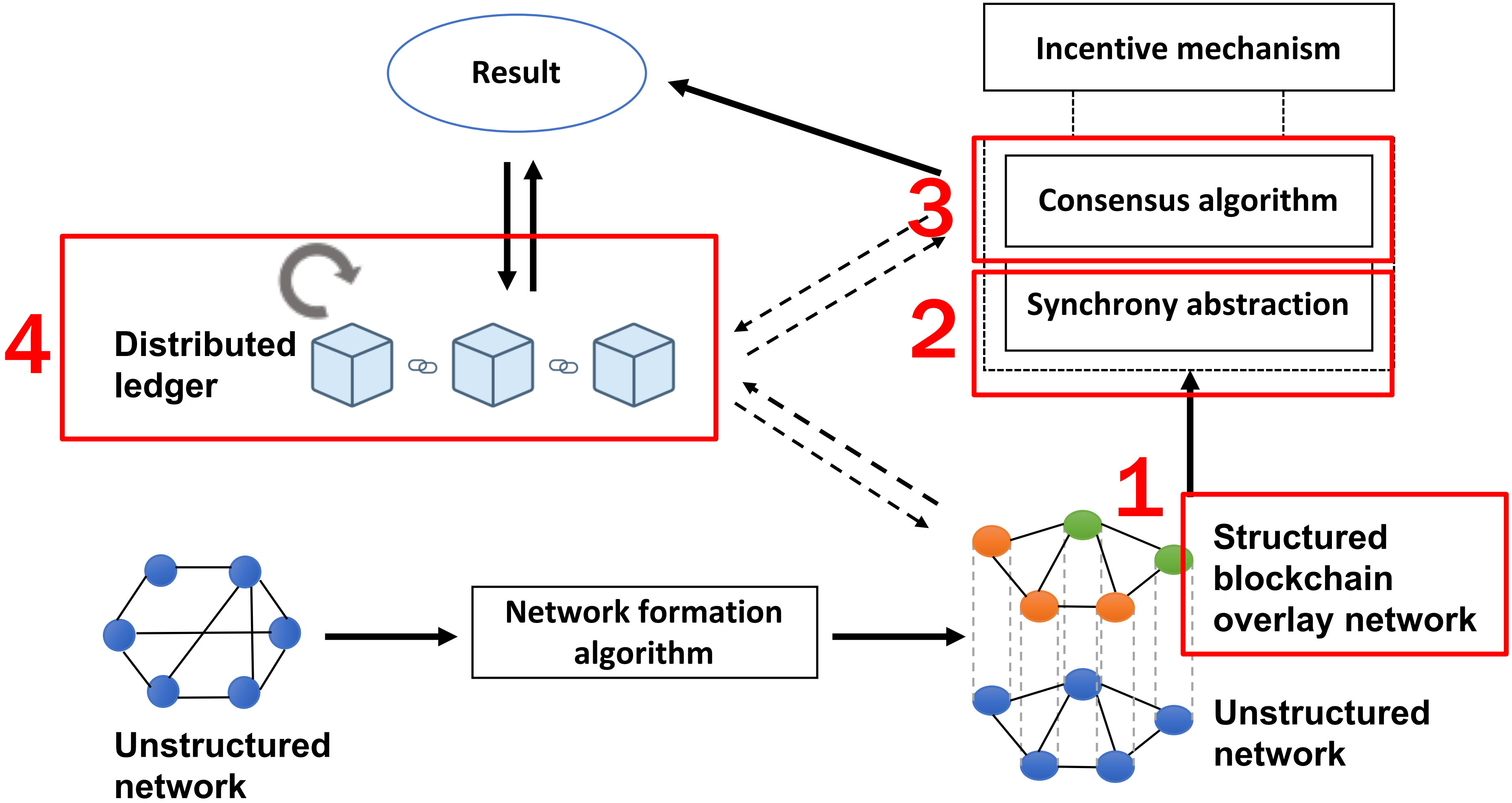
Technical view

- **(General) Blockchain:** Blockchain is a shared, distributed **ledger** maintained by peer nodes in a decentralized network.

- **(Specific) Blockchain:** A chained data structure composed of a number of time-series data blocks that are tamper-proof by cryptographic methods.

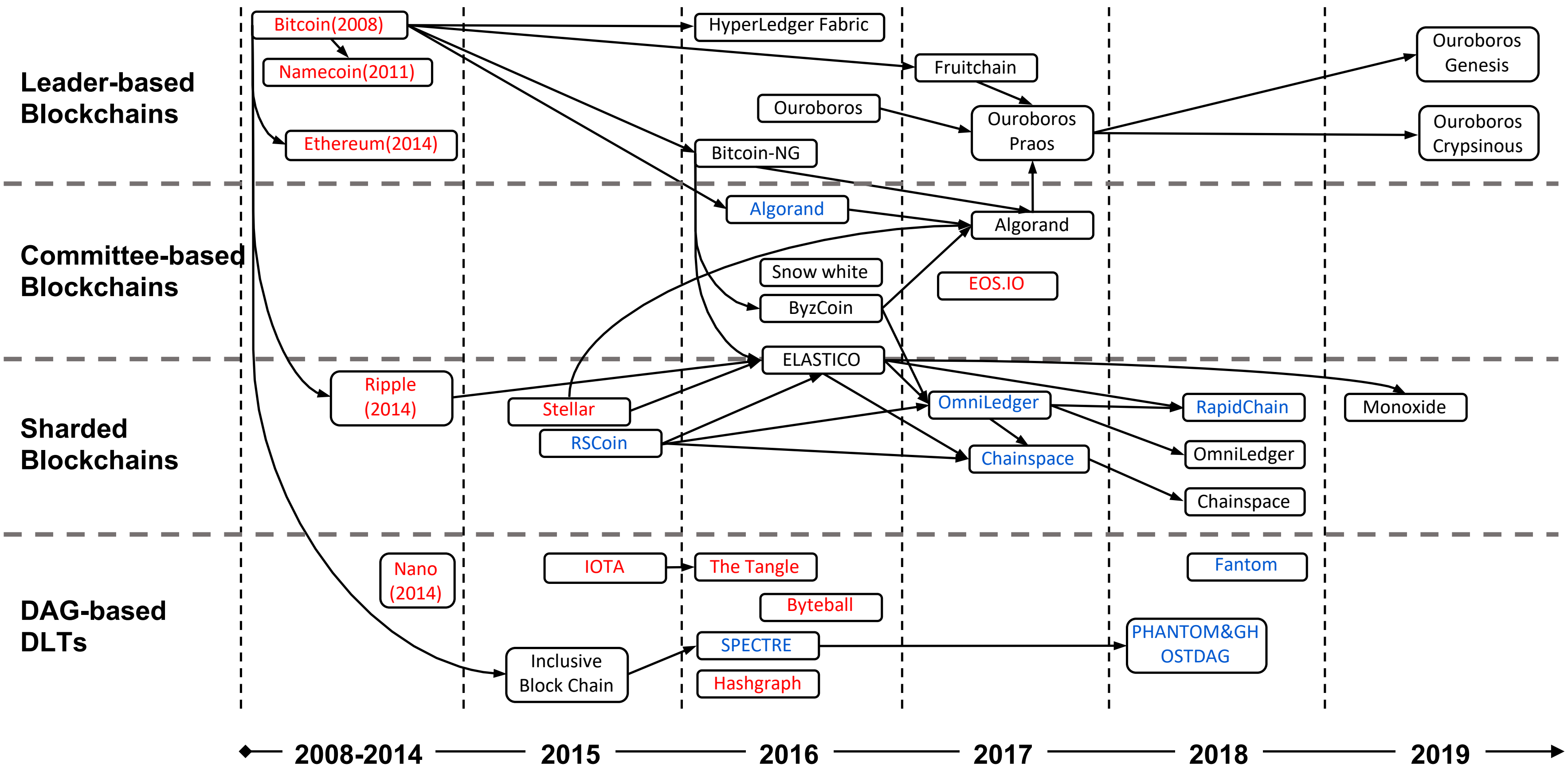


Technical view



This framework is part of the second on-going work. Minghui Xu and Chunchi Liu have equal contributions to this figure.

Overview of related works



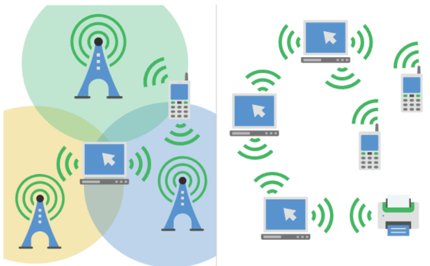
Popular blockchain protocols mostly target for wired networks

Wired and wireless tell different stories

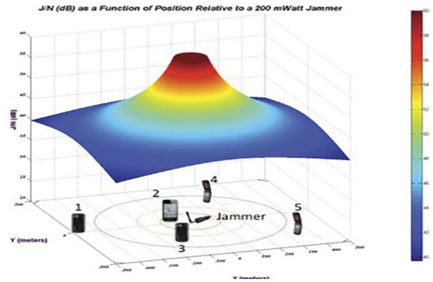
Aspect	Wired Network	Wireless Network
Transmission	unicast and multicast	Broadcast is heavily used
Contention	CSMA/CD	CSMA/CA; Interference adversarial jamming
Topology	Usually, static	Usually, dynamic
Resource	Ipv4 and Ipv6	Limited spectrum resource
Signal	Through cables	Fading channel and obstacles
Security and privacy	Protected by firewall	Signal spread in the air

Consensus in air

Efficiency and security of blockchain protocol



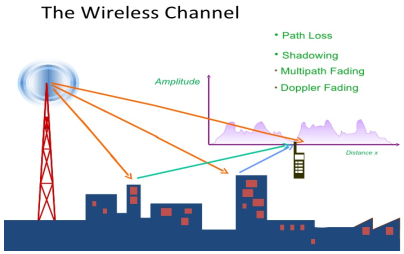
Broadcast transmission



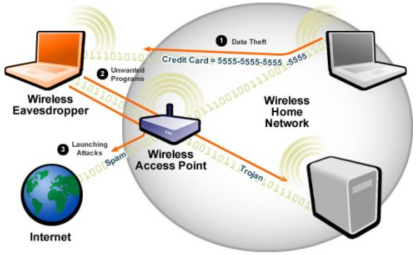
Interference and Jamming

Mobility

Limited spectrum resource

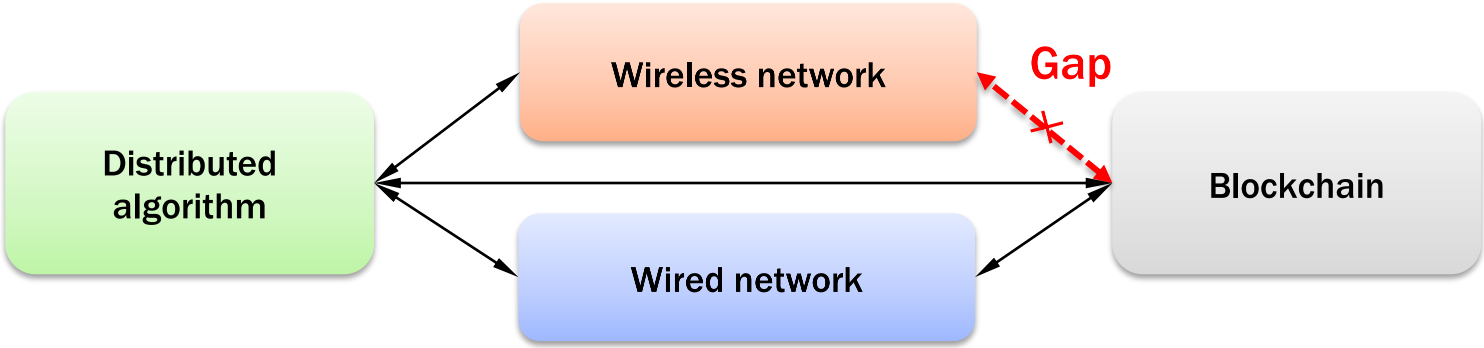


Fading channel



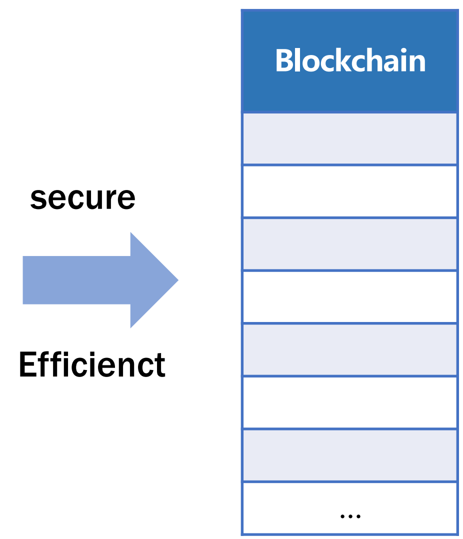
Signal in the air

Overview of related works



Hard work has been done in recent 20 years to design various specific distributed algorithm in wirelss networks, including the leader election algorithm, consensus algorithm, Fault tolerant (FT) and Byzatine Fault Tolerant (BFT) consensus algorithm.

Research	Leader election	Consensus		
		Simple	FT	BFT
ON 02	Yes	-	-	-
AGF+ 04	Yes	-	-	-
CDG+ 05	-	Yes	Yes	-
AZ 12	-	Yes	Yes	-
MNC 13	Yes	-	-	Yes
Newport 14	Yes	Yes	-	-
RN 18	-	-	Yes	-
PNL 19	Yes	Yes	Yes	-



Still need more work towards a blockchain protocol

1. BLOWN: A Blockchain Protocol for Wireless Networks under Adversarial SINR（Major Revision, TMC）

2. wChain: A Fast Fault-Tolerant Blockchain Protocol for Multi-hop Wireless Networks (TWC 21)

Contributions of BLOWN

BLOWN (BLOckchain protocol for Wireless Networks)

Contributions:

- To the best of our knowledge, BLOWN is the first protocol that is particularly designed for single hop wireless networks under a realistic adversarial SINR model.
- The communication features of wireless networks are utilized to develop a novel, general Proof-of-Channel (PoC) consensus protocol, which leverages the natural properties of wireless networks, including broadcast communication and channel competition.
- We develop a UC-style protocol for BLOWN and formally prove BLOWN's persistence and liveness properties by showing that it satisfies concrete chain growth, common prefix, chain quality properties. Finally, simulations are conducted to demonstrate our theoretical analysis.

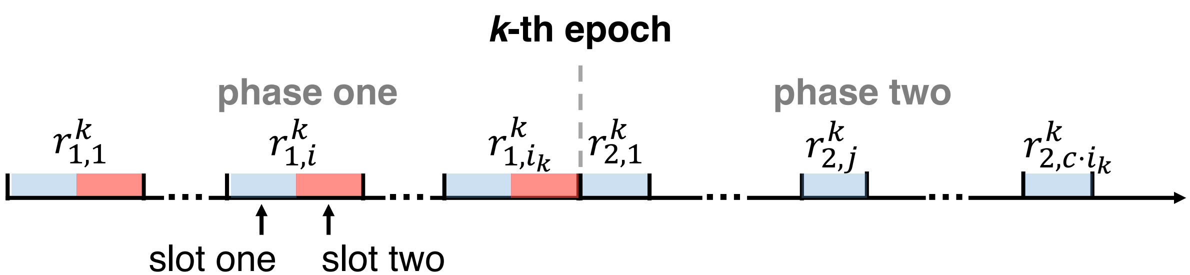
Model and Assumption

Network: a set V of N nodes arbitrarily deployed in a communication space; nodes are in a single hop wireless network

Interference: Signal-to-Interference-plus-Noise-Ratio (SINR) model

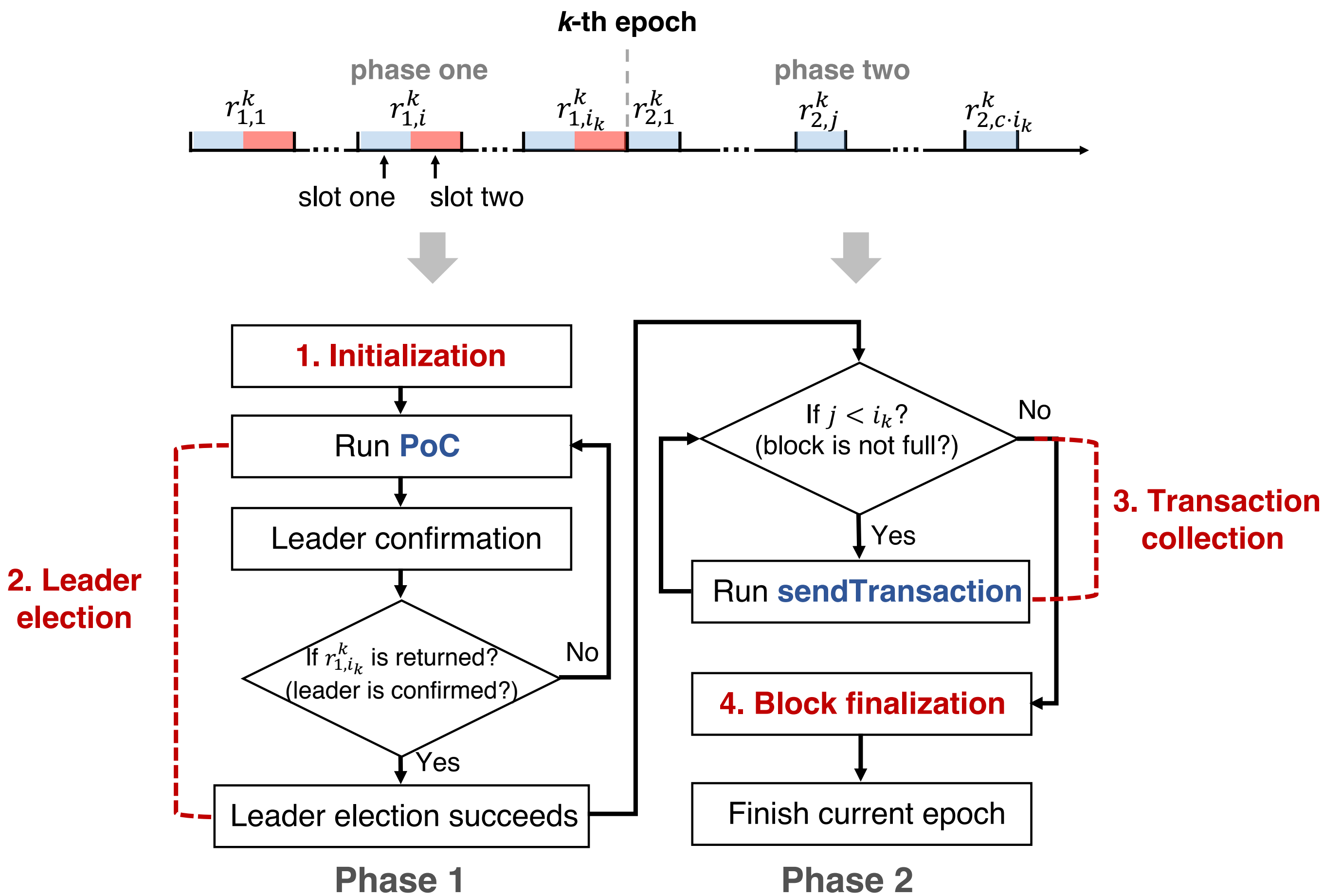
$$SINR(u,v) = \frac{\mathcal{S}}{\mathcal{N} + \mathcal{I}} = \frac{P_u \cdot d(u,v)^{-\alpha}}{\mathcal{N} + \sum_{w \in S \setminus \{u\}} P_w \cdot d(w,v)^{-\alpha}} \geq \beta$$

Synchrony: synchronized network with Epoch-based execution. At each epoch, no more than one block can be generated.



Adversary (A): A can make noise for any honest node at any time to launch jamming attacks. However, to leave a chance for an honest node to communicate, A is $(1-\epsilon, T)$-bounded at any time interval I of length T; can create different identities to launch sybil attack, but only controls less than 50% wealth (coins) of the entire network

BLOWN Protocol Overview



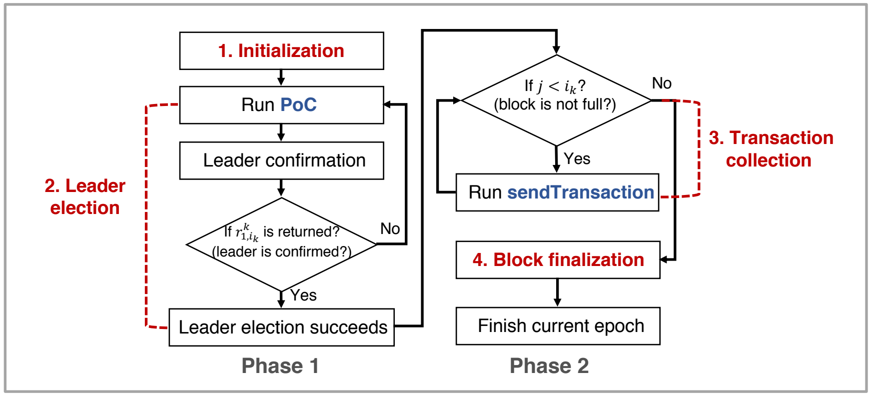
Protocol in Detail – Phase 1

Phase 1: Initialization + Leader election

Algorithm 3 BLOWN P_1 protocol

▷ Initialization
$h_v, \pi_v, l_v =$ *Sortition*$(sk_v, \text{seed}||\text{role}, \tau, w_v, W)$
$p_v = \hat{p}, c_v = 0, T_v = 1, i = 1, l'_v = l_v$
▷ Leader election
while TRUE **do**
  if $l_v > 0$ **then** ▷ As a potential leader
    ▷ slot one of $r^k_{1,i}$
    run PoC
    ▷ slot two of $r^k_{1,i}$
    if v sends a message in slot one **then**
      v listens on the channel
      if v senses an idle channel **then**
        return $r^k_{1,i_k} (i_k = i)$ ▷ run P_2 as a leader
    else
      v sends $m \leftarrow MSG(r^k_{1,i}, l_v)$
  else ▷ As a follower
    ▷ slot one of $r^k_{1,i}$
    v listens on the channel to receive a message
    ▷ slot two of $r^k_{1,i}$
    if v receives m from u and has $\mathcal{I} + \mathcal{N} < \theta$ in slot one **then**
      if v senses an idle channel **then**
        v recognize u as the leader
        return $r^k_{1,i_k} (i_k = i)$ ▷ run P_2 as a follower
    else
      v sends $m \leftarrow MSG(r^k_{1,i}, l_v)$
  $i = i + 1$

Phase 1



Algorithm 2 PoC subroutine

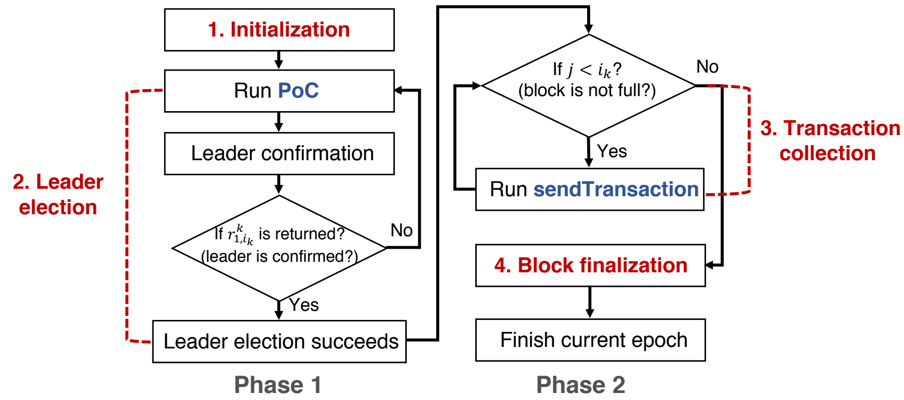
if v decides to send a message based on p_v **then**
  v sends $m \leftarrow MSG(r^k_{1,i}, l_v)$
else
  if channel is idle **then**
    $p_v = \min\{(1+\gamma)p_v, \hat{p}\}$
    $T_v = \max\{1, T_v - 1\}$
  else
    if receives a message m **then**
      $p_v = (1+\gamma)^{-1} p_v$
      $l_v = l_v - 1$
$c_v = c_v + 1$
if $c_v \geq T_v$ **then**
  $c_v = 1$
  if there is no idle rounds in the past T_v rounds **then**
    $p_v = (1+\gamma)^{-1} p_v,$
    $T_v = T_v + 2$

PoC Subroutine

Protocol in Detail – Phase 1

Phase 2:

- **Transaction collection**
- **Block finalization**



Algorithm 5 BLOWN P_2 protocol

▷ Transaction collection
while $j < c \cdot i_k$ **do**
  if $l_v > 1$ **then** ▷ As a leader
    v listens on the channel to receive a message
    if receives $m_T.\mathrm{tx} \neq \perp$ **then**
      $txp_v[j] = m_T.tx$
  else ▷ As a follower
    run **sendTransaction**
  $j = j + 1$
▷ Block finalization
if $j = c \cdot i_k$ **then**
  if $l_v > 0$ **then** ▷ As a leader
    $B_v^k \leftarrow Packup(txp_v)$
    $BC_v^k \leftarrow append(\mathrm{BC}_v^{k-1}, B_v^k)$
    send $m_B \leftarrow MSGB(BC_v^k, B_v^k, r_{2,j}^k, l_v, \mathrm{role}, w_v, h_v, \pi_v, l'_v)$
  else ▷ As a follower
    if $\mathrm{Vrfy}(m_B) \neq \perp \; \&\& m_B.BH_u^{k-1} = BH_v^{k-1}$ **then**
      $append(\mathrm{BC}_v^{k-1}, m_B.B_u^k)$

Algorithm 4 sendTransaction subroutine

if v decides to send a message based on p_v **then**
  $m_T \leftarrow MSGT(tx, r_{2,j}^k, l_v)$, and v sends (m_T, σ_T)
else
  if channel is idle **then**
    $p_v = \min\{(1+\gamma)p_v, \hat{p}\}$
    $T_v = \max\{1, T_v - 1\}$
  else
    if receives a message (m_T, σ_T) **then**
      $p_v = (1+\gamma)^{-1} p_v$
$c_v = c_v + 1$
if $c_v \geq T_v$ **then**
  $c_v = 1$
  if there is no idle round in the past T_v rounds **then**
    $p_v = (1+\gamma)^{-1} p_v$,
    $T_v = T_v + 2$

Persistence and Liveness

Persistence. If an honest node v proclaims a transaction as stable, other honest nodes, if queried, either report the same result or report error messages.

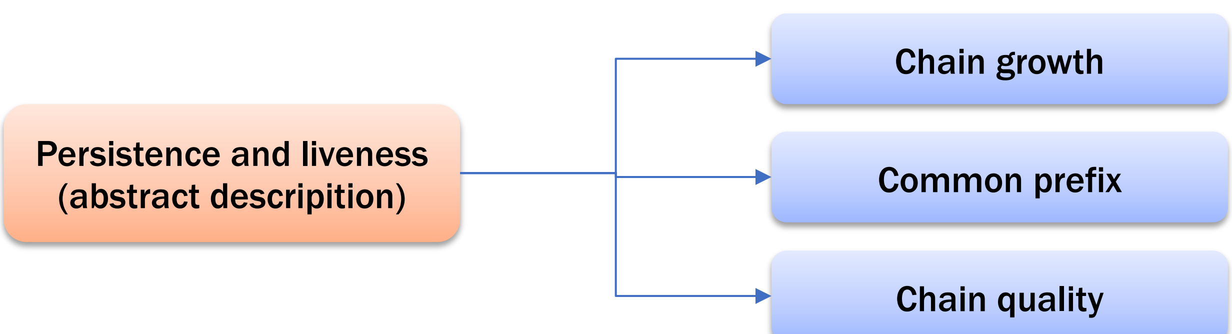
Liveness. If an honest node generates a transaction and contends to send it in phase two, the BLOWN protocol can add it to the blockchain within $O(cw_{max}\lambda)$ epochs w.h.p.

Persistence and liveness can be further divided into three concrete properties [1]

Chain growth property. With parameters $\tau \in (0,1]$, $k \in N$. Consider two chains C_1, C_2 possessed by two honest nodes at the onset of two epochs $e_1 < e_2$ with e_2 at least k epochs ahead of e_1. It holds that $len(C_2) - len(C_1) \geq \tau \cdot k$, where τ is the speed coefficient.

Common prefix property. With parameters $k \in N$, the C_1, C_2 possessed by two honest nodes at the onset of the epoch $e_1 < e_2$ are such that $C_1^{\lceil k} \preccurlyeq C_2$, where $C_1^{\lceil k}$ denotes the chain obtained by removing the last k blocks from C_1 and $\preccurlyeq$ denotes the prefix relation.

Chain quality property. With parameters $\tau \in (0,1]$, $k \in N$, Consider any portion of length at least l of the chain possessed by an honest party at the onset of an epoch; the ratio of blocks originating from the adversary is at most $1 - \mu$, where μ is the chain quality coefficient.



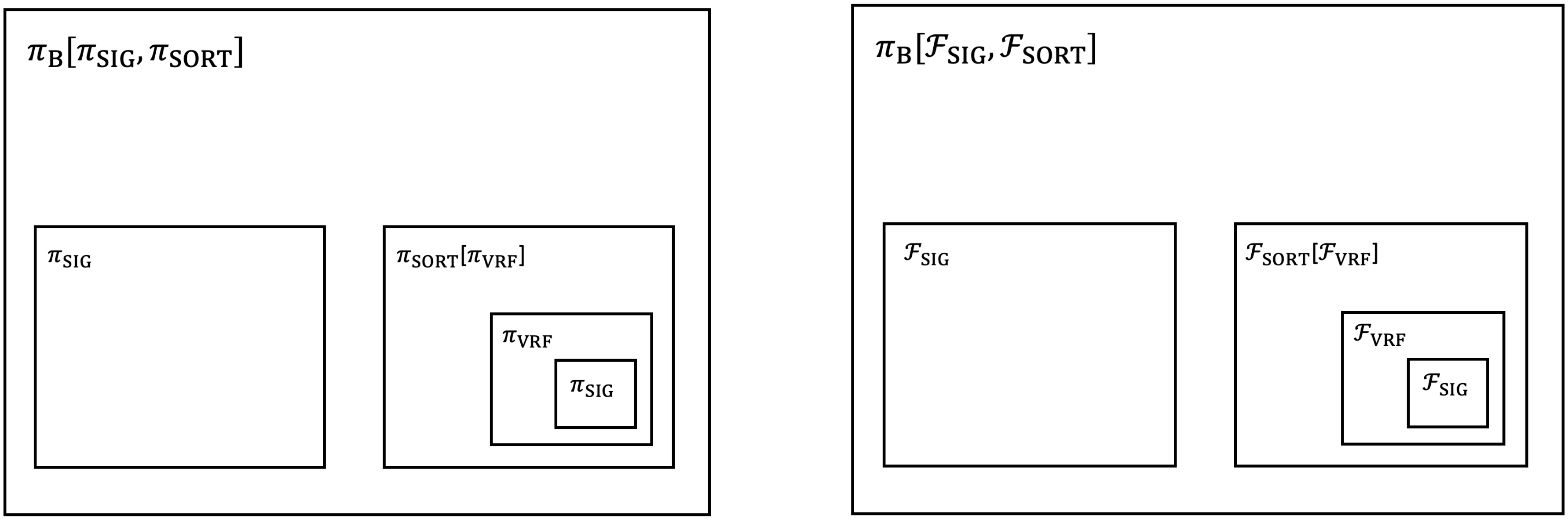
[1] The bitcoin backbone protocol: Analysis and applications, Juan A. Garay, Aggelos Kiayias, and Nikos Leonardos.

Hybrid Experiment

Universal composition

Environment $\mathcal{Z}$

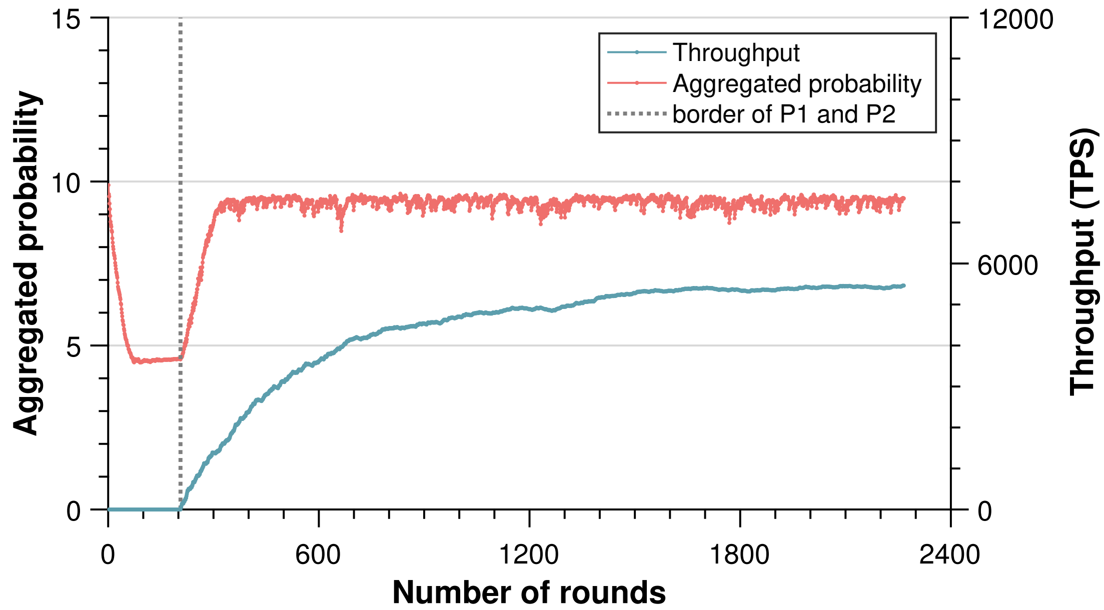
Real BLOWN **Ideal BLOWN**



We prove that Environment $\mathcal{Z}$ can not distinguish between $\pi_B[\pi_{SIG}, \pi_{SORT}]$ and $\pi_B[\mathcal{F}_{SIG}, \mathcal{F}_{SORT}]$. Any good property achieved by ideal BLOWN can be achieved by real BLOWN.

Simulation Result

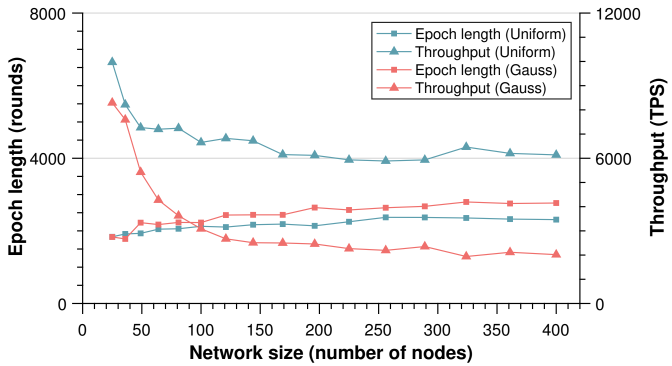
Correctness and Efficiency (One-epoch execution)



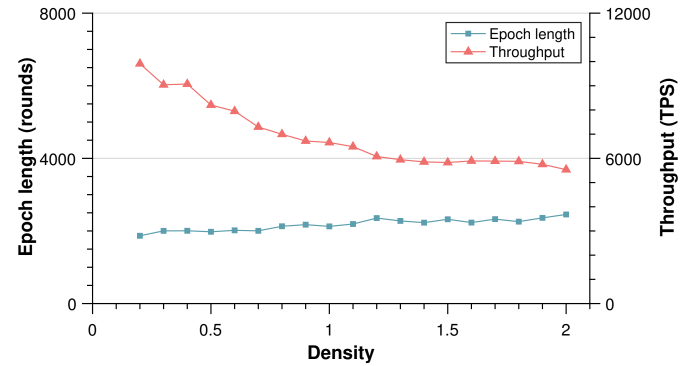
The gray dash borderline distinguishes P1 and P2. BLOWN can rapidly adjust transmission probability to reduce the noise in the channel to help achieve successful communications. it only takes 206 rounds to finished leader election (0.206s in real implementation). p_v and throughput respectively converge to 9.37 and 5399 TPS.

Simulation Result

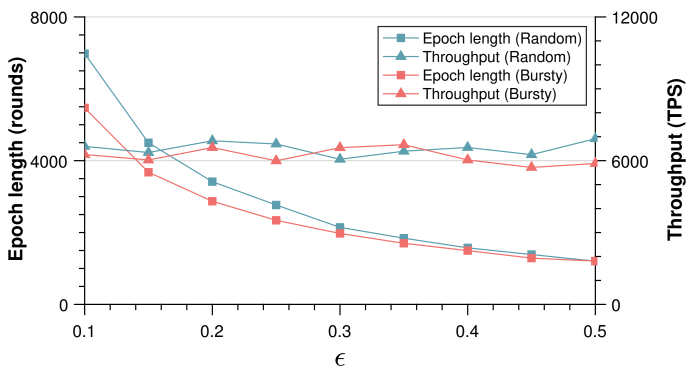
How is BLOWN impacted by network size, network density, different types of jamming attackers, power of adversarial jammming, and percentage of sybil nodes?



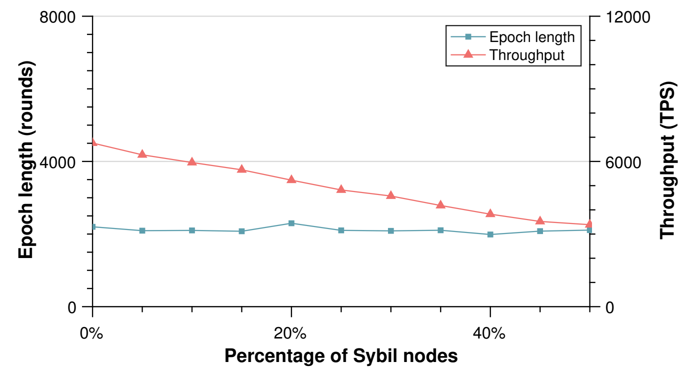
(a) Epoch length and throughput vs. the network size N, where $density = 1$, $d = \sqrt{N} \times \sqrt{N}$.



(b) Epoch length and throughput vs. the density, $where\ d = 10, N = 100$.



(c) Epoch length and throughput vs. ϵ, where $density = 1, N = 100$.



(c) Epoch length and throughput vs. the percentage of Sybil nodes, where $density = 1,\ d = 10,\ N = 100$.

1. BLOWN: A Blockchain Protocol for Wireless Networks under Adversarial SINR （Major Revision, TMC）

2. wChain: A Fast Fault-Tolerant Blockchain Protocol for Multi-hop Wireless Networks (TWC 21)

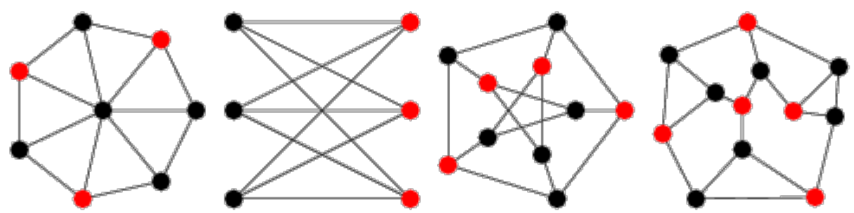
Design Objectives

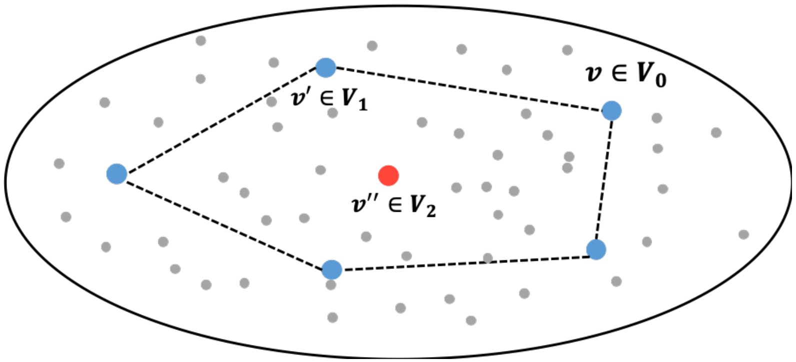
- Multi-Hop-Oriented: target for multi-hop wireless networks
- Full decentralization: should be a distributed protocol that does not rely on a centralized authority. This also indicates that the protocol should be free from any single point of failure.
- Fault-Tolerance: The protocol should be robust enough against $f = \left\lfloor \frac{N}{2} \right\rfloor$ faulty nodes defined in the network model. In particular, we should address the tricky problem when nodes become faulty within an epoch.
- High Efficiency: has low communication complexity, providing low latency and high throughput.
- Persistence and Liveness: As a blockchain protocol, wChain should satisfy persistence and liveness defined in the protocol analysis.

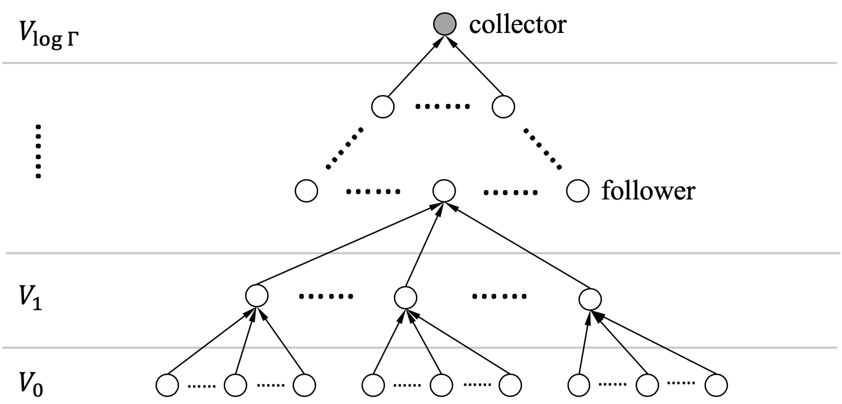
Spanner Construction

Maximum Independent Set (MIS), $O(\log N)$

Spanner Construction, $O(\log N \log \Gamma)$







$V_{\log \Gamma} \subseteq V_{\log \Gamma - 1} \subseteq \cdots \subseteq V_1 \subseteq V_0 = V$

In the i-th round

- nodes in V_i constitute an MIS of V_{i-1} with respect to $r_i = 2i$;
- each node $v \in V_{i-1} \setminus V_i$ has a parent node $u \in V_i$ and $d(v,u) \leq r_i$;
- $V_{\log \Gamma}$ only contains one root node.

Data Aggregation

Algorithm 2: *DataAggregation*$(data_v)$ Subroutine

Function *DataAggregation*$(data_v)$
  Initially, $m_v \leftarrow MSG(data_v)$, $M_v = \{m_v\}$
  ▷ In $R_i (i = 1, 2, \cdots, \log \Gamma)$:
  if $v \in V_{i-1} \setminus V_i$ **then**
    for $\mu \cdot \log N$ *slots* **do**
      send M_v with probability $p = \frac{1}{\sigma \lambda'}$ and power $P_i = 2\mathcal{N}\beta r_i^{\alpha}$
  else
    if $v \in V_i$ **then**
      for $\mu \cdot \log N$ *slots* **do**
        listen on the channel
        if *receive a valid* M_u **then**
          $M_v \leftarrow add(M_v, M_u)$

Theorem 2. Efficiency of data aggregation and reaggregation. The runtime of the data aggregation subroutine is upper bounded by $O(\log N \log \Gamma)$ slots w.h.p., and the runtime of the reaggregation subroutine is upper bounded by $O(f \log N \log \Gamma)$ slots w.h.p.

Algorithm 3: *Reaggregation*$(data_v)$ Subroutine

▷ as a leader
while *true* **do**
  ▷ slot one
  broadcast M_ℓ^{data}
  ▷ slot two
  listen on the channel
  ▷ slot three
  if *sense noise* $> \mathcal{N}$ *in slot two* **then**
    broadcast $m_\ell \leftarrow MSG(reaggregation_\ell)$
  else
    broadcast $m_\ell \leftarrow MSG(stop_\ell)$ and break
  ▷ data reaggregation
  wait for aggregated data from a collector
▷ as a follower
while *true* **do**
  ▷ slot one
  listen on the channel
  ▷ slot two
  if *receive* M_ℓ^{data} *in slot one and* $data_v \notin M_\ell^{data}$ **then**
    broadcast $m_v \leftarrow MSG(miss_v)$
  ▷ slot three
  listen on the channel
  ▷ data reaggregation
  if *receive reaggregation message in slot three* **then**
    run *SpannerConstruction*
    run *DataAggregation*$(data_v)$
  else
    break

wChain: Fast Fault-Tolerant Blockchain Protocol

Algorithm 4: Fast Fault-Tolerant Blockchain Protocol

▷ PREPARE
▷ as a leader
broadcast $m_\ell \leftarrow MSG(view_\ell)$
listen on the channel for $\mu \log N \log \Gamma$ slots
execute $Reaggregation(view_\ell)$
▷ as a follower
if *receive* $view_u$ *from a leader* **then**
  run $DataAggregation(view_v)$
else
  abandon the current epoch
execute $Reaggregation(view_v)$
▷ COMMIT
▷ as a leader
if $|\{m \in M_\ell^{view} | m.data = view_\ell\}| \geq f + 1$ **then**
  broadcast $m_\ell \leftarrow MSG(correct_\ell)$
  listen on the channel for $\mu \log N \log \Gamma$ slots
execute $Reaggregation(tx_v)$
▷ as a follower
if *receive* $correct_\ell$ *from a leader* **then**
  run $DataAggregation(tx_v)$
else
  abandon the current epoch
execute $Reaggregation(tx_v)$
▷ DECIDE
▷ as a leader
$B_\ell \leftarrow packup(M_\ell^{tx})$
$BC_\ell \leftarrow append(BC_\ell, B_\ell)$
broadcast $BC_\ell^{i+} \leftarrow extract(BC_\ell, M_\ell^{view})$
▷ as a follower
if *receive* BC_ℓ^{i+} *from the leader* **then**
  update(BC_v, BC_ℓ^{i+})
else
  abandon the current epoch

PREPARE phase:
help a leader obtain a global view

COMMIT phase:
- **Leader should successfully receive no less than f view messages to ensure that it has identical views as the majority.**
- **Collect transactions**

DECIDE phase:
Generate and broadcast a new block

Theoretical Analysis

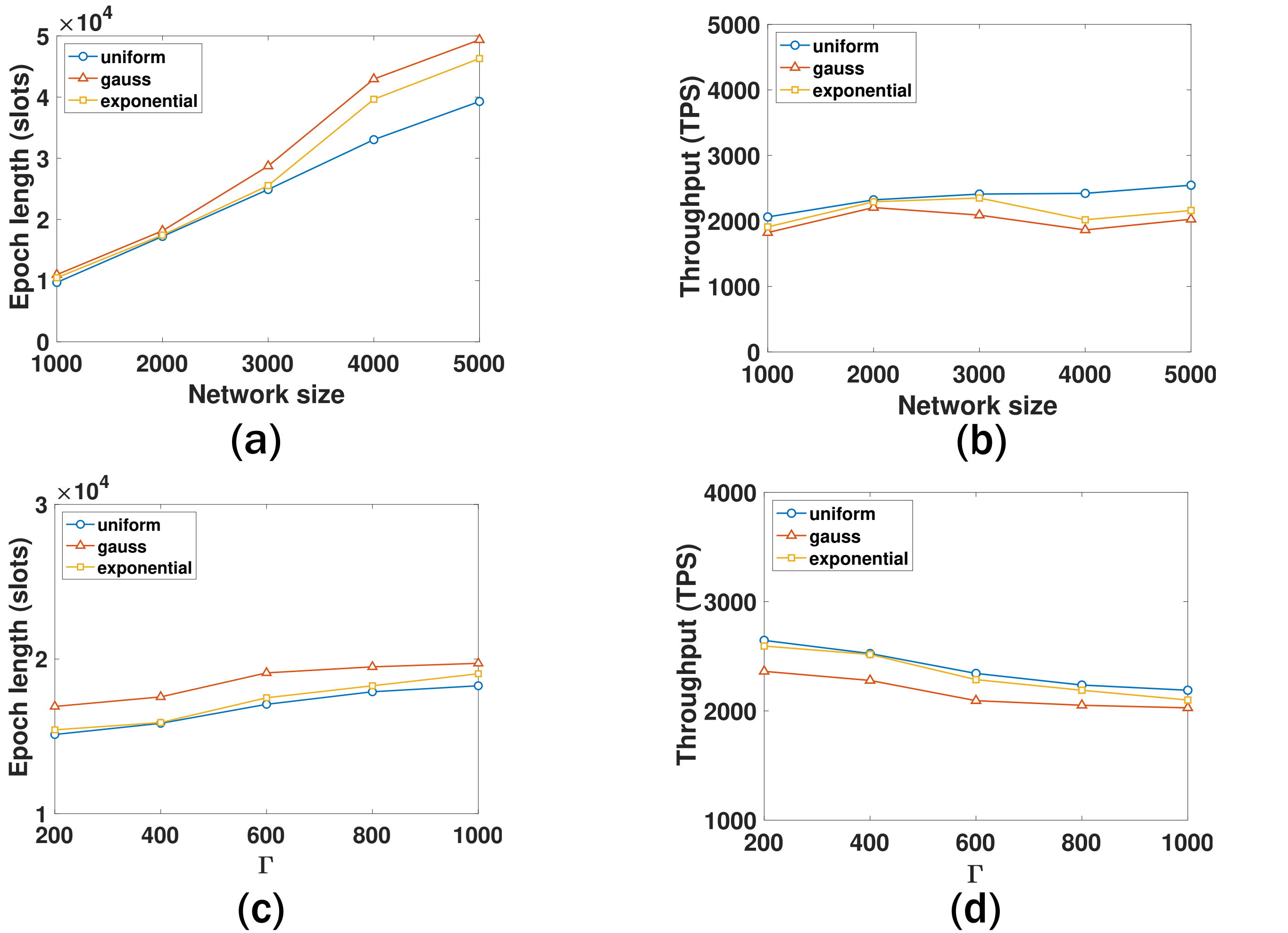
Algorithm 4: Fast Fault-Tolerant Blockchain Protocol

▷ PREPARE
▷ as a leader
broadcast $m_\ell \leftarrow \mathit{MSG}(view_\ell)$
listen on the channel for $\mu \log N \log \Gamma$ slots
execute $\mathit{Reaggregation}(view_\ell)$
▷ as a follower
if *receive* $view_u$ *from a leader* **then**
  run $\mathit{DataAggregation}(view_v)$
else
  abandon the current epoch
execute $\mathit{Reaggregation}(view_v)$
▷ COMMIT
▷ as a leader
if $|\{m \in M_\ell^{view} | m.data = view_\ell\}| \geq f + 1$ **then**
  broadcast $m_\ell \leftarrow \mathit{MSG}(correct_\ell)$
  listen on the channel for $\mu \log N \log \Gamma$ slots
execute $\mathit{Reaggregation}(tx_v)$
▷ as a follower
if *receive* $correct_\ell$ *from a leader* **then**
  run $\mathit{DataAggregation}(tx_v)$
else
  abandon the current epoch
execute $\mathit{Reaggregation}(tx_v)$
▷ DECIDE
▷ as a leader
$B_\ell \leftarrow packup(M_\ell^{tx})$
$BC_\ell \leftarrow append(BC_\ell, B_\ell)$
broadcast $BC_\ell^{i+} \leftarrow extract(BC_\ell, M_\ell^{view})$
▷ as a follower
if *receive* BC_ℓ^{i+} *from the leader* **then**
  update(BC_v, BC_ℓ^{i+})
else
  abandon the current epoch

Theorem 2. Persistence. If a non-faulty node v proclaims a transaction tx_v in the position tx_i^j, other nodes, if queried, should report the same result. Here tx_i^j is called stable only when the current block index is more than $i + k$, namely k-stable.

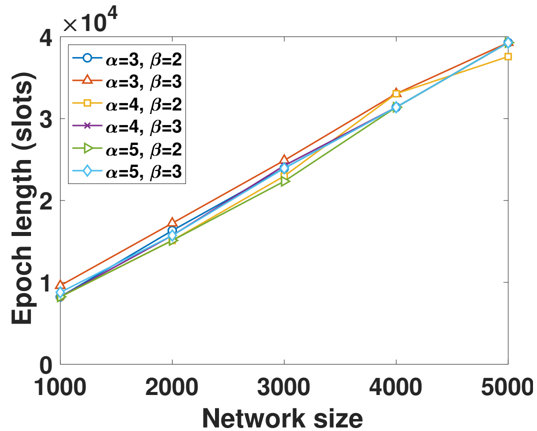
Theorem 3. Liveness. If a non-faulty node generates a transaction and contends to send it, the $wChain$ protocol can add it to the blockchains within T slots w.h.p., where the upper bound of T is $O(\log N \log \Gamma)$ when crash failures happen in a low frequency, and the worst-case upper bound is $O(f \log N \log \Gamma)$.

Evaluation

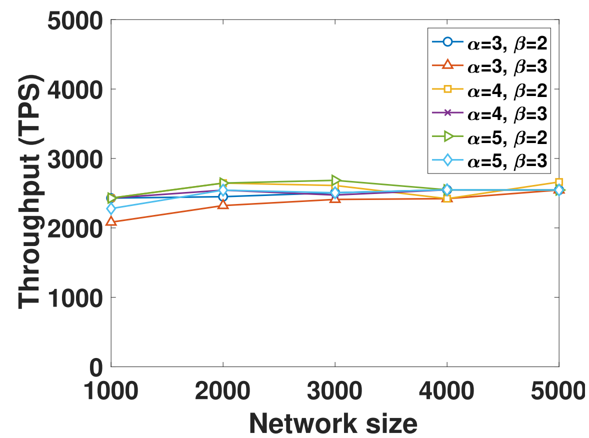


The performance of wChain vs. the network size N and Γ (under uniform, normal or exponential distributions).

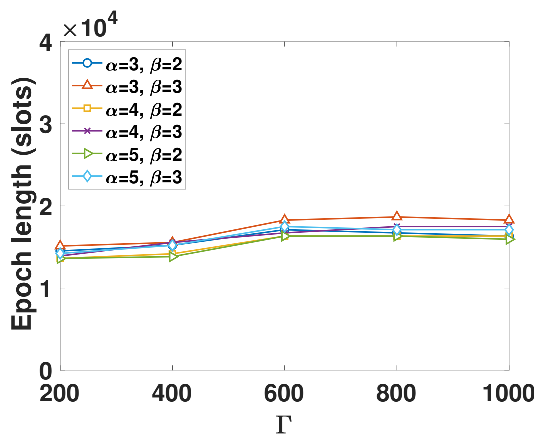
Evaluation



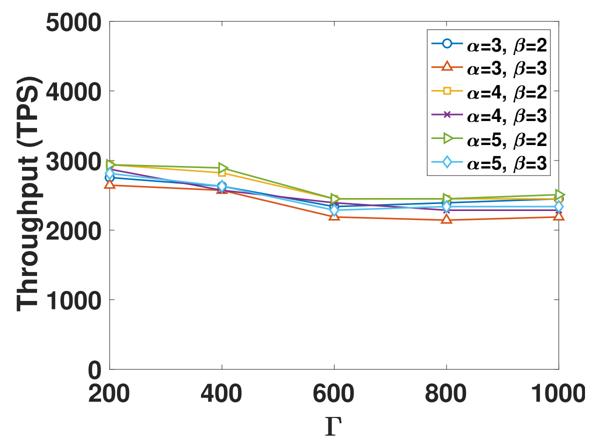
(a)



(b)



(c)



(d)

The performance of wChain with various α and β (under a uniform distribution).

Conclusions and Future Directions

❑ **Conclusions:**

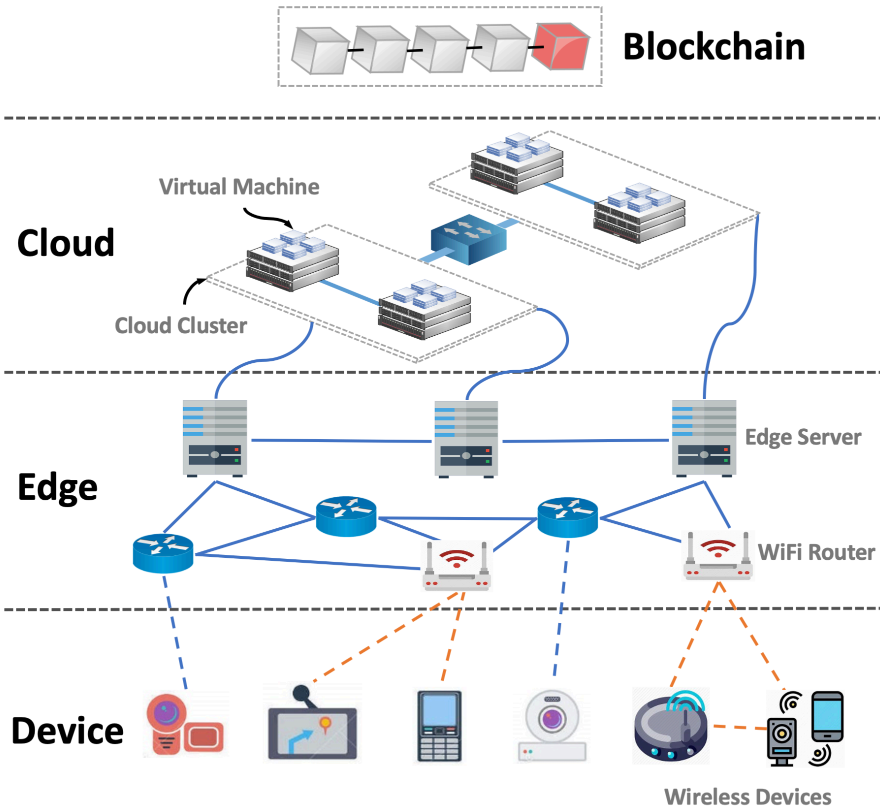
BLOWN is the first single-hop wireless blockchain protocol under adversarial jamming. It embodies a novel PoC consensus algorithm. wChain is the first multi-hop wireless blockchain protocol, and that it is complementary to BLOWN. They both have nice properties and show good performance.

❑ **Future Directions:**

1. Investigate the the byzantine fault-tolerant versions of BLOWN and wChain, consider more realistic problems such as fading channel, obstacles.
2. Build up a simulator of wireless blockchain. Further implement wireless blockchains in intelligent swarms (e.g., robotics, UAV, smart dust)
3. It is also interesting to explore blockchain-secured swarm intelligence.

Conclusions and Future Directions

1) **Blockchain Fundamentals**: revolutionize the blockchain technology in three layers (i.e., cloud, edge, device);
2) **Blockchain Applications**: utilize blockchain to solve practical security problems of access control, Software Defined Network (SDN), and decentralized learning, etc.;
3) **Preparations for Quantum Era**: quantum game theory and quantum distributed algorithms.



LatticeChain (Current)
- ❑ 3D ledger beyond chain (1D) and DAG (2D)

CloudChain (Previous)
- ✓ cloud-oriented blockchain using shared-memory and RDMA

BC-SDNC (Future)
BC-DPSGD (Future)
- ❑ trusted SDN control plane
- ❑ trusted decentralized learning

Tokoin (Previous)
VaccineSys (Previous)
- ✓ access control for IoT
- ✓ cyber-physical trust extension

wChain (Previous)
BLOWN (Previous)
- ✓ sinlge-hop wireless blockchain
- ✓ multi-hop wireless blockchain

Thank You!

THE GEORGE WASHINGTON UNIVERSITY

WASHINGTON, DC

Q & A