PennState

CSE 597: Security of Emerging Technologies

Module: Formal Verification

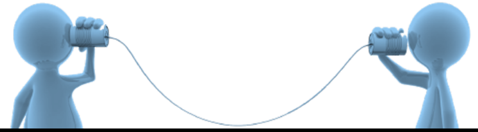
Prof. Syed Rafiul Hussain

Systems and Network Security (SyNSec) Research Group

Department of Computer Science and Engineering

The Pennsylvania State University

Critical Infrastructure using Cellular Network



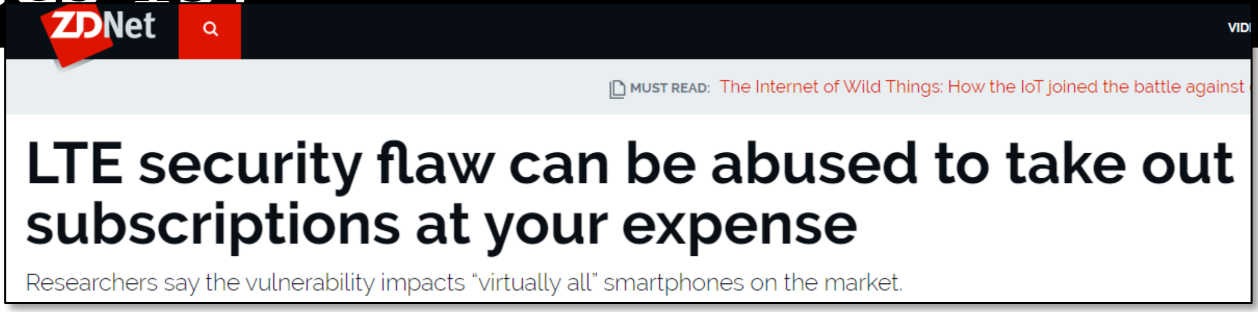
Problem Statement: How can we systematically verify the design of 4G & 5G network protocols with respect to promised security and privacy guarantees?

(CCS'19)

TC

New flaws in 4G, 5G allow attackers to intercept calls and track phone locations

Zack Whittaker @zackwhittaker / 11:39 am EST • February 24, 2019



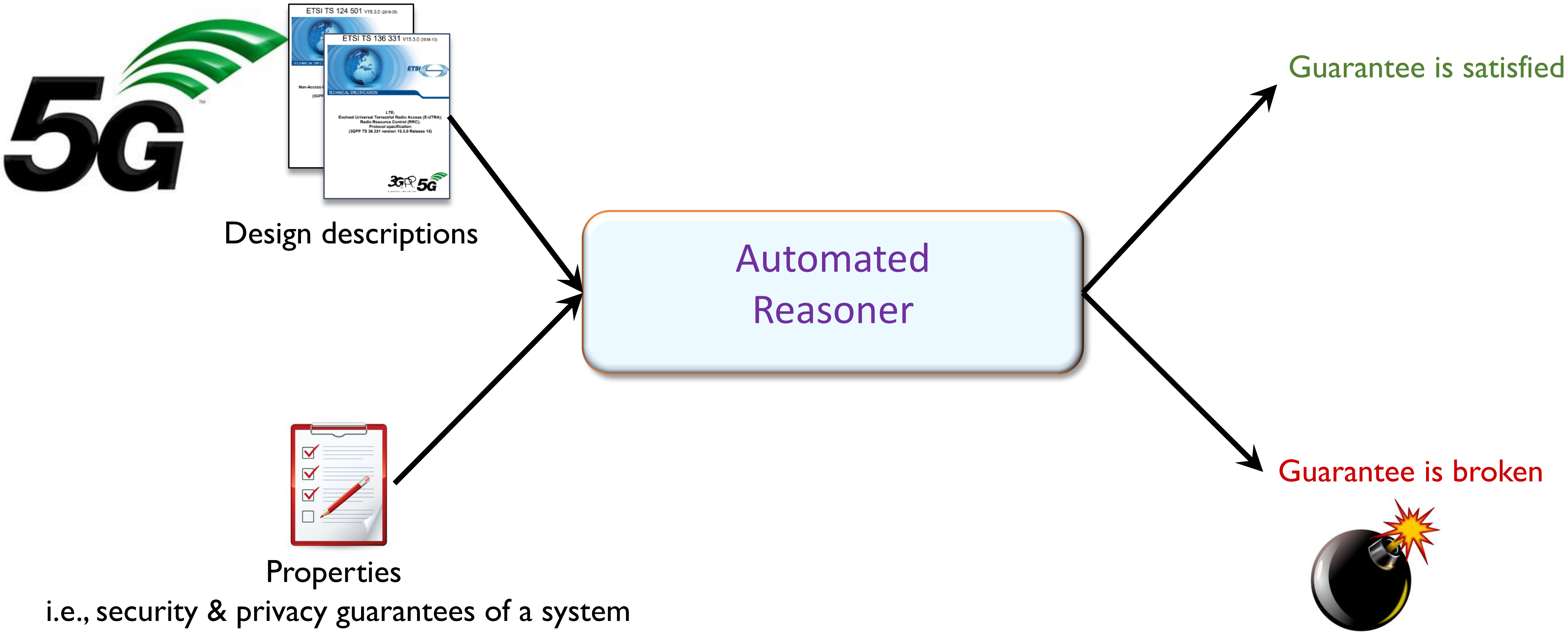
ZDNet

MUST READ: The Internet of Wild Things: How the IoT joined the battle against

LTE security flaw can be abused to take out subscriptions at your expense

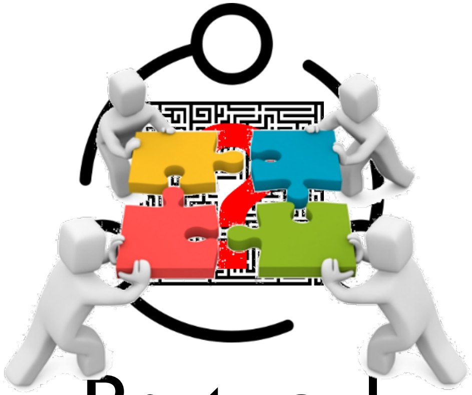
Researchers say the vulnerability impacts "virtually all" smartphones on the market.

High-level Goal



Challenges

PennState

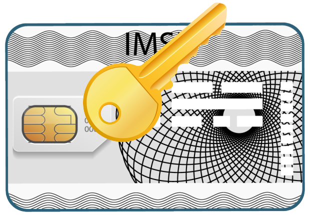


Protocol Complexity

Stateful Network	+	Cryptographic Constructs (encryption, message authentication code, certificate)
Multiple Participants	+	Intertwined Sub-Procedures
Qualitative Properties (temporal ordering of events)	+	Quantitative Properties (rate of receiving a message)

Background (Cellular Device or User Equipment)

UE

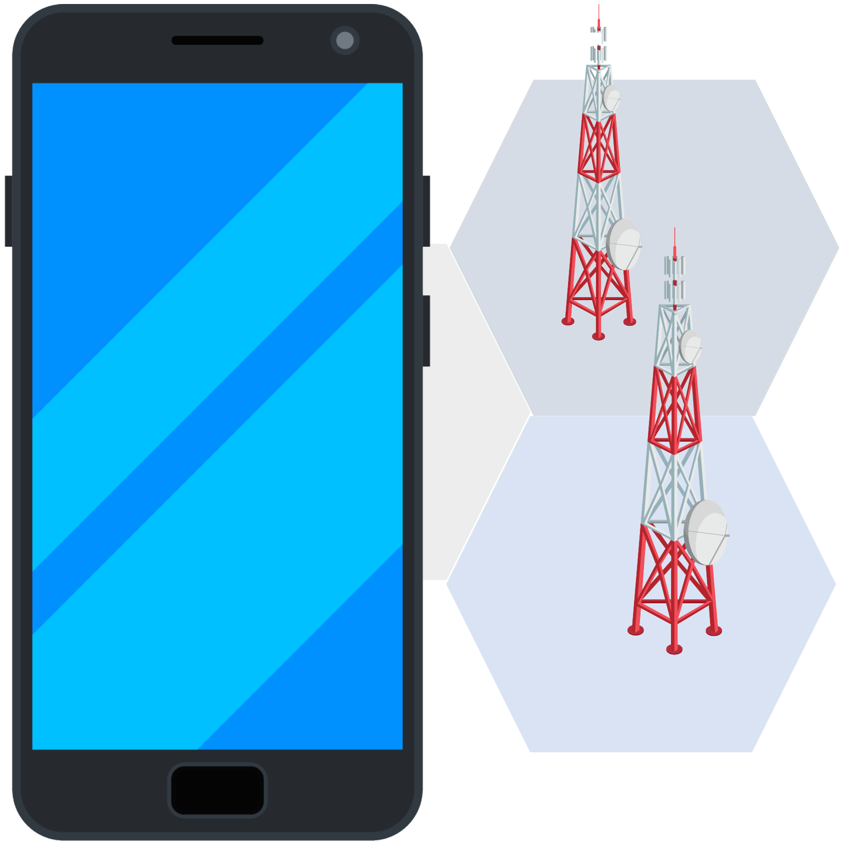


IMSI = International Mobile Subscriber Identity

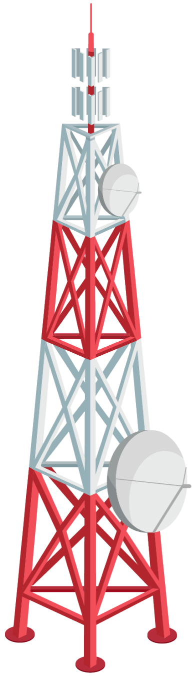
IMEI = International Mobile Equipment Identity

Background (4G System Architecture)

Registration Area

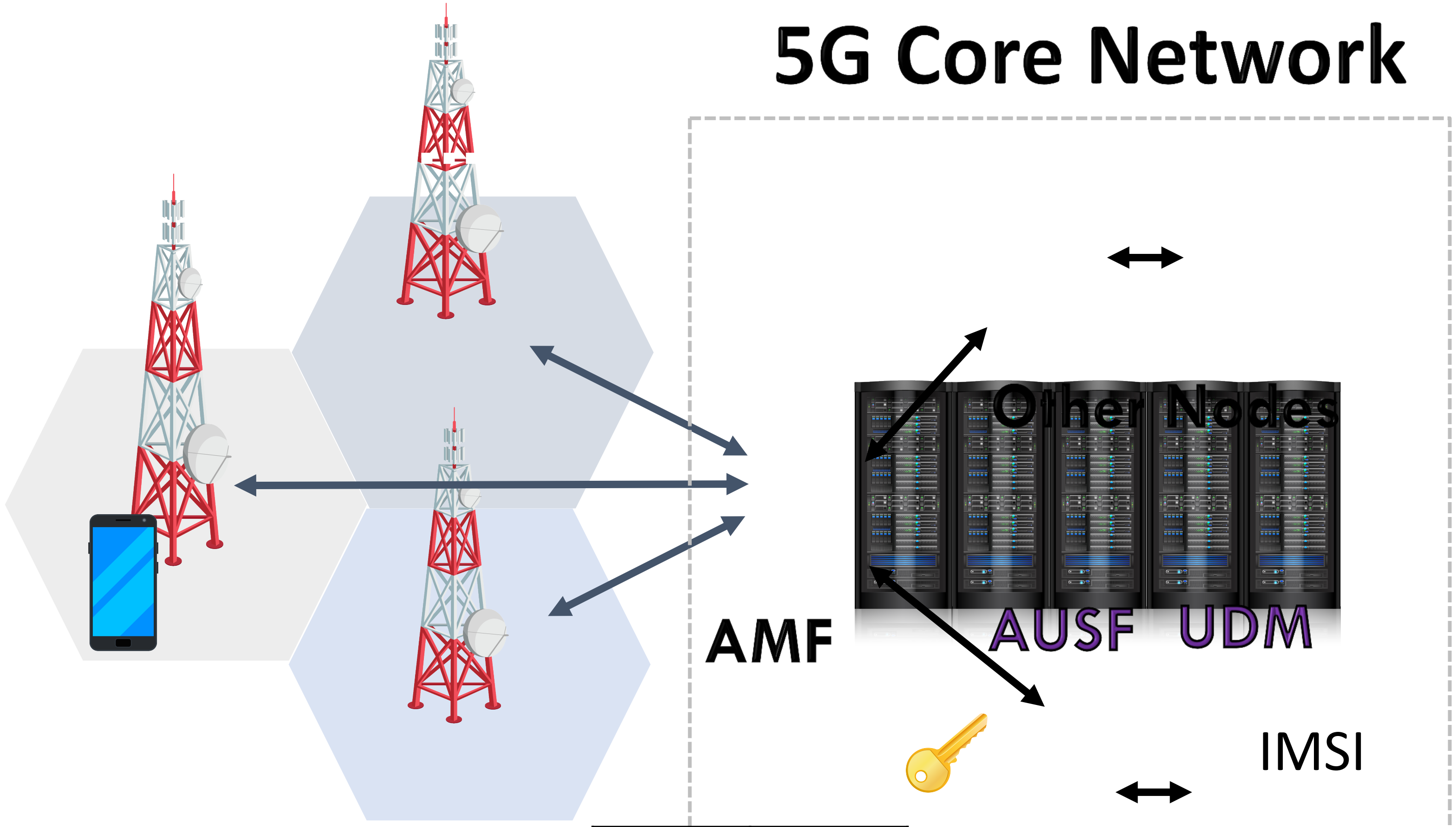


Base Station (gNB)



Background (4G System Architecture)

PennState



IMSI	Key
IMSI_1	Key_1
IMSI_2	Key_2
IMSI_victim	Key_victim

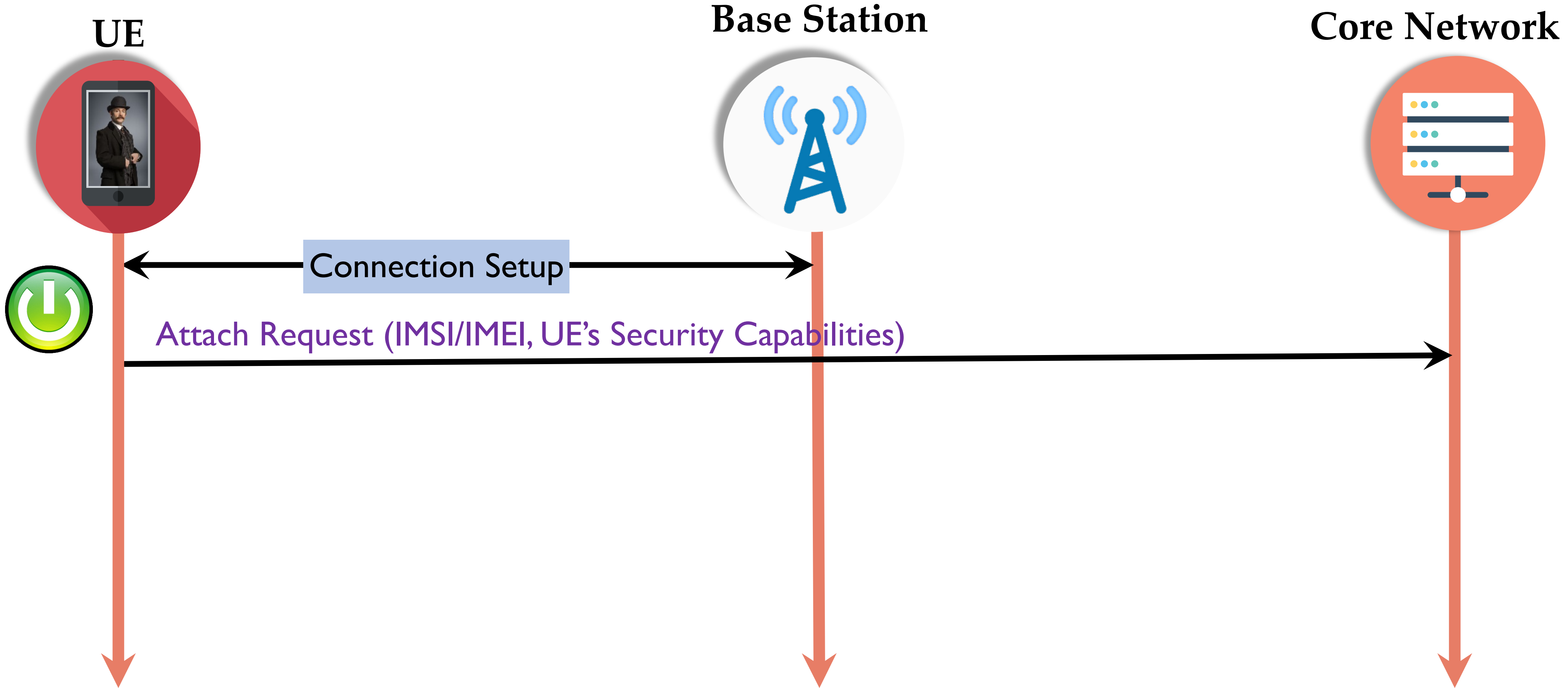
Attach/Registration Procedure

Identification

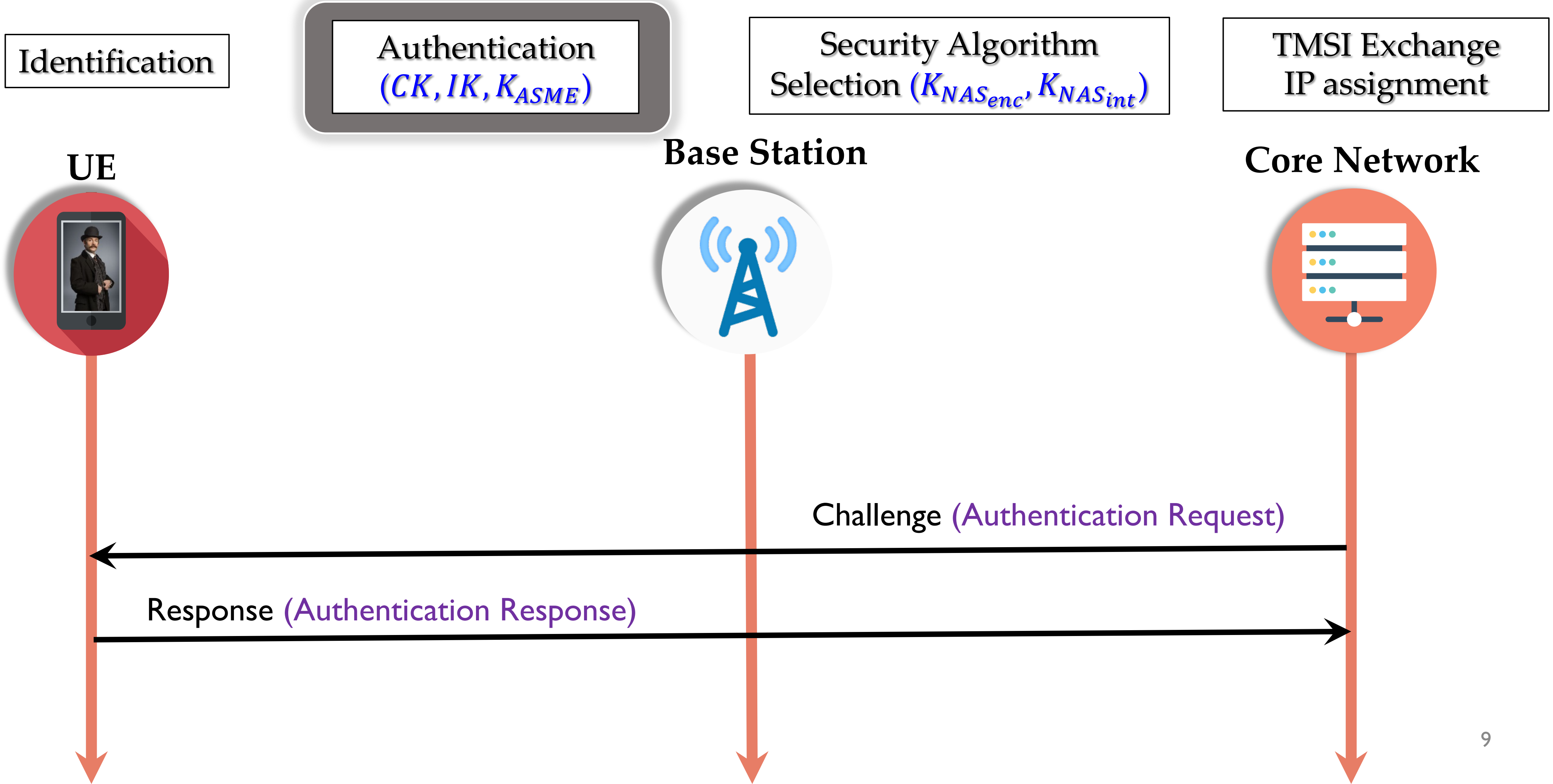
Authentication (CK, IK, K_{ASME})

Security Algorithm Selection $(K_{NAS_{enc}}, K_{NAS_{int}})$

TMSI Exchange
IP assignment



Attach/Registration Procedure



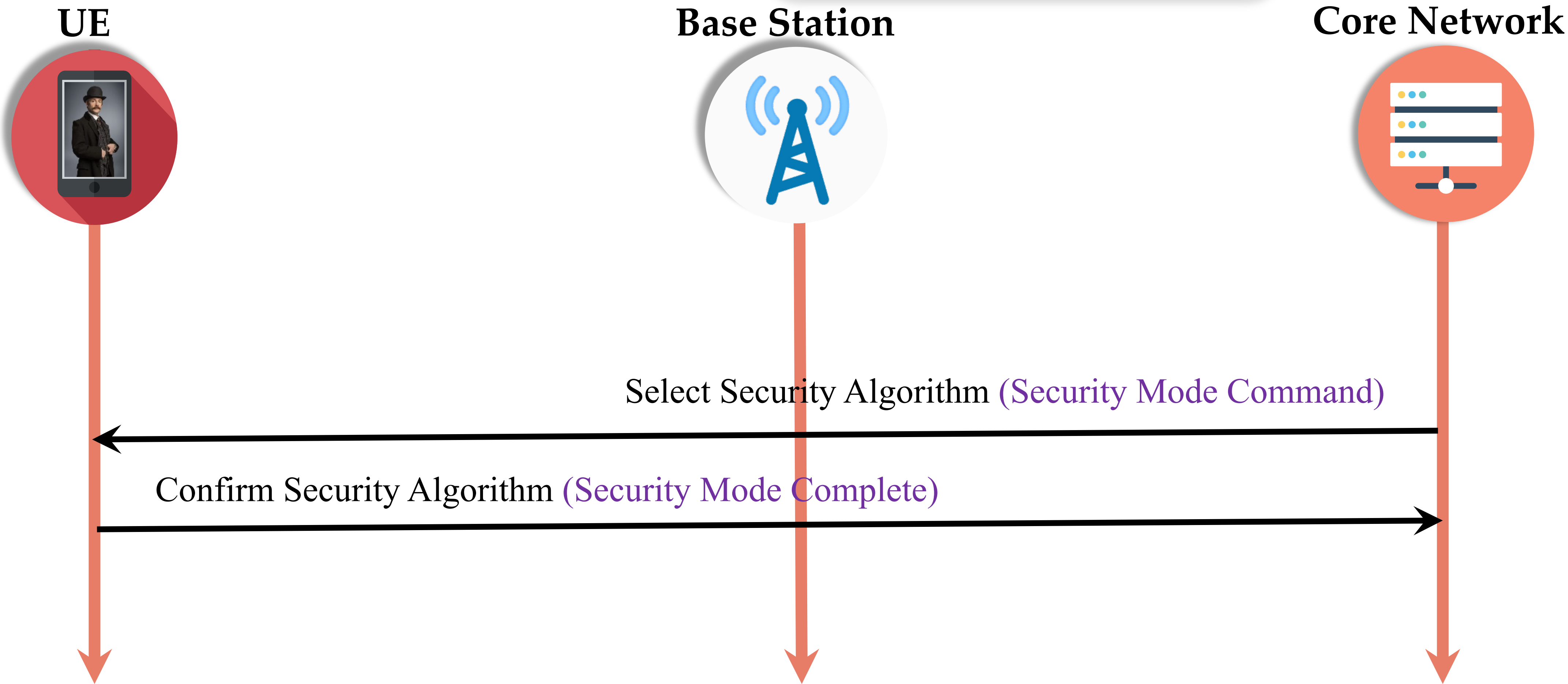
Attach/Registration Procedure

Identification

Authentication (CK, IK, K_{ASME})

Security Algorithm Selection $(K_{NAS_{enc}}, K_{NAS_{int}})$

TMSI Exchange
IP assignment



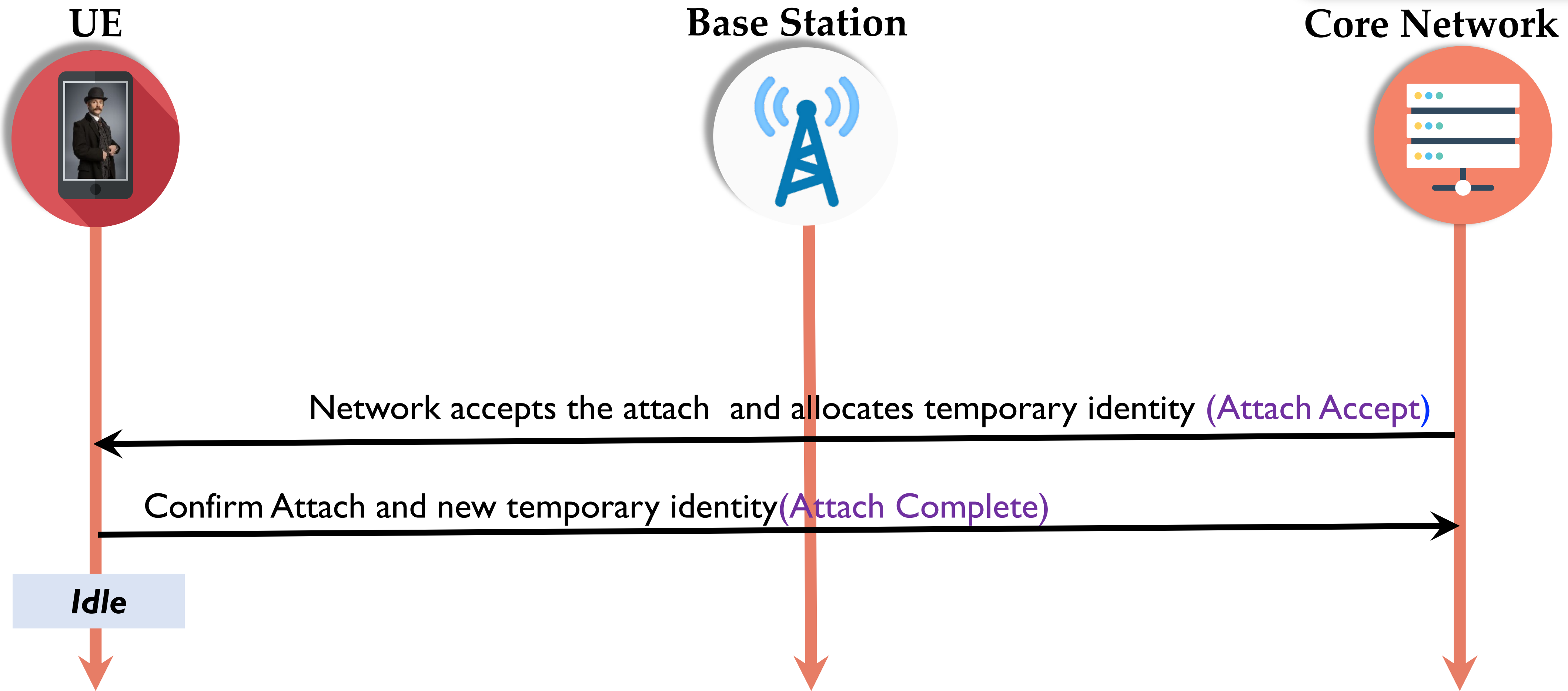
Attach Procedure

Identification

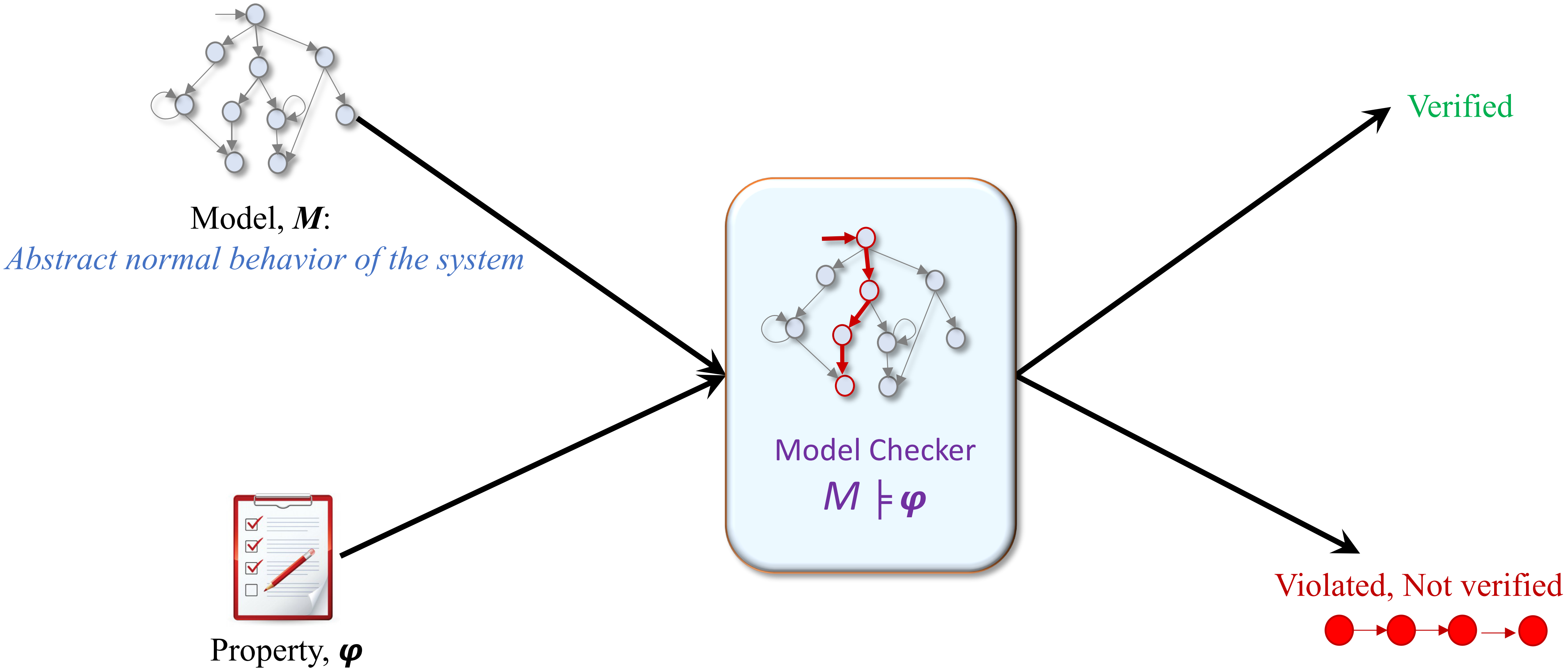
Authentication (CK, IK, K_{ASME})

Security Algorithm Selection $(K_{NAS_{enc}}, K_{NAS_{int}})$

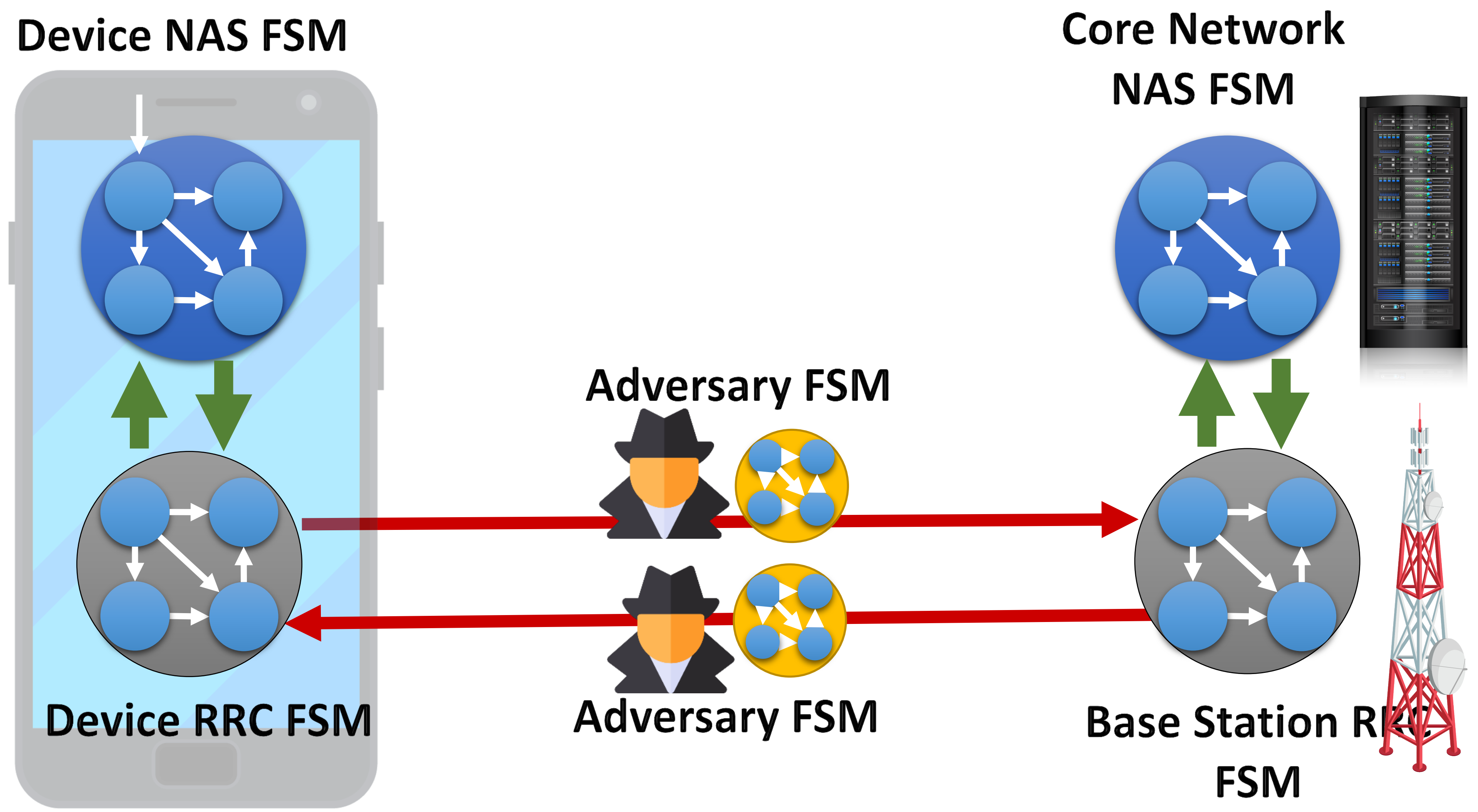
TMSI Exchange
IP assignment



Model Checking



Dolev-Yao Adversary Model



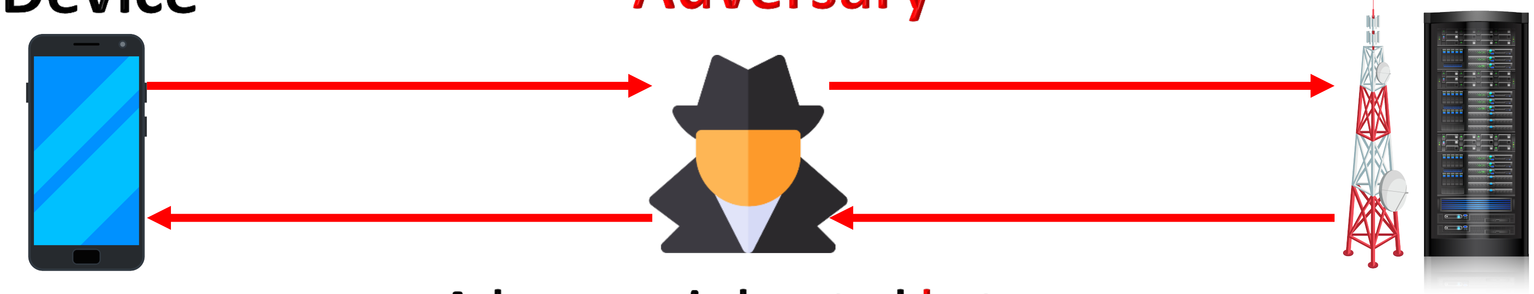
Dolev-Yao Adversary Model

PennState

Cellular Device

Adversary

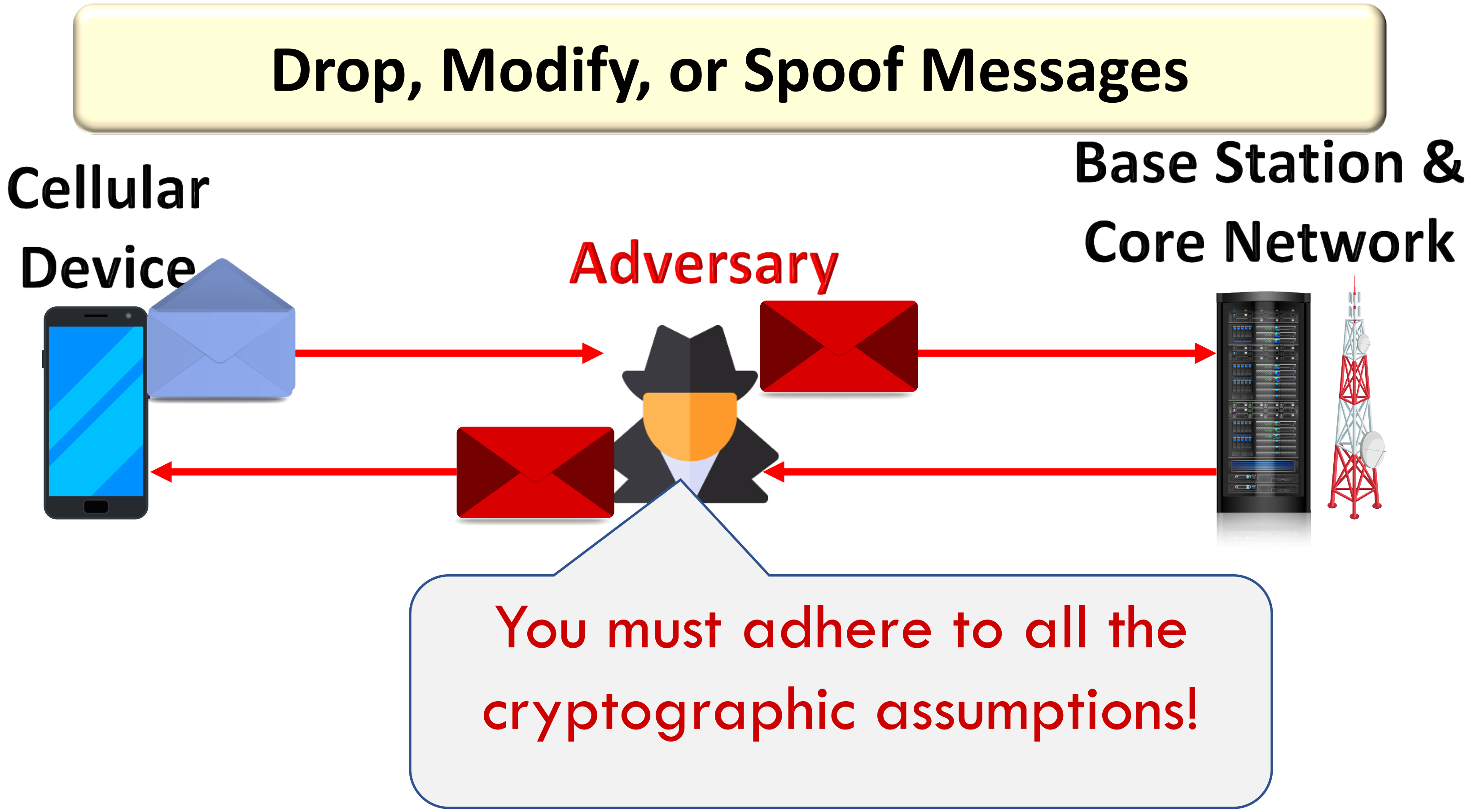
Base Station & Core Network



Adversary is located **between** cellular device and Base Station

Treated as environment variable

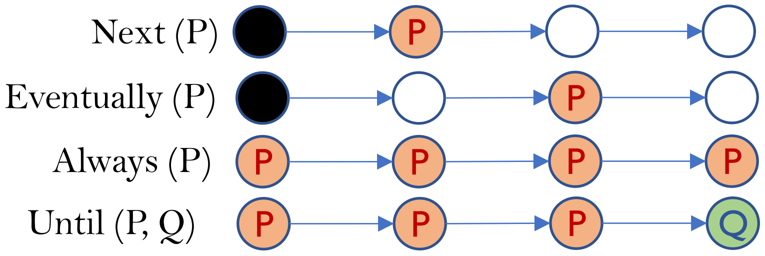
Dolev-Yao Adversary Model

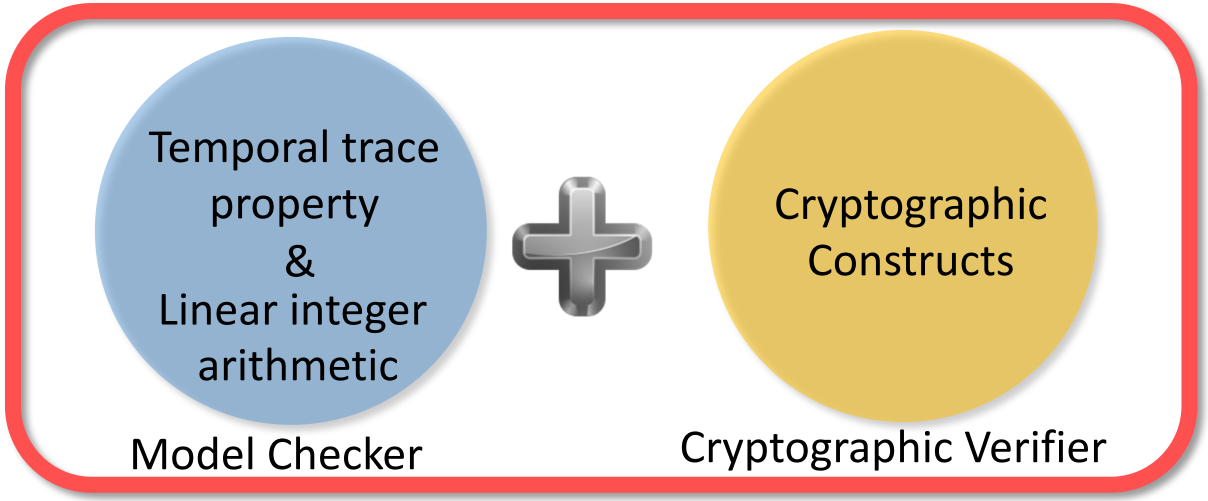


Key Insight of Our Adversarial Testing Framework

❑ Property characteristics

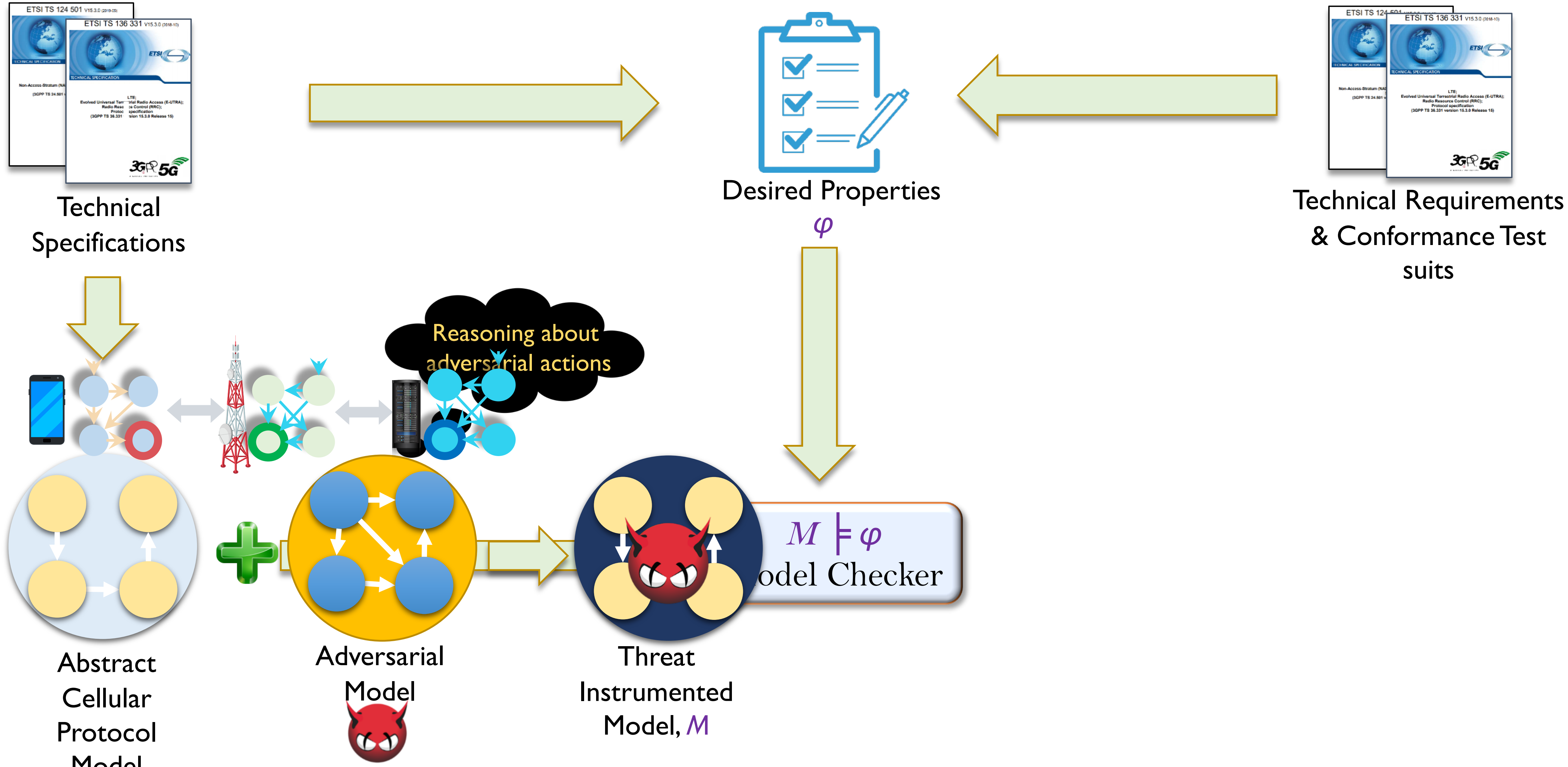
- ✓ Temporal ordering of events
- ✓ Cryptographic constructs
- ✓ Linear integer arithmetic and other predicates
  - SQN++ and verify SQN ≤ XSQN ≤ (SQN + range)



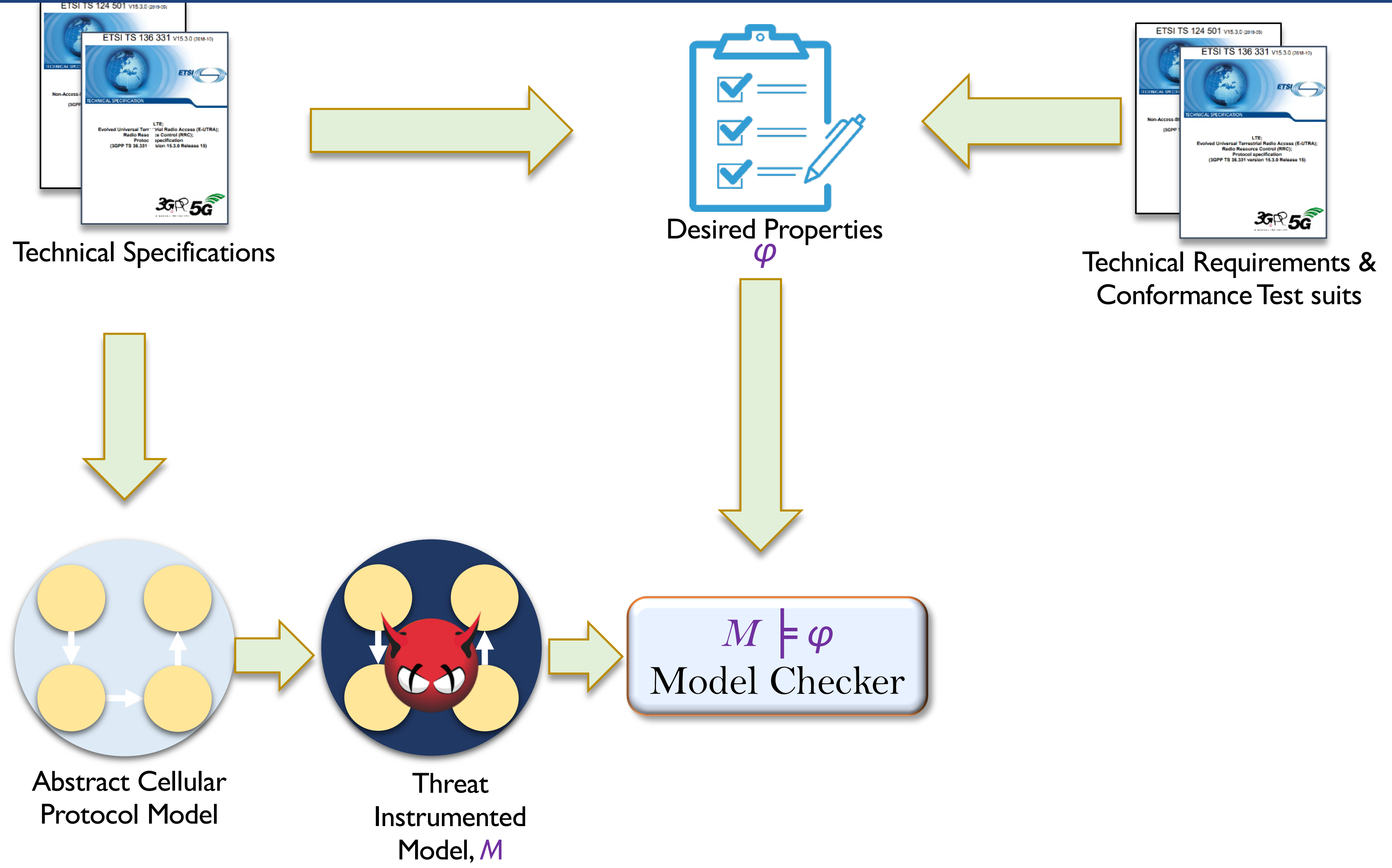


How can we leverage reasoning power of these two?

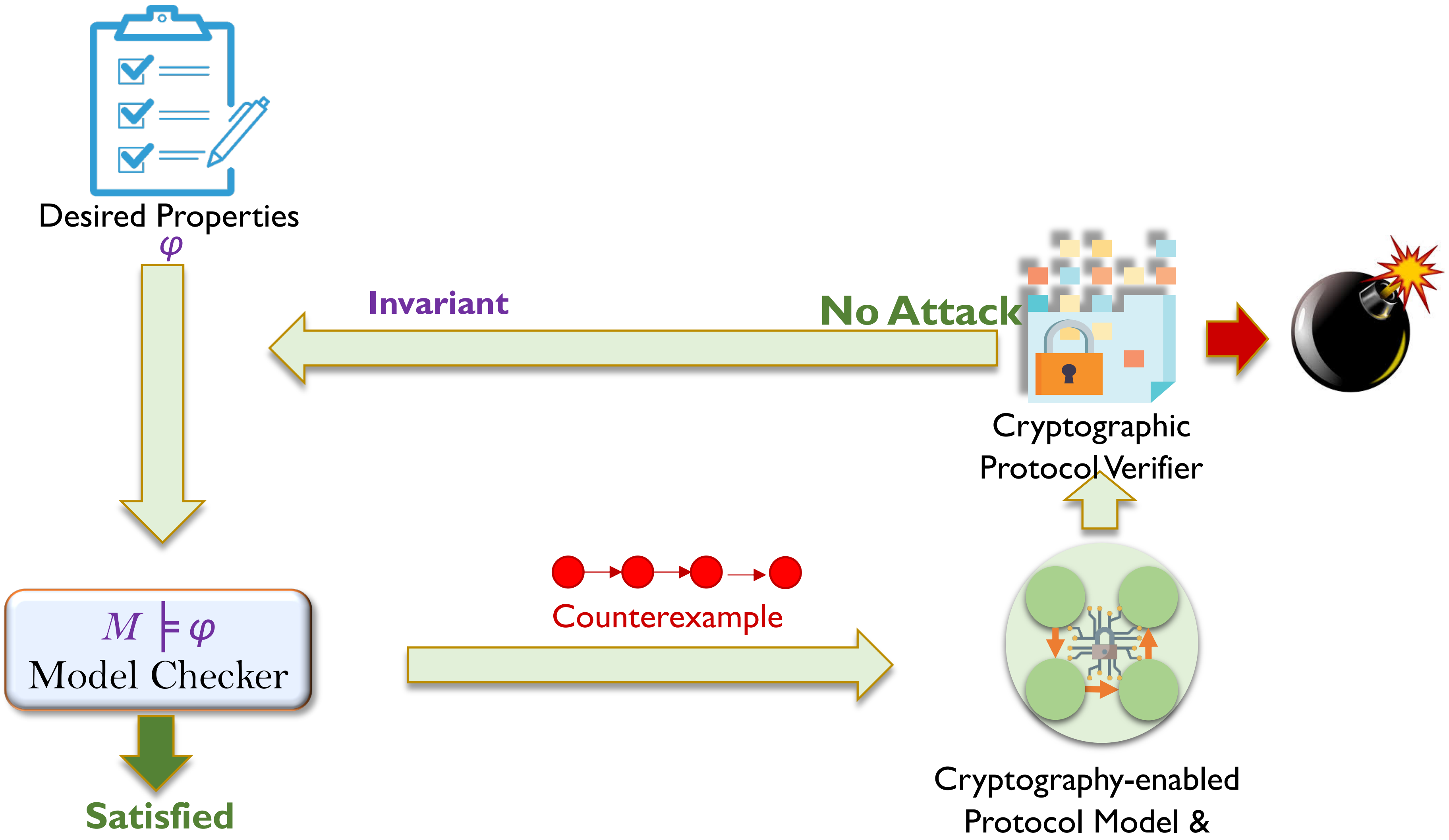
Adversarial Testing Framework: LTEInspector



Adversarial Testing Framework: LTEInspector



Adversarial Testing Framework: LTEInspector



Findings

PennState

5G 11 new attacks

4G LTE 10 new attacks

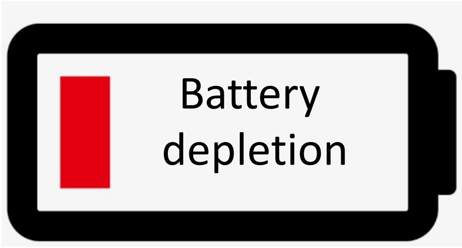

Authentication Bypass


Location tracking


Service Profiling

TMSI exposure

Overbilling




Artificial Chaos

Model Checking

PennState

- Model checking is the exhaustive exploration of the state space of a system, typically to see if an error state is reachable. It produces concrete counterexamples.
- The state explosion problem refers to the large number of states in the model.
- Temporal logic allows you to specify properties with concepts like "eventually" and "always".
- Keywords:
  - Model checking is an automated technique
  - Model checking verifies transition systems
  - Model checking verifies temporal properties

  Model checking falsifies by generating counterexamples A model checker is a program that checks if a (transition) system satisfies a (temporal) property 9

Verification vs. Falsification

PennState

- What is verification?
  - Prove that a property of a system holds
- What is falsification?
  - Disprove that a property holds

Verification vs. Falsification

PennState

- An automated verification tool
  - can report that the system is verified (with a proof);
  - or that the system was not verified.

- When the system was not verified, it would be helpful to explain why
  - Model checkers can output an error counterexample: a concrete execution scenario that demonstrates the error.

- Can view a model checker as a falsification tool –
  - The main goal is to find bugs

- So what can we verify or falsify?

Temporal Properties

- Temporal Property
  - A property with time-related operators such as "invariant" or "eventually"

- Invariant(p)
  - is true in a state if property p is true in every state on all execution paths starting at that state
  - G, AG, □ ("globally" or "box" or "forall")

- Eventually(p)
  - is true in a state if property p is true at some state on every execution path starting from that state F, AF, ◊ ("future" or "diamond" or "exists")

An Example Concurrent Program

PennState

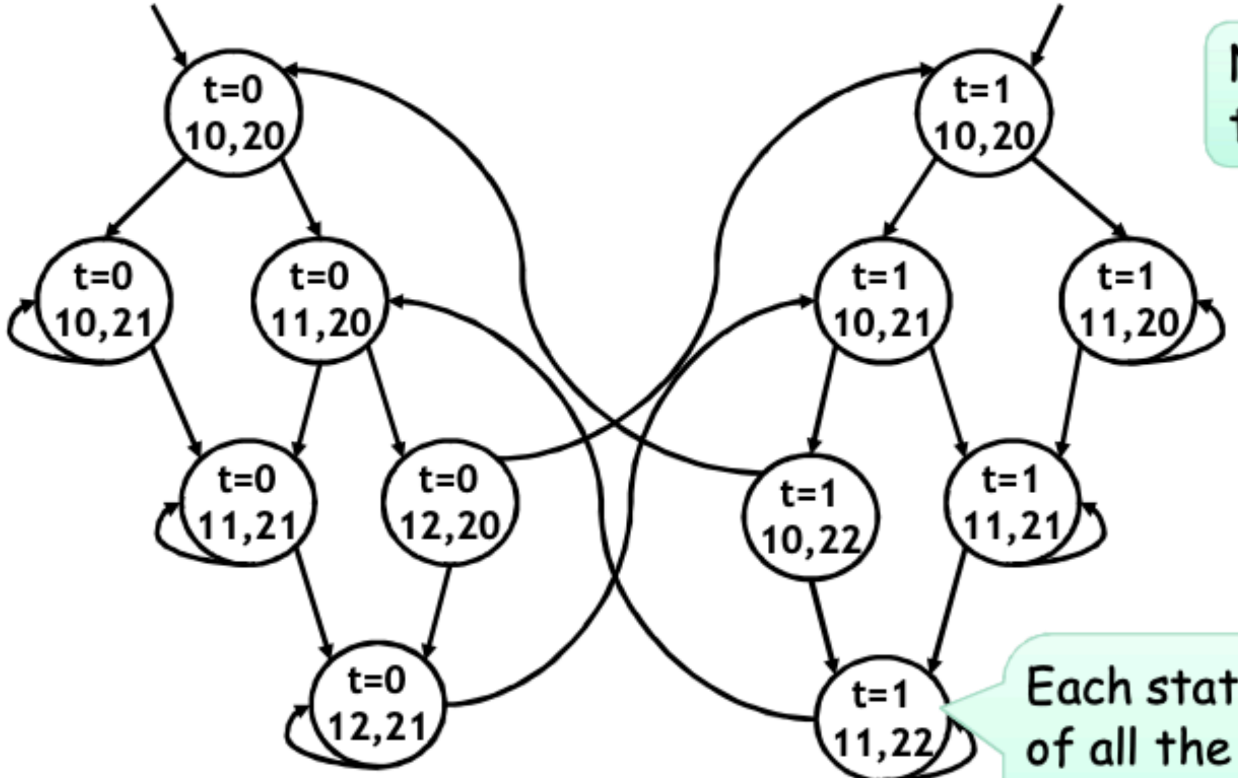
- A simple concurrent mutual exclusion program
- Two processes execute asynchronously
- There is a shared variable turn
- Two processes use the shared variable to ensure that they are not in the critical section at the same time
- Can be viewed as a "fundamental" program: any bigger concurrent one would include this one

```
10: while (true){
11:     wait(turn == 0);
        // critical section
12:     work(); turn = 1;
13: }

// concurrently with

20: while (true) {
21:     wait(turn == 1);
        // critical section
22: work(); turn = 0;
23: }
```

Reachable States of the Example Program



Next: formalize this intuition …

Each state is a valuation of all the variables: `turn` and the two program counters for two processes

Analyzed System is a Transition System

- Labeled transition system

  $T = (S, I, R, L)$ –

  S = Set of states // standard FSM

  $I \subseteq S$ = Set of initial states // standard FSM

  $R \subseteq S \times S$ = Transition relation // standard FSM

  $L: S \rightarrow 2^{AP}$ = Labeling function // this is new!

- AP: Set of atomic propositions (e.g., "x=5"∈AP)
  - Atomic propositions capture basic properties
  - For software, atomic props depend on variable values
  - The labeling function labels each state with the set of propositions true in that state

Example Properties of the Program

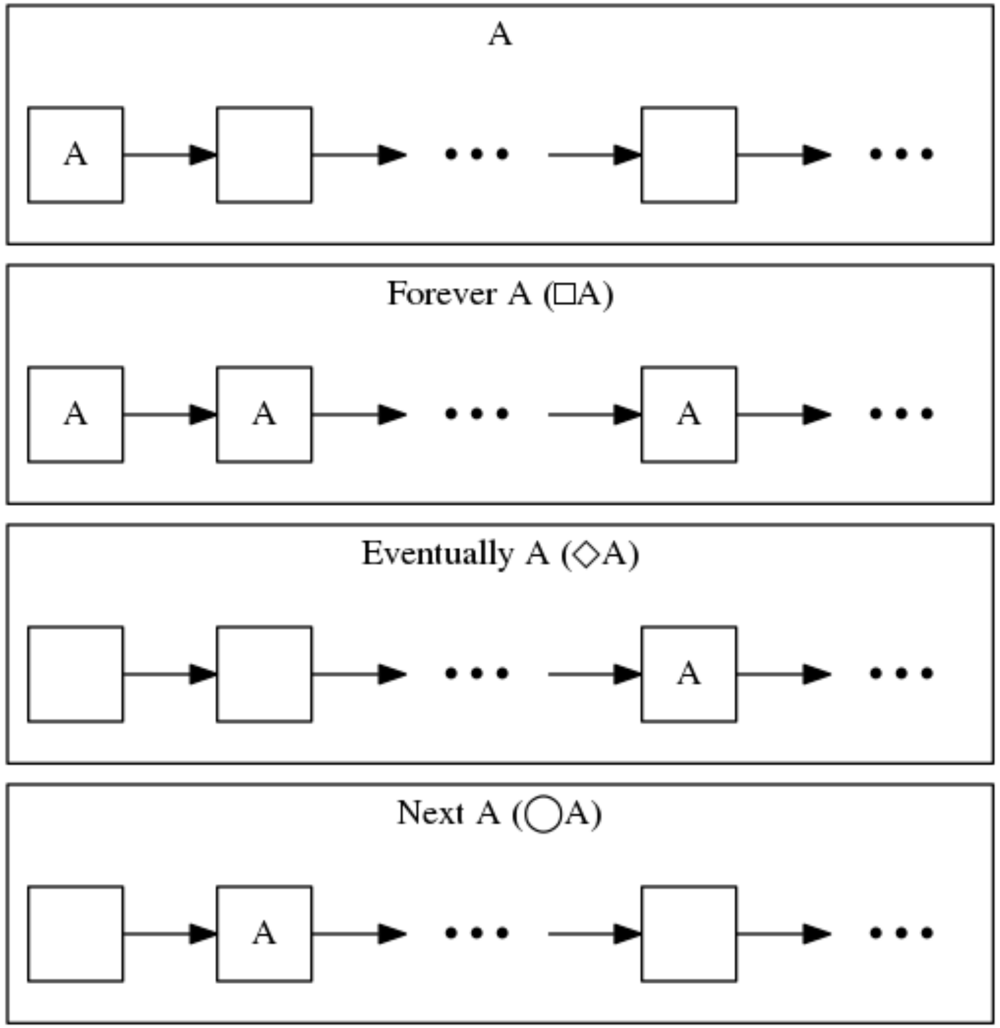
- "In all the reachable states (configurations) of the system, the two processes are never in the critical section at the same time"
  - "pc1=12", "pc2=22" are atomic properties for being in the critical section
  - Invariant ($\neg$ (PC1=12 $\wedge$ PC2 = 22)

- "Eventually the first process enters the critical section
  - Eventually (PC1 = 12)

Temporal Logics

PennState

- There are four basic temporal operators:
- X p Next p, p holds in the next state
- G p Globally p, p holds in every state, p is an invariant
- F p Future p, p will hold in a future state, p holds eventually
- p U q p Until q, assertion p will hold until q holds
- Precise meaning of these temporal operators are defined on execution paths



Execution Paths

- A path in a transition system is an infinite sequence of states
  - $(s0, s1, s2, \ldots)$, such that $\forall i \geq 0.\ (si, si+1) \in R$
- A path $(s0, s1, s2, \ldots)$ is an execution path if $s0 \in I$
- Given a path $x = (s0, s1, s2, \ldots)$
  - x_i denotes the ith state: s_i
  - x^i denotes the i-th suffix: $(s_i, s_{i+1}, s_{i+2}, \ldots)$
  - In some temporal logics one can quantify paths starting from a state using path quantifiers
    - A : for all paths
    - E : there exists a path

Paths and Predicates

- We write

$$X \models p$$

"the path x makes the predicate p true"

- x is a path in a transition system
- p is a temporal logic predicate •

- Example: A x. $X \models G\ (\neg(pc1=12 \wedge pc2=22))$

Next Class

PennState

- Linear Temporal Logic (LTL)
- Computation Tree Logic (CTL)
- SAT/SMT Solver
- Model Checker with NuXMV

Thanks

Thanks to Bor-Yuh Evan Chang for some slides.