Towards Formalising Sustainable Security Systems

Liliana Pasquale

9th Conference on Formal Methods in Software Engineering (FormaliSE 2021)
Agenda

1. Introduction to Sustainable Security
2. 3 Key Ideas to Support Sustainable Security
3. A Sustainable Security Roadmap
1 Introduction to Sustainable Security
Sustainability in Software Engineering

Software sustainability has been generally considered as the capacity of a software system to endure [Venters et al., 2017]

Sustainable Software:
[Beckers et al., 2015]

Software Engineering for Sustainability:
[Beckers et al., 2015]
Sustainability in Software Engineering

Software sustainability has been generally considered as the capacity of a software system to endure [Venters et al., 2017]

Sustainable Software: principles, practices and process that contribute to software endurance [Beckers et al., 2015]

Software Engineering for Sustainability: [Beckers et al., 2015]
Sustainability in Software Engineering

Software sustainability has been generally considered as the capacity of a software system to endure [Venters et al., 2017]

**Sustainable Software:** principles, practices and process that contribute to software endurance  
[Beckers et al., 2015]

**Software Engineering for Sustainability:** Building software systems that support one or more dimensions of sustainability  
- Environmental  
- Economical  
- Individual  
- Societal  
- Technical.
Sustainable Security

Software sustainability has been generally considered as the capacity of a software system to endure [Venters et al., 2017]

**Sustainable Software**: principles, practices and process that contribute to software endurance [Beckers et al., 2015]
Sustainable Security

Capacity of Software to Endure Satisfaction of Security Requirements

Security  Autonomy

Ensuring Sustainable Security in Cyber-Physical Systems is Challenging!
Warehouse of the zefirP Company

- Truck Bay 1 (Incoming)
- Truck Bay 2 (Outgoing)
- Inventory DB
- Office
- Receiving
- Drone
- Mobile HMI
- Human Operator
- Packing
- Loading
- Storage Zone 1
- Storage Zone 2
- Storage Zone 3
- RDU
Warehouse of the zefirP Company

- Truck Bay 1 (Incoming)
- Truck Bay 2 (Outgoing)
- Receiving
- Storage Zone 1
- Storage Zone 2
- Storage Zone 3
- Packing
- Loading
- Office
- Inventory DB
- Drone
- Mobile HMI
- Human Operator
- RDU
- Warehouse of the zefirP Company
Warehouse of the zefirP Company

- Truck Bay 1 (Incoming)
- Inventory DB
- Office
- Truck Bay 2 (Outgoing)
- Receiving
- Drone
- Mobile HMI
- Human Operator
- Loading
- Packing
- Storage Zone 1
- Storage Zone 2
- Storage Zone 3
- RDU

Warehouse of the zefirP Company
Warehouse of the zefirP Company

- Truck Bay 1 (Incoming)
- Truck Bay 2 (Outgoing)
- Receiving
- Office
- Inventory DB
- Loading
- Packing
- Human Operator
- Mobile HMI
- Drone
- Storage Zone 1
- Storage Zone 2
- Storage Zone 3
- RDU
Warehouse of the zefirP Company

- Truck Bay 1 (Incoming)
- Truck Bay 2 (Outgoing)
- Inventory DB
- Office
- Receiving
- Loading
- Packing
- Mobile HMI
- Human Operator
- Storage Zone 1
- Storage Zone 2
- Storage Zone 3
- Drone
- RDU
Sustainable Security

Capacity of Software to Continuously Satisfy Security Requirements

Security Autonomy

- Reason about and counteract threats brought by an extended attack surface
Extended Attack Surface: Cyber-Physical Threats
Extended Attack Surface: Cyber-Physical Threats
Extended Attack Surface: Physical-Cyber Threats
Sustainable Security

Capacity of Software to Continuously Satisfy Security Requirements

Security  Autonomy

- Reason about and counteract threats brought by an extended attack surface
- Handle uncertainties brought by unexpected threats
Uncertainty: Unexpected Threats

Office

Receiving

Drone

Storage Zone 1

RDU

Engineering Workstation

Machine Programmer

Mobile HMI

Human Operator

Loading

Packing

Truck Bay 1 (Incoming)

Truck Bay 2 (Outgoing)

Engineering Workstation

Machine Programmer

Human Operator

Mobile HMI

RDU

Storage Zone 1

Storage Zone 2

Storage Zone 3

Truck Bay 1 (Incoming)

Truck Bay 2 (Outgoing)
Sustainable Security

Capacity of Software to Continuously Satisfy Security Requirements

Security • Autonomy

- Reason about and counteract threats brought by an extended attack surface
- Handle uncertainties brought by unexpected threats
- Endure stakeholders’ engagement
Stakeholders: Human Operators need Explanations
Sustainable Security

Capacity of Software to Continuously Satisfy Security Requirements

- Reason about and counteract threats brought by an extended attack surface
- Handle uncertainties brought by unexpected threats
- Endure stakeholders’ engagement
Adaptive Security

Sustainable Security
3 Key Ideas to Support Sustainable Security

• Formalize and reason about the extended attack surface

• Discover and counteract new threats

• Provide explanations to human operators
3 Key Ideas to Support Sustainable Security

• Formalize and reason about the extended attack surface

• Discover and counteract new threats

• Provide explanations to human operators
Topology

**Location**
of objects and agents

**Structure** of space

- Proximity
- Reachability
Topology

Location of objects and agents

Structure of space

- Proximity
- Reachability
Topological representation of a warehouse setting. The warehouse has three storage zones labeled as Storage Zone 1, Storage Zone 2, and Storage Zone 3. There are two truck bays: Truck Bay 1 (Incoming) and Truck Bay 2 (Outgoing). The office is connected to all areas. The RDU (Remote Desktop Unit) is located in Storage Zone 2, and there is a drone for aerial operations. The connectivity of physical areas is illustrated with arrows and labels for proximity and reachability. The structure of space includes location of objects and agents, with a focus on inventory database (Inventory DB) and connectivity of physical areas.
Topology

Location of objects and agents

Structure of space

- Proximity
- Reachability

Connectivity of digital areas.

Office

Receiving

Drone

Storage Zone 1

Inventory DB

Loading

Human Operator

Mobile HMI

RDU

Packing

Truck Bay 1 (Incoming)

Truck Bay 2 (Outgoing)

Storage Zone 2

Storage Zone 3

Connectivity of digital areas.

• Proximity
• Reachability

Structure of space

Location of objects and agents
Topology

Location of objects and agents

Structure of space

- Proximity
- Reachability

Placement of objects and agents.

Connectivity of physical/digital areas.

- Truck Bay 1 (Incoming)
- Truck Bay 2 (Outgoing)
- Office
- Inventory DB

- Receiving
- Drone
- Mobile HMI
- Human Operator

- Packing
- RDU

Storage Zones

- Storage Zone 1
- Storage Zone 2
- Storage Zone 3
**Topology**

- **Location** of objects and agents
- **Structure** of space
- **Placement** of objects and agents.
- **Connectivity** of physical/digital areas.
- **Proximity** and **Reachability**
- **Proximity** Colocation in the same area.

Diagram showing the structure of the space with locations such as Truck Bay 1 (Incoming), Truck Bay 2 (Outgoing), Office, Receiving, Drone, Inventory DB, Human Operator, Mobile HMI, Packing, Loading, RDU, Storage Zone 1, Storage Zone 2, and Storage Zone 3.
Topology

Location
of objects and agents

Structure
of space

Placement
of physical objects and agents.

Connectivity
of physical/digital areas.

Proximity
Colocation in the same area.

Reachability
Accessibility to physical areas or objects.

Proximity
Reachability

• Proximity
• Reachability

Connectivity

Office

Inventory DB

Truck Bay 1 (Incoming)

Truck Bay 2 (Outgoing)

Receiving

Drone

Loading

Packing

Human Operator

Mobile HMI

Storage Zone 1

Storage Zone 2

Storage Zone 3

RDU

Structure

Location

Placement

Connectivity

Reachability

Proximity

Topology

Human

Mobile HMI

Drone

RDU

Inventory DB

Storage Zone 1

Storage Zone 2

Storage Zone 3

Office

Truck Bay 1 (Incoming)

Truck Bay 2 (Outgoing)

Receiving

Drone

Loading

Packing

Human Operator

Mobile HMI

RDU

Inventory DB

Connectivity

Proximity

Reachability
**Topology**

- **Location** of objects and agents
- **Structure** of space
- **Placement** of physical objects and agents.
- **Connectivity** of physical/digital areas.
- **Proximity**
- **Reachability**

**Reachability**  
Accessibility to digital areas or objects.

**Proximity**  
Colocation in the same area.

- **Office**
- **Receiving**
- **Drone**
- **Mobile HMI**
- **Human Operator**
- **Inventory DB**
- **Packing**
- **Loading**
- **Truck Bay 1 (Incoming)**
- **Truck Bay 2 (Outgoing)**
- **Storage Zone 1**
- **Storage Zone 2**
- **Storage Zone 3**
- **RDU**
Requirements for Modelling Topology

• Represent structure and communication
• Enable reasoning about the effects of topological changes

Process Calculi

• $\pi$-calculus
• Ambient Calculus [Cardelli & Gordon ‘98, Tsigkanos et al. ‘14]

• Bigraphical Reactive systems (BRS) [Milner ‘09]
  – Extend bigraphs with well defined semantics of dynamic behaviour.
Bigraphs

**Place graph**
- A forest of trees
- Nesting

**Link graph**
- A hypergraph of named edges over the set of nodes of the place graph
- Many-to-many relationships among nodes
Bigraphs – Algebraic Notation

\[ P.Q \quad \text{Nesting (P contains Q)} \]
\[ -i \quad \text{Site numbered i} \]
\[ K \xrightarrow{w}.(U) \quad \text{Node associated with control K having ports with names in vector w. K contains U} \]
\[ /x.U \quad \text{Ports with name x in U are connected} \]
\[ P \parallel Q \quad \text{Juxtaposition of roots} \]
\[ P \circ Q \quad \text{Composition} \]
\[ P | Q \quad \text{Juxtaposition of children} \]
Bigraphs – Algebraic Notation

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P.Q$</td>
<td>Nesting ($P$ contains $Q$)</td>
</tr>
<tr>
<td>$\neg i$</td>
<td>Site numbered $i$</td>
</tr>
<tr>
<td>$K \over(\overrightarrow{w}) \cdot (U)$</td>
<td>Node associated with control $K$ having ports with names in vector $w$. $K$ contains $U$</td>
</tr>
<tr>
<td>$\mathit{x} \cdot U$</td>
<td>Ports with name $x$ in $U$ are connected</td>
</tr>
<tr>
<td>$P \parallel Q$</td>
<td>Juxtaposition of roots</td>
</tr>
<tr>
<td>$P \circ Q$</td>
<td>Composition</td>
</tr>
<tr>
<td>$P \mid Q$</td>
<td>Juxtaposition of children</td>
</tr>
</tbody>
</table>

\[(\text{Room}_\text{Office} \cdot (\text{Agent}_{\text{Mallory}} \mid \text{HMI}_\text{wlink})) \parallel (\text{Room}_\text{wifiarea} \cdot (\text{Wifi}_\text{wlink,lan} \mid \text{(-1)}))\]

\[\parallel ((\text{Room}_\text{StgZone2,wlink} \cdot (\text{RDU}_1,wlink \cdot (\text{Vaccine}))) \mid \text{Room}_\text{StgZone3} \cdot \text{(Cooled)})\]
Bigraphs – Algebraic Notation

\[ P.Q \] Nesting (P contains Q)

\[-i\] Site numbered i

\[ K \leftarrow \{U\} \] Node associated with control K having ports
with names in vector w. K contains U

\[/x.U\] Ports with name x in U are connected

\[ P \parallel Q \] Juxtaposition of roots

\[ P \circ Q \] Composition

\[ P \mid Q \] Juxtaposition of children

\[ (\text{RoomOffice} \cdot (\text{Agent}_{\text{Mallory}} \mid HMI_{\text{wlink}})) \parallel (\text{Room}_{\text{wifiarea}} \cdot (\text{Wifi}_{\text{wlink,lan}} \mid (-1))) \parallel ((\text{Room}_{\text{StgZone2,wlink}} \cdot (\text{RDU}_{1,wlink} \cdot (\text{Vaccine}))) \mid \text{Room}_{\text{StgZone3}} \cdot (\text{Cooled})) \]
Bigraphs – Algebraic Notation

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(P.Q)</td>
<td>Nesting ((P) contains (Q))</td>
</tr>
<tr>
<td>(-i)</td>
<td>Site numbered (i)</td>
</tr>
<tr>
<td>(K_{\vec{w}}(U))</td>
<td>Node associated with control (K) having ports with names in vector (w). (K) contains (U)</td>
</tr>
<tr>
<td>(/x.U)</td>
<td>Ports with name (x) in (U) are connected</td>
</tr>
<tr>
<td>(P \parallel Q)</td>
<td>Juxtaposition of roots</td>
</tr>
<tr>
<td>(P \circ Q)</td>
<td>Composition</td>
</tr>
<tr>
<td>(P \mid Q)</td>
<td>Juxtaposition of children</td>
</tr>
</tbody>
</table>

\[
\left(\text{Room}_{\text{Office}} \cdot (\text{Agent}_{\text{Mallory}} \mid \text{HMI}_{\text{wlink}})\right) \parallel \left(\text{Room}_{\text{wifiarea}} \cdot (\text{Wifi}_{\text{wlink,lan}} \mid (-1))\right) \\
\parallel \left(\left(\text{Room}_{\text{StgZone2,wlink}} \cdot (\text{RDU}_{1,wlink} \cdot (\text{Vaccine}))\right) \mid \text{Room}_{\text{StgZone3}} \cdot (\text{Cooled})\right)
\]
## Bigraphs – Algebraic Notation

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P.Q$</td>
<td>Nesting ($P$ contains $Q$)</td>
</tr>
<tr>
<td>$\neg i$</td>
<td>Site numbered $i$</td>
</tr>
<tr>
<td>$K_{\overline{w}}.(U)$</td>
<td>Node associated with control $K$ having ports with names in vector $w$. $K$ contains $U$</td>
</tr>
<tr>
<td>$/x.U$</td>
<td>Ports with name $x$ in $U$ are connected</td>
</tr>
<tr>
<td>$P \parallel Q$</td>
<td>Juxtaposition of roots</td>
</tr>
<tr>
<td>$P \circ Q$</td>
<td>Composition</td>
</tr>
<tr>
<td>$P \mid Q$</td>
<td>Juxtaposition of children</td>
</tr>
</tbody>
</table>

$$
\begin{align*}
& (\text{Room}_{\text{Office}} \ (\text{Agent}_{\text{Mallory}} \mid \text{HMI}_{\text{wlk}})) \parallel (\text{Room}_{\text{wifiarea}} \cdot (\text{Wifi}_{\text{wlk,lan}} \mid (-1))) \\
& \parallel ((\text{Room}_{\text{StgZone2},\text{wlk}} \cdot (\text{RDU}_{1,\text{wlk}} \cdot (\text{Vaccine})))) \mid \text{Room}_{\text{StgZone3}} \cdot (\text{Cooled})
\end{align*}
$$
Dynamic Behaviour

**Reaction Rules** \( (R \rightarrow R') \)

A portion of the bigraph matching a redex \( R \) is rewritten as the reactum \( R' \).

**For Example:** Action *enter_room*

![Diagram showing an example of a reaction rule and its corresponding reactum. The left side shows the redex with an agent in room 0 and room 1 being matched. The right side shows the reactum with the agent moving to room 1.](https://via.placeholder.com/150)

\[ Agent_{n \cdot 0} \mid Room_{r \cdot 1} \mid _2 \quad Room_{r \cdot (Agent_{n \cdot 0} \mid _1)} \mid _2 \]
Specifying Security Requirements

A topological configuration described by a bigraph $C$ satisfies a property if the bigraph specifying the property can be matched against $C$.

**Example:** Violation of the vaccine integrity (SR1)

An RDU transporting the vaccine is in a storage zone that is locked and not cooled

$$\text{Room}_x. (\text{RDU}_y. (\text{Vaccine}) \mid \text{Locked})$$

**Security Requirements:** Branching Time Temporal Logic (CTL).

$$AG(\neg(SR1))$$

Enabling Automated Reasoning

The BRS-based specification is transformed into an equivalent Labelled Transition System (LTS).

Each LTS state represents a different bigraph configuration.

Each LTS transition represents a different application of the reaction rules leading to new configurations.

Enabling Automated Reasoning

The process is iterated by exploring all configurations and generating new LTS states accordingly.

Speculative Threat Analysis

• Identifies potential violations of security requirements that take place in future evolutions of the cyber-physical space.

• Interpret the BRS over an LTS

• Perform explicit state model checking to discover LTS states representing violations

Computing Security Controls

When a state (P) immediately before a violating state is entered

- Disable all actions in P leading to a violating states OR

---

Computing Security Controls

When a state (P) immediately before a violating state is entered

- Disable all actions in P leading to a violating states OR
- Enforce the execution of an action that can lead to a safe state

Computing Security Controls

When a violating state (V) is reached

• Disable all transitions in V leading to a violating states AND
• Enforce the execution of a transition(s) leading to a safe state

3 Key Ideas to Support Sustainable Security

• Formalize and reason about the extended attack surface

• Discover and counteract new threats

• Provide explanations to human operators
Lifecycle for Handling Unknown Threats

- Vulnerability
- Environment Change

Update → Diagnosis → Threat Analysis → Identification of Security Controls

Revision → Diagnosis → Update
Lifecycle for Handling Unknown Threats

Analysis → Diagnosis → Revision → Analysis

D, P → Diagnosis → Revision → D, P
Evolving Requirement Specifications

Abductive Reasoning

In event-driven system descriptions, abduction would be used to identify a trace of events and system transitions (starting from the initial state) that would prove a given requirement.
Abductive Learning

• In refutation mode, abduction allows the generation of counter-examples (incorrect system transitions) as diagnostic information of properties violation.

• If the abductive procedure finds such a set (of incorrect state transitions), then acts as a set of counter-examples to the validity of P.
Abductive Learning - Example

Example described above formalized using propositional logic programming:

D: \( In(RDU, \text{Room1}) \land Connected(\text{Room1}, \text{Room2}) \land Enter(RDU, \text{Room2}) \rightarrow In(RDU, \text{Room2}) \)

P: \( In(\text{DangMaterial}, \text{Room}) \rightarrow \neg(RDU, \text{Room}) \)
Abductive Learning - Example

Example described above formalized using propositional logic programming:

D: \( \text{In}(RDU, Room1) \land \text{Connected}(Room1, Room2) \land \text{Enter}(RDU, Room2) \rightarrow \text{In}(RDU, Room2) \)

P: \( \text{In}(\text{DangMaterial}, Room) \rightarrow \neg(RDU, Room) \)
Inductive Learning

$$D \cup h \vdash \Delta^+$$

$$D \cup h \not\vdash \Delta^-$$

Aims to find hypotheses, in the form of rules, that are consistent with the description of the system (background knowledge) to explain a given set of examples.
Dear Search Committee,

I am currently a postdoctoral researcher working with Prof. Bashar Nuseibeh at Lero, the Irish Software Engineering Research Centre, in Limerick (Ireland). I received my Ph.D. in Computer Science from Politecnico di Milano in 2011, and I am applying for a tenure-track faculty position in your department.

In particular, my work has focused on using runtime requirements models to enable self-adaptive software. In particular, my work has focused on using runtime requirements models to enable self-adaptive software. In particular, my work has focused on using runtime requirements models to enable self-adaptive software. In particular, my work has focused on using runtime requirements models to enable self-adaptive software.

\[ D \cup h \vdash \Delta^+ \]
\[ D \cup h \not\vdash \Delta^- \]

\[ \Delta^+ = \{ \text{In}(RDU, \text{Room } 1), \text{Connected}(\text{Room } 1, \text{Room } 2), \text{Enter}(RDU, \text{Room } 2), \text{In}'(RDU, \text{Room } 2) \} \]

\[ \Delta^- = \text{In}(RDU, \text{Room } 1), \text{Connected}(\text{Room } 1, \text{Room } 2), \text{Enter}(RDU, \text{Room } 2), \text{In}(\text{DangMaterial}, \text{Room } 2), \text{In}'(RDU, \text{Room } 2) \} \]

\[ D' : \quad \text{In}(RDU, \text{Room } 1) \wedge \text{Connected}(\text{Room } 1, \text{Room } 2) \wedge \text{Enter}(RDU, \text{Room } 2) \wedge \neg \text{In}(\text{DangMaterial}, \text{Room } 2) \rightarrow \]

\[ \text{In}(RDU, \text{Room } 2) \]
2 3 Key Ideas to Support Sustainable Security

- Formalize and reason about the extended attack surface
- Discover and counteract new threats
- Provide explanations to human operators
**Objective**

Providing explanations about why a system produces a certain behaviour

- E.g., justifying why a certain security requirement is violated

Providing a human operator with a full model of the system, the operating environment and their current state is infeasible due to information overload.
Abstraction is Key

Abstraction can be used to reduce the complexity of the model

- Provide a high level of detail for the aspects of the system that affect satisfaction of some requirements of interest, while leaving irrelevant aspects under-specified.

- Aligned with the principles of Situation Awareness Oriented Design

- Can facilitate decision-making.

- Abstraction has been used in previous work to provide explanations about machine interfaces [Combefis et al. 2011]
Abstraction is Key

Abstraction can be used to reduce the complexity of the model

- Provide a high level of detail for the aspects of the system that affect satisfaction of some requirements of interest, while leaving irrelevant aspects under-specified.

- Abstraction has been used in previous work to provide explanations about machine interfaces [Combefis et al. 2011]
Example – Abstract State Machines (ASM)

• **ASM**: A set of rules that are conditioned by, and may generate updates for, states.
• **State** => defined as a set of locations
• **Location** => identified by a function symbol and a list of parameters associated with values
Example – Abstract State Machines (ASM)

- **ASM**: A set of rules that are conditioned by, and may generate updates for, states.
- **State** => defined as a set of locations
- **Location** => identified by a function symbol and a list of parameters associated with values

System Model

```
forall r in Rooms
  if curtemp(r) > destemp(r) + h then aircond(r) := on
  if curtemp(r) < destemp(r) - h then aircond(r) := off
  if curtemp(r) > destemp(r) + k then heating(r) := off
  if curtemp(r) < destemp(r) - k then heating(r) := on
  if presence(r) then lights(r) := on
```
Example – Abstract State Machines (ASM)

- **ASM**: A set of rules that are conditioned by, and may generate updates for, states.
- **State** => defined as a set of locations
- **Location** => identified by a function symbol and a list of parameters associated with values

**System Model**

```
forall r in Rooms
  if curtemp(r) > destemp(r) + h then aircond(r) := on
  if curtemp(r) < destemp(r) - h then aircond(r) := off
  if curtemp(r) > destemp(r) + k then heating(r) := off
  if curtemp(r) < destemp(r) - k then heating(r) := on
  if presence(r) then lights(r) := on
```

**Constraints**

- \( c_1 \): \( \forall r \in \text{Rooms}, \neg (\text{aircond}(r) = \text{on} \land \text{heating}(r) = \text{on}) \)
- \( c_2 \): \( \forall r \in \text{Rooms}, \text{window}(r) = \text{open} \Rightarrow (\text{aircond}(r) = \text{off} \land \text{heating}(r) = \text{off}) \)
- \( c_3 \): \( \forall r \in \text{Rooms} \setminus \text{Halls}, \neg \text{presence}(r) \Rightarrow \text{lights}(r) = \text{off} \)
Focus on Interesting Variables

c_1) \forall r \in Rooms, \neg (\text{aircond}(r) = \text{on} \land \text{heating}(r) = \text{on})

for all r in Rooms
  if \text{curtemp}(r) > \text{destemp}(r) + h then \text{aircond}(r) := \text{on}
  if \text{curtemp}(r) < \text{destemp}(r) - h then \text{aircond}(r) := \text{off}
  if \text{curtemp}(r) > \text{destemp}(r) + k then \text{heating}(r) := \text{off}
  if \text{curtemp}(r) < \text{destemp}(r) - k then \text{heating}(r) := \text{on}
  if \text{presence}(r) then \text{lights}(r) := \text{on}

for all r in \{q\}
  if \text{curtemp}(r) > \text{destemp}(r) + h \ldots
Focus on Interesting Locations

$c_1) \forall r \in \text{Rooms}, \neg(\text{aircond}(r) = \text{on} \land \text{heating}(r) = \text{on})$

The only locations affecting the violated constraint are $\text{aircond}(q)$ and $\text{heating}(q)$

---

**Step 1**

\[
\begin{align*}
\text{forall } r \in \{q\} & \\
\text{if } \text{curtemp}(r) > \text{destemp}(r) + h \ldots
\end{align*}
\]

---

**Step 2**

\[
\begin{align*}
\text{if } \text{curtemp}(q) > \text{destemp}(q) + h & \text{ then } \text{aircond}(q) := \text{on} \\
\text{if } \text{curtemp}(q) < \text{destemp}(q) - h & \text{ then } \text{aircond}(q) := \text{off} \\
\text{if } \text{curtemp}(q) > \text{destemp}(q) + k & \text{ then } \text{heating}(q) := \text{off} \\
\text{if } \text{curtemp}(q) < \text{destemp}(q) - k & \text{ then } \text{heating}(q) := \text{on}
\end{align*}
\]
Focus on Interesting Value

The values of aircond(r) and heating(r) are both on

We can splice the model considering 3 sets of rules:

• **R1**: those that set one of the interesting locations to the value observed in the state at time of violation
• **R2**: those that set one of the interesting locations to a value different from what has been observed in the state at the time of violation
• **R3**: those that do not update the interesting locations

\[ c_1 \) \forall r \in \text{Rooms}, \neg(\text{aircond}(r) = \text{on} \land \text{heating}(r) = \text{on}) \]

\[
\begin{align*}
\text{if } \text{curtemp}(q) > \text{destemp}(q) + h \text{ then } \text{aircond}(q) := \text{on} \\
\text{if } \text{curtemp}(q) < \text{destemp}(q) - h \text{ then } \text{aircond}(q) := \text{off} \\
\text{if } \text{curtemp}(q) > \text{destemp}(q) + k \text{ then } \text{heating}(q) := \text{off} \\
\text{if } \text{curtemp}(q) < \text{destemp}(q) - k \text{ then } \text{heating}(q) := \text{on}
\end{align*}
\]
Focus on Interesting Value

$c_1) \forall r \in \text{Rooms}, \neg(\text{aircond}(r) = \text{on} \land \text{heating}(r) = \text{on})$

The values of aircond(r) and heating(r) are both on

We can splice the model considering 3 sets of rules:

- **R1**: those that set one of the interesting locations to the value observed in the state at time of violation
- **R2**: those that set one of the interesting locations to a value different from what has been observed in the state at the time of violation
- **R3**: those that do not update the interesting locations

\[
\begin{align*}
\text{if } \text{curtemp}(q) > \text{destemp}(q) + h & \text{ then } \text{aircond}(q) := \text{on} \\
\text{if } \text{curtemp}(q) < \text{destemp}(q) - h & \text{ then } \text{aircond}(q) := \text{off} \\
\text{if } \text{curtemp}(q) > \text{destemp}(q) + k & \text{ then } \text{heating}(q) := \text{off} \\
\text{if } \text{curtemp}(q) < \text{destemp}(q) - k & \text{ then } \text{heating}(q) := \text{on}
\end{align*}
\]
Focus on Interesting Value

c_1) \forall r \in \text{Rooms}, \neg(\text{aircond}(r) = \text{on} \land \text{heating}(r) = \text{on})

The values of \text{aircond}(r) and \text{heating}(r) are both on

We can splice the model considering 3 sets of rules:

• **R1**: those that set one of the interesting locations to the value observed in the state at time of violation
• **R2**: those that set one of the interesting locations to a value different from what has been observed in the state at the time of violation
• **R3**: those that do not update the interesting locations

---

```plaintext
if curtemp(q) > destemp(q) + h then aircond(q) := on
if curtemp(q) < destemp(q) - h then aircond(q) := off
if curtemp(q) > destemp(q) + k then heating(q) := off
if curtemp(q) < destemp(q) - k then heating(q) := on
```
Focus on Interesting Value

\( c_1 \) \( \forall r \in \text{Rooms}, \neg(\text{aircond}(r) = \text{on} \land \text{heating}(r) = \text{on}) \)

The values of aircond\( (r) \) and heating\( (r) \) are both on

We can splice the model considering 3 sets of rules:

- **R1**: those that set one of the interesting locations to the value observed in the state at time of violation
- **R2**: those that set one of the interesting locations to a value different from what has been observed in the state at the time of violation
- **R3**: those that do not update the interesting locations

---

\[
\begin{align*}
\text{curtemp}(q) &> \text{desttemp}(q) + h \\
\text{curtemp}(q) &< \text{desttemp}(q) - k \\
\text{curtemp}(q) &\geq \text{desttemp}(q) - h \\
\text{curtemp}(q) &\leq \text{desttemp}(q) + k
\end{align*}
\]
3 A Sustainable Security Roadmap
Formalize and reason about the extended attack surface

• Exploit possible compositionality of security properties

• Explore model-based diagnosis techniques (e.g., hierarchical diagnosis [Mozetič, 1991][Siddiqi, 2007]), widely explored in control theory to reduce the computational complexity of the diagnosis.
Discover and Counteract New Threats

Continuous Threat Analysis

Digital Twin

Identification of Security Controls

Threat Analysis

Vulnerability discovery

Enactment of Security Controls

Real System
Provide Explanations to Human Operators

• Formalize and create a collection of abstraction strategy that can be systematically selected to provide explanations

• Select a suitable level of abstraction depending on:
  - the cognitive abilities of the human operator
  - the time available to a human operator to make a decision
References


THANK YOU!