# *Heimdall:* Towards Risk-Aware Network Management Outsourcing

Yuejie Wang[1], **Qiutong Men[2]**, Yongting Chen[3], Jiajin Liu[3], Gengyu Chen[4], Ying Zhang[5], Guyue Liu[1], Vyas Sekar[4]

[1]Peking University, [2]New York University, [3]New York University Shanghai, [4]Carnegie Mellon University, [5]Meta

# Network Management is Now Outsourced

- Outsourcing network management is cheaper than running it in-house.

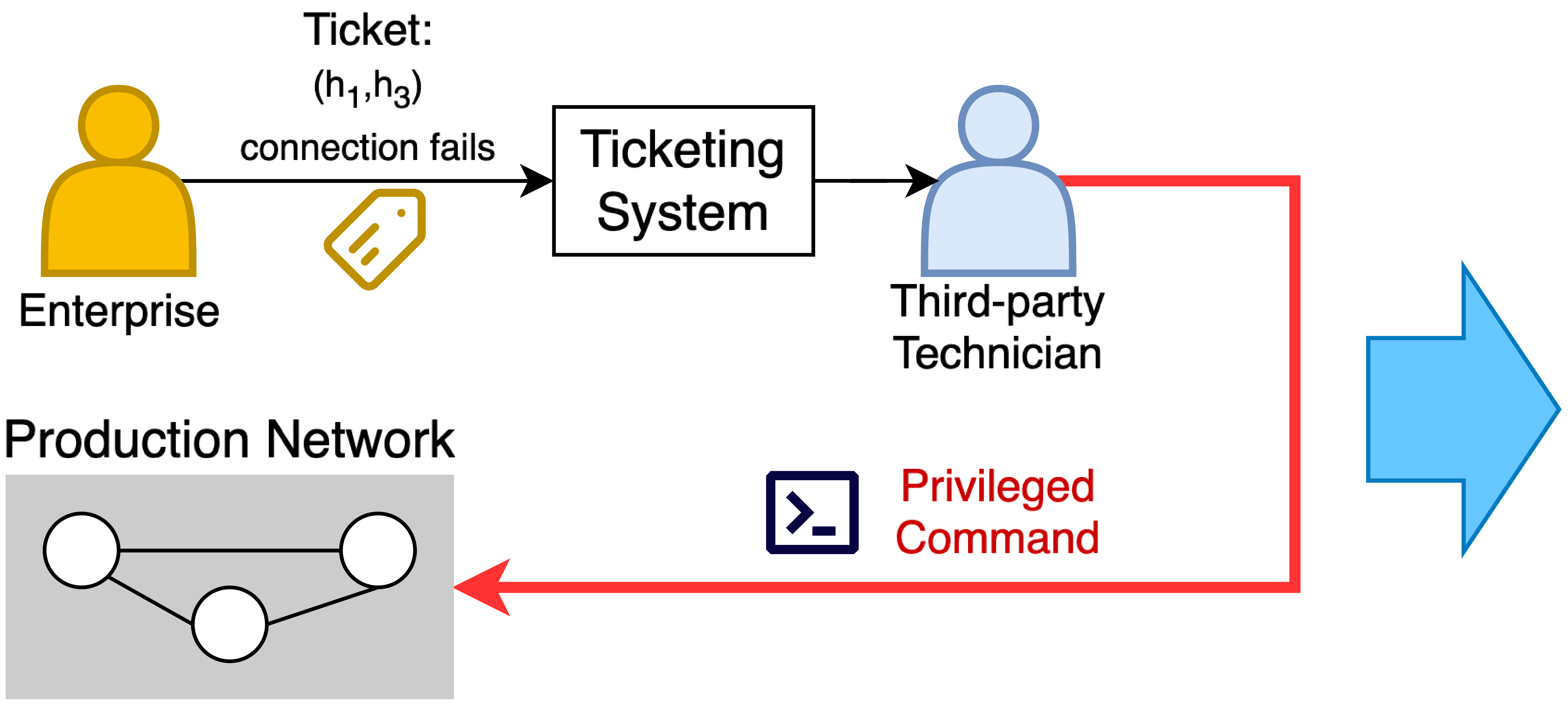
# Network Management is Now Outsourced

- Outsourcing network management is cheaper than running it in-house.
- “Managed Services Market Size to Reach USD 309.4 Billion by 2025”.

# Network Management is Now Outsourced

- Outsourcing network management is cheaper than running it in-house.
- “Managed Services Market Size to Reach USD 309.4 Billion by 2025”.
- Provided by many companies, including Verizon, Fujitsu, and IBM.

# Raises Significant Security Concern



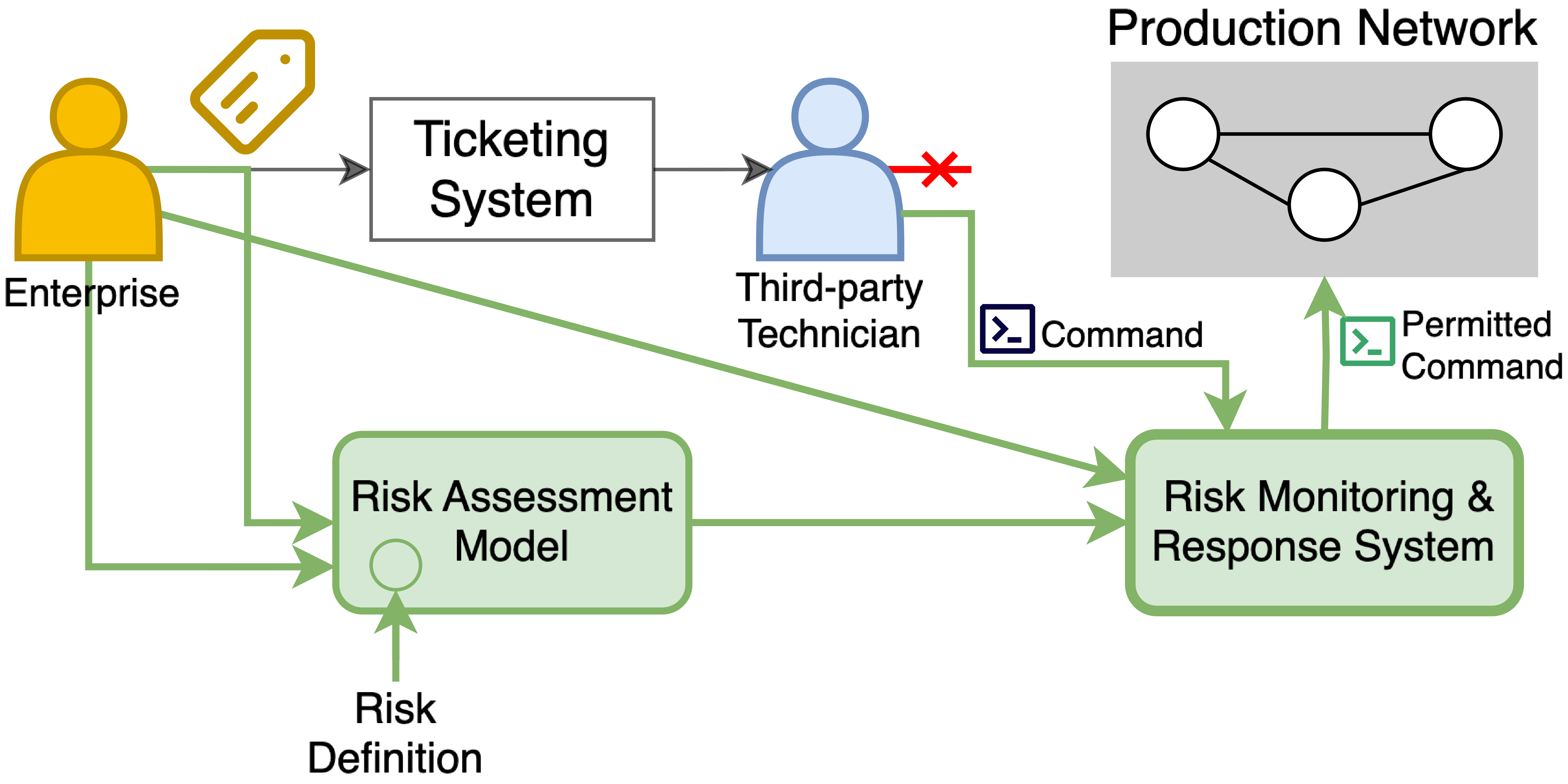
Improper Privilege

Ransomware Injection,
Service Degradation,
Large-scale service outage,
…

!

# We need ***risk-aware network management outsourcing***

- Our goal: defining and calculating the risk of changes to the network, and guarding the network through risk monitoring and response



# Threat Model

- **Users** do not participate in network management
- **Administrator** is trustworthy
- **Technician** has expertise but could be the adversarial

# Outline

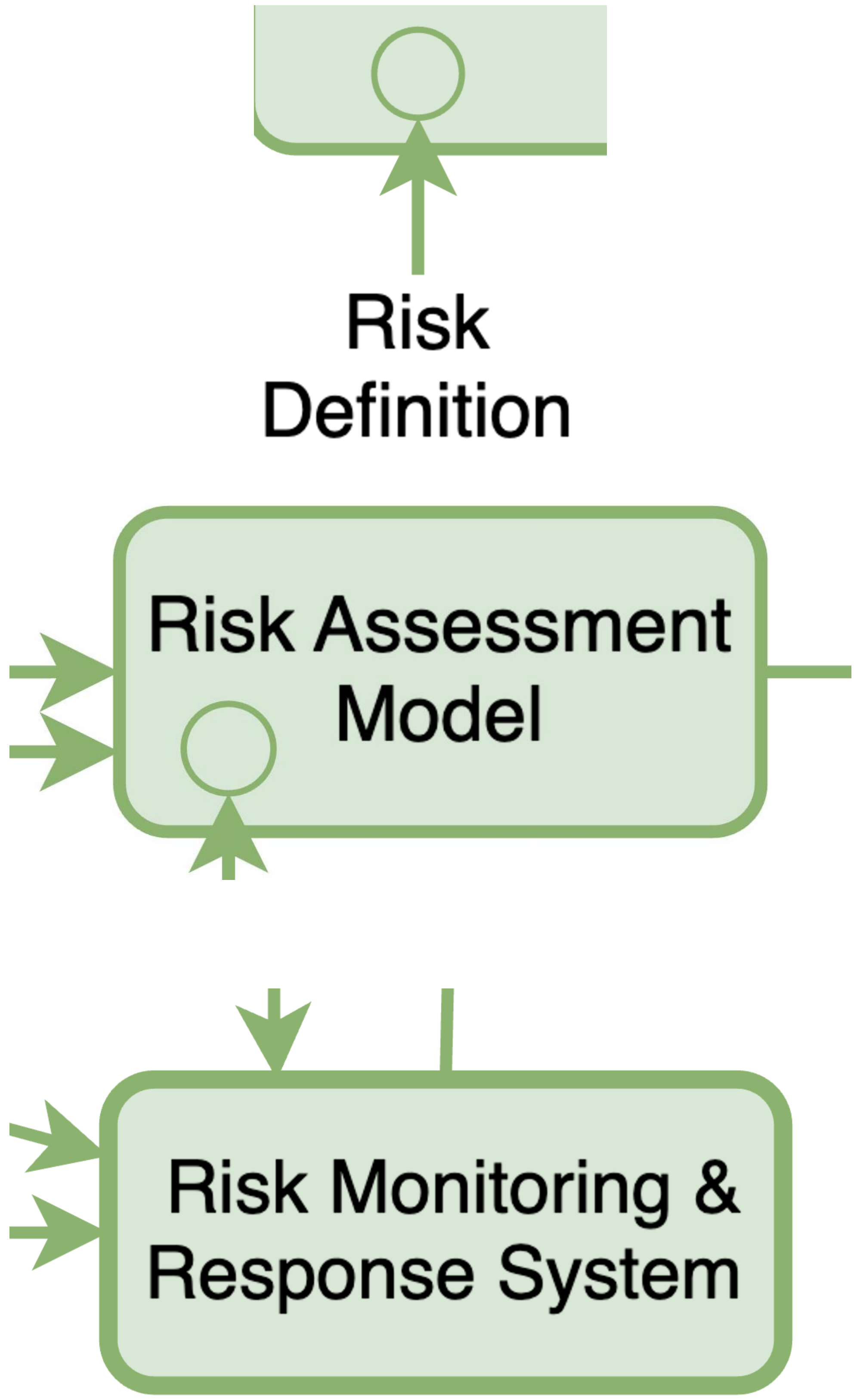
- Inputs
- Workflow
- Evaluation
- Conclusion

# Main Components to Build a Risk-aware Management Framework

- ***Risk Definition***: quantitating the potential impact from value perspective

- ***Risk Assessment***: calculating the impact accurately and efficiently

- ***Risk Monitoring and Response***: dynamically enforcing policies based on the real-time assessment of risks



# How do we define risk today?

- **Risk is associated with individual commands**
  - `shutdown` vs. `show`

# How do we define risk today?

- **Risk is associated with individual commands**
  - `shutdown` vs. `show`
- **Not Flexible**
  - The same command may imply different risks under various configurations (e.g. ospf areas)

# How do we define risk today?

- **Risk is associated with individual commands**
  - `shutdown` vs. `show`
- **Not Flexible**
  - The same command may imply different risks under various configurations (e.g. ospf areas)
- **Coarse-grained**
  - Cisco IOS supports only up to 16 privilege level for commands

# Our Asset-based Quantitative Risk Model

- Fundamental principle: $Risk(E) = \mathbb{P}(E) * \mathbb{C}(E)$
  - Risk = Probability of Occurrence * Consequence

# Our Asset-based Quantitative Risk Model

- Fundamental principle: $Risk(E) = \mathbb{P}(E) * \mathbb{C}(E)$
    - Risk = Probability of Occurrence * Consequence
- ***Assets*** are the primary concerns of an enterprise, including physical equipment and software
- ***Ticket*** resolution projects to a series of events modifying network configurations

# Our Asset-based Quantitative Risk Model

- Consider a ticket $T$.
- $\mathbb{P}(S|T) \coloneqq$ probability asset $S$ can be affected during resolving $T$
- We calculate risk of a ticket based on assets value $S.value$
- Then:

$$Risk(T) \coloneqq \sum_{S \in Assets} \mathbb{P}(S|T) * S.value$$

# How to accurately assess risk of a ticket?

- **Token-matching on config files[1]: hard to capture precises consequence dependency**
  - May falsely link all interfaces within a router, then spread exponentially to neighbor interfaces.
  - Results in a dense risk consequence graph difficult to reason about

[1] Theophilus Benson, Aditya Akella, and David Maltz. 2009. Unraveling the complexity of network management. In Proceedings of the 6th USENIX symposium on Networked systems design and implementation (NSDI'09). USENIX Association, USA, 335–348.

# How to accurately assess risk of a ticket?

- **Token-matching on config files[1]: hard to capture precises consequence dependency**
  - May falsely link all interfaces within a router, then spread exponentially to neighbor interfaces.
  - Results in a dense risk consequence graph difficult to reason about
- **Our approach:**
  - **Construct dependency graph leveraging data-plane information**
  - **Consequence probability with preference order and root cause estimation**

[1] Theophilus Benson, Aditya Akella, and David Maltz. 2009. Unraveling the complexity of network management. In Proceedings of the 6th USENIX symposium on Networked systems design and implementation (NSDI'09). USENIX Association, USA, 335–348.

# How risky is a privileged command?

- **Verification of config changes: correctness is hard to specify**
- SecGuru[1] finds in datacenter network, most firewall rules are redundant

[1] Karthick Jayaraman, Nikolaj Bjørner, Jitu Padhye, Amar Agrawal, Ashish Bhargava, Paul-Andre C Bissonnette, Shane Foster, Andrew Helwer, Mark Kasten, Ivan Lee, Anup Namdhari, Haseeb Niaz, Aniruddha Parkhi, Hanukumar Pinnamraju, Adrian Power, Neha Milind Raje, and Parag Sharma. 2019. Validating datacenters at scale. In Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM '19). Association for Computing Machinery, New York, NY, USA, 200–213. https://doi.org/10.1145/3341302.3342094

# How risky is a privileged command?

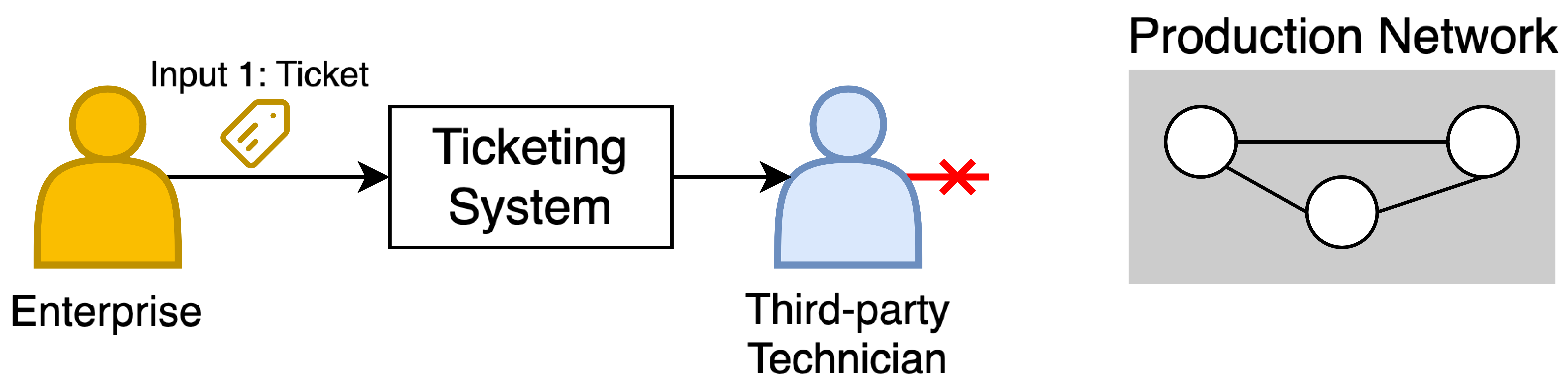
- **Verification of config changes: correctness is hard to specify**
- SecGuru[1] finds in datacenter network, most firewall rules are redundant
- **Our Approach:**
  - **Centralized Reference Monitor with risk and policy input**
  - **Fine-grained risk monitoring and access privilege management**

[1] Karthick Jayaraman, Nikolaj Bjørner, Jitu Padhye, Amar Agrawal, Ashish Bhargava, Paul-Andre C Bissonnette, Shane Foster, Andrew Helwer, Mark Kasten, Ivan Lee, Anup Namdhari, Haseeb Niaz, Aniruddha Parkhi, Hanukumar Pinnamraju, Adrian Power, Neha Milind Raje, and Parag Sharma. 2019. Validating datacenters at scale. In Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM '19). Association for Computing Machinery, New York, NY, USA, 200–213. https://doi.org/10.1145/3341302.3342094

# Outline

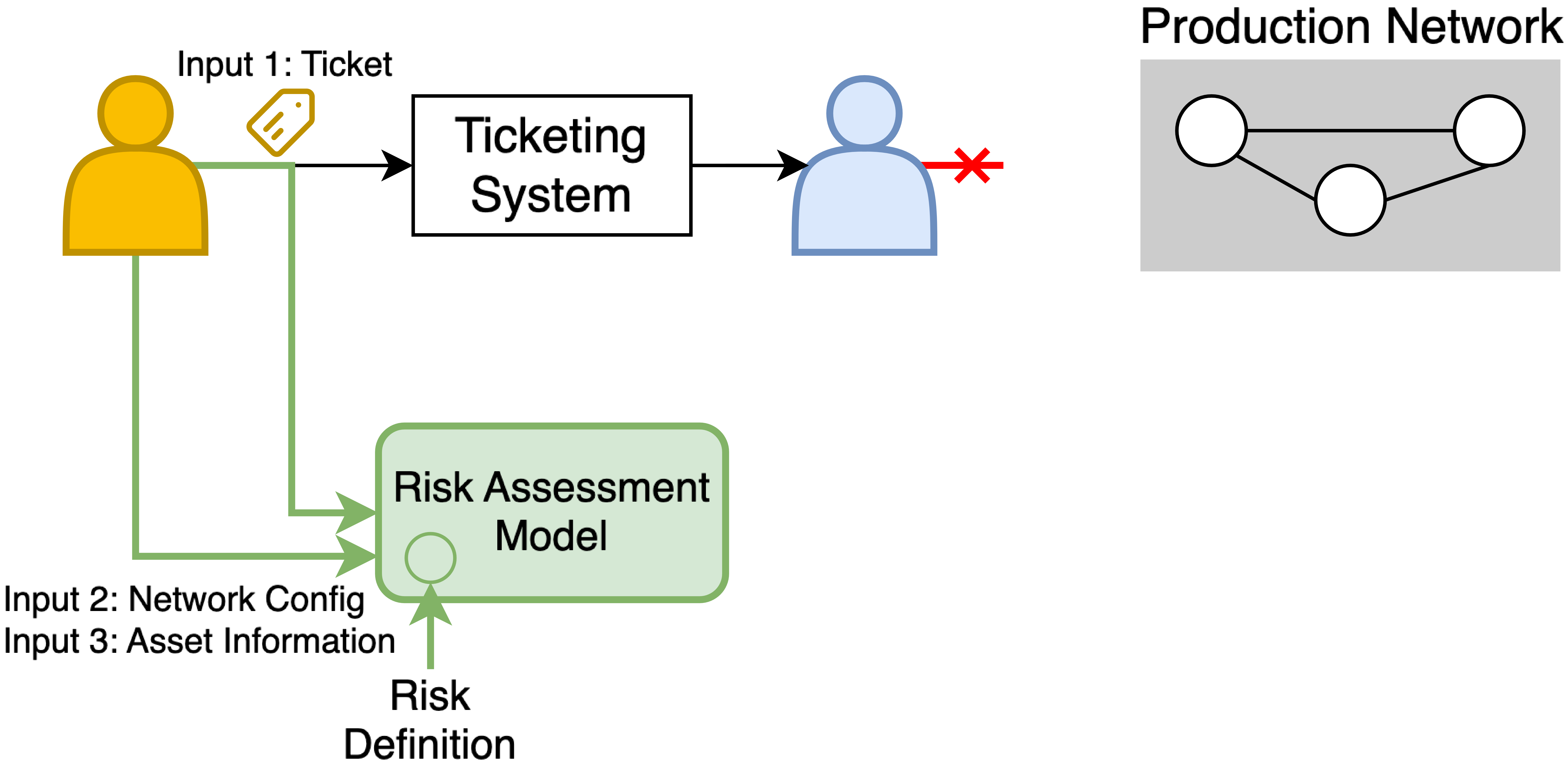
- Inputs
- Workflow
- Evaluation
- Conclusion

# Heimdall’s Risk Management Workflow



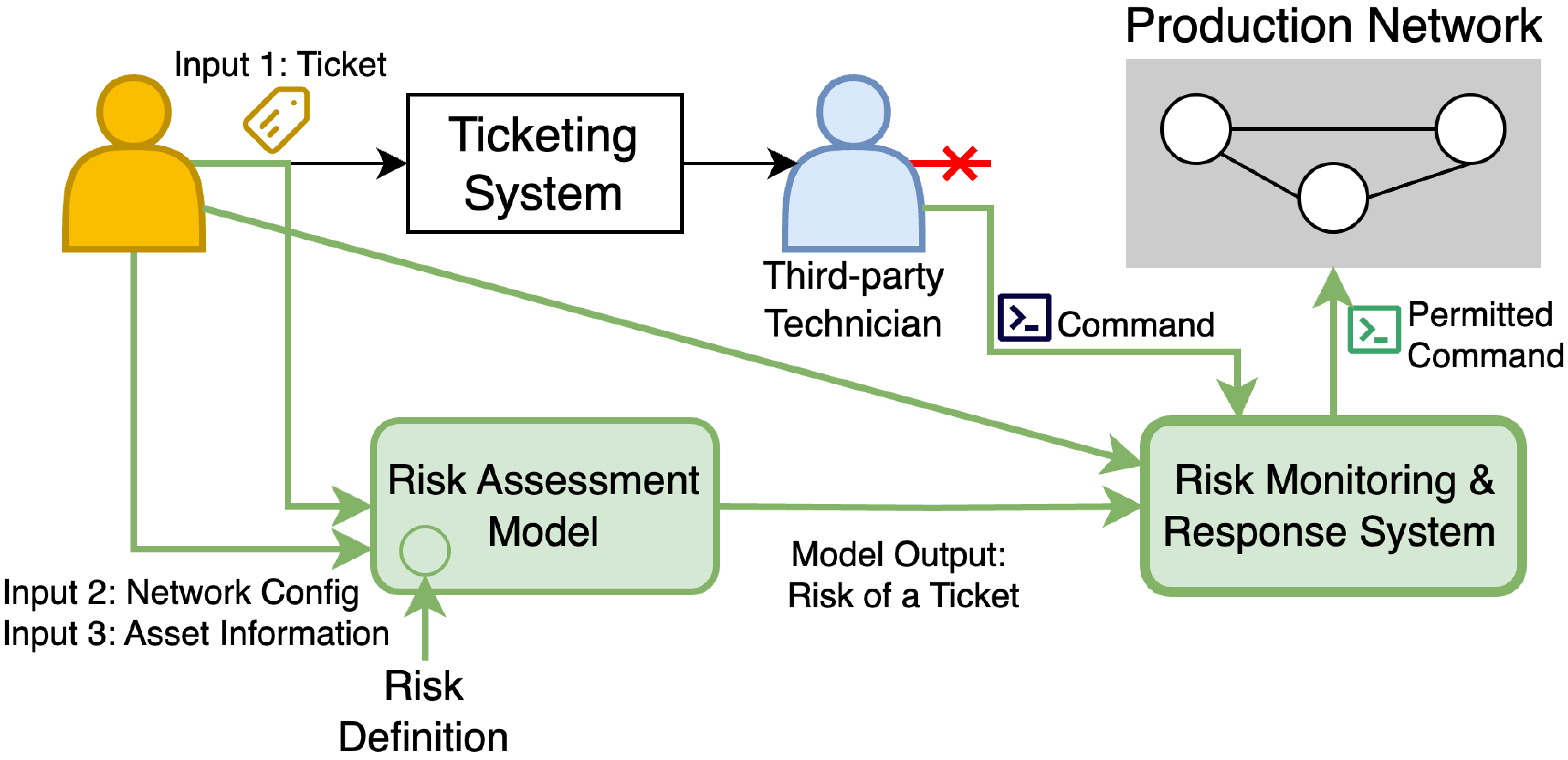
# Heimdall’s Risk Management Workflow

- Feed ticket information into risk assessment model



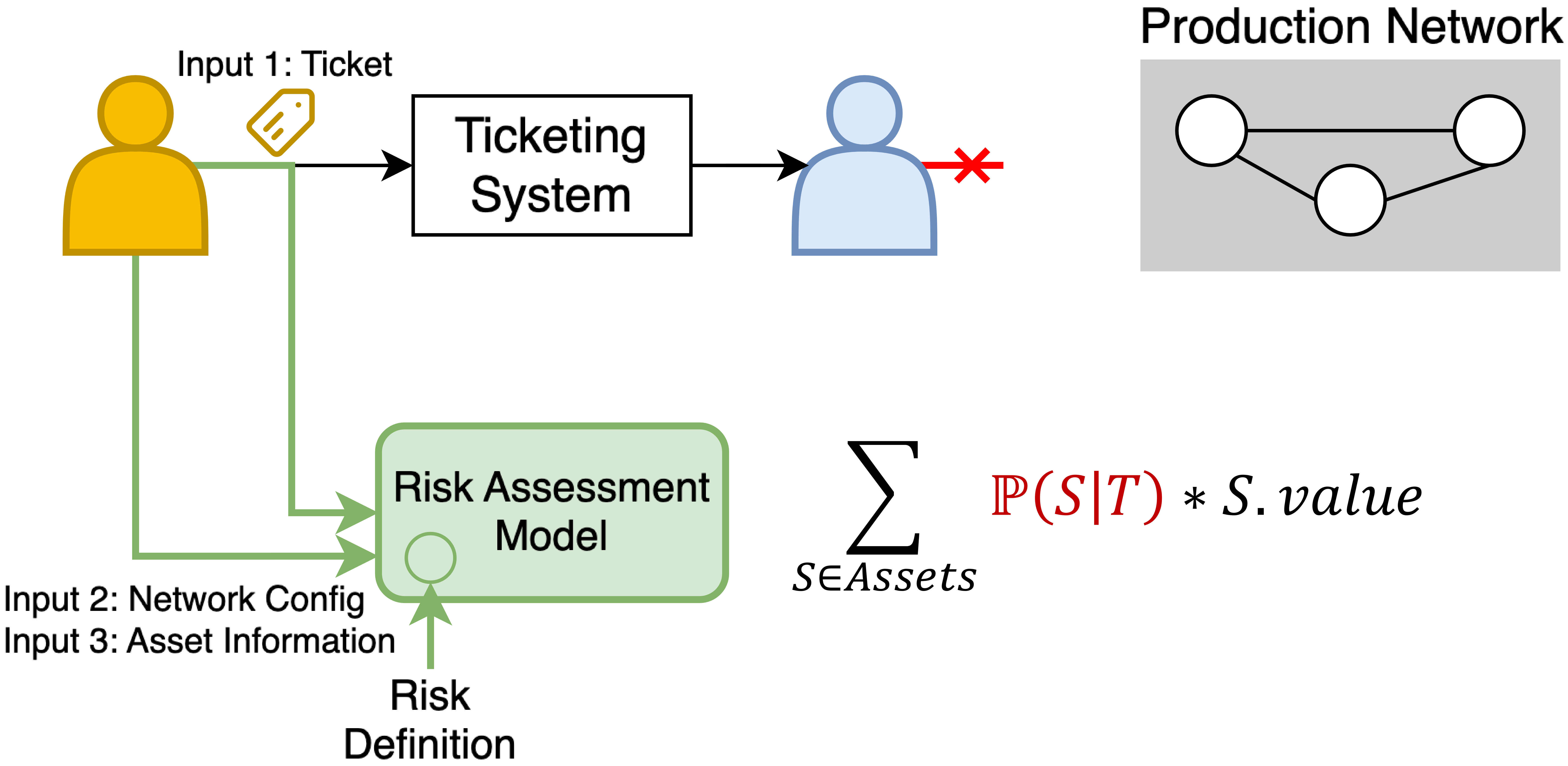
# Heimdall’s Risk Management Workflow

- Keep monitoring risks as technician performing actions



# Heimdall’s Risk Management Workflow

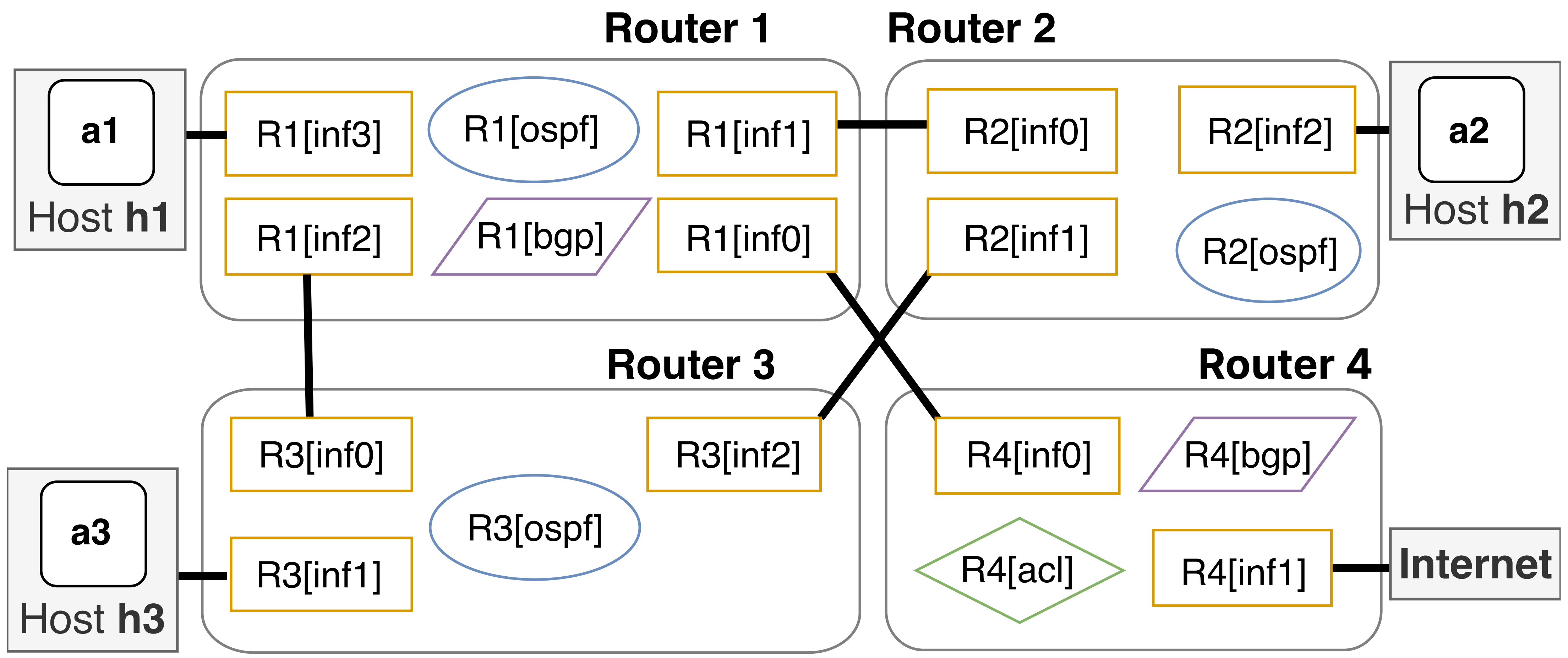
- Feed ticket information into risk assessment model



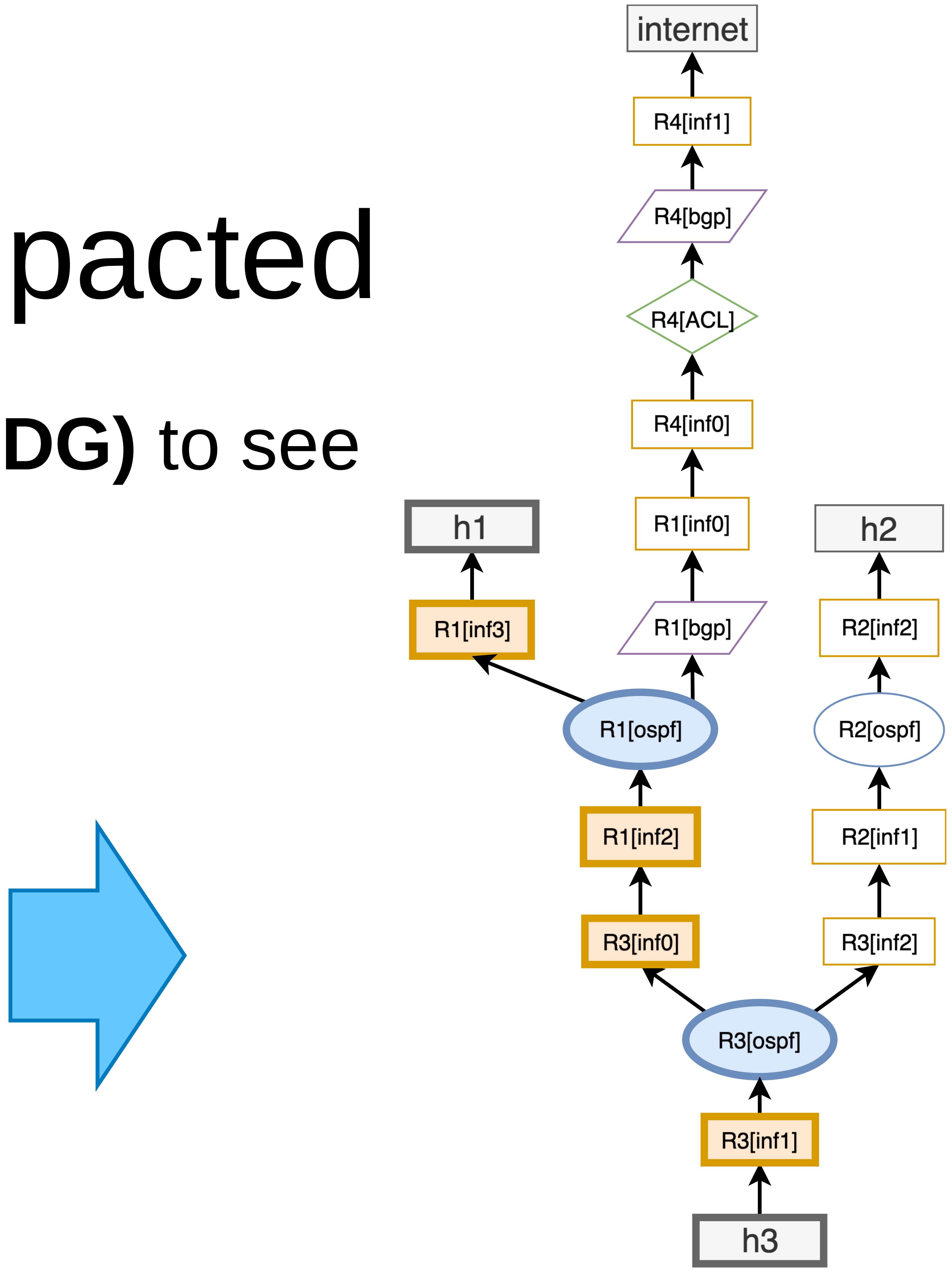
$$\sum_{S\in Assets} \mathbb{P}(S|T) * S.value$$

# Assessing Consequence: Finding How Assets are Impacted

- Constructing **Risk Dependency Graph (RDG)** to see how Assets are impacted through blocks



Example Network



Example RDG

# How to construct an RDG?

- Constructing **Risk Dependency Graph (RDG)** to see how Assets are impacted through blocks
- **Our approach**
  - **Construct Risk Dependency Graph leveraging data-plane information**
  - **A set of 5 rules covering dependency of hosts, interfaces, routing protocols, and ACL items**

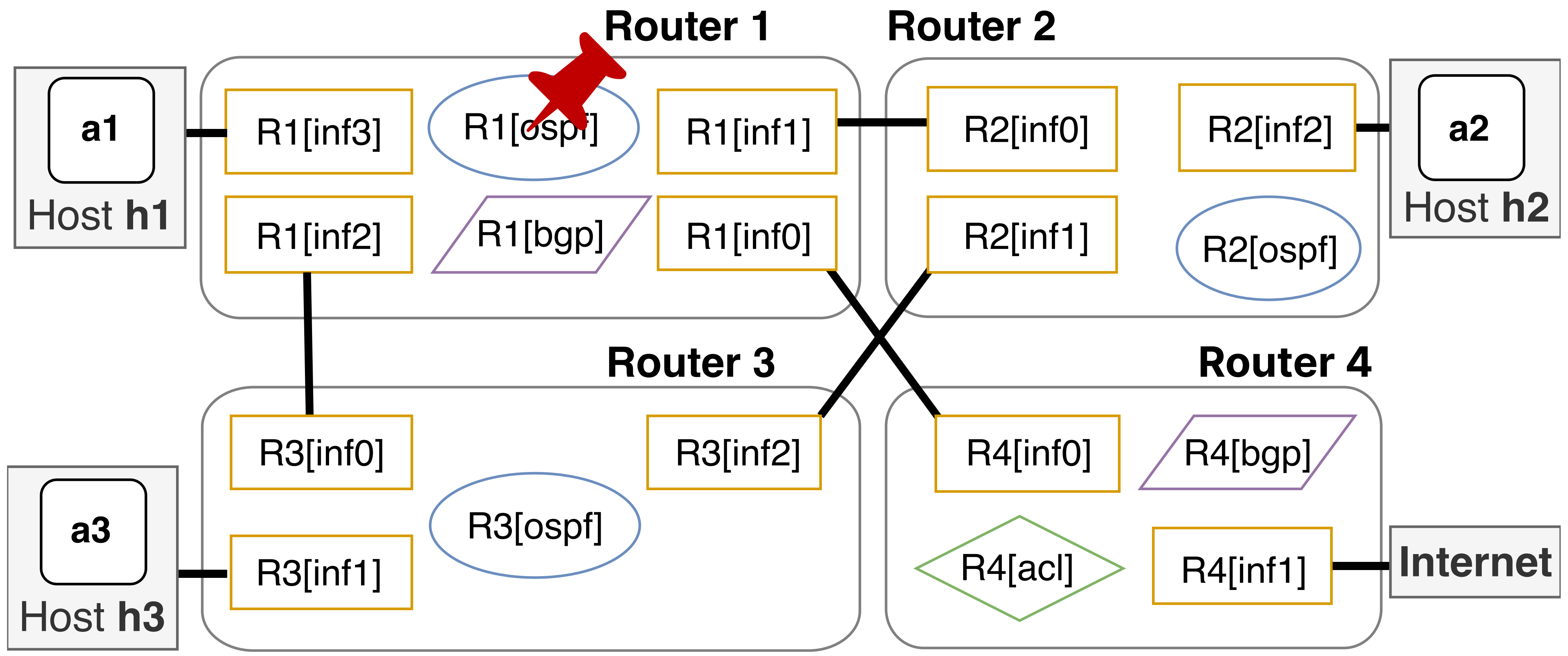
# Assessing Likelihood of Consequence: Finding Probability Assets are Impacted

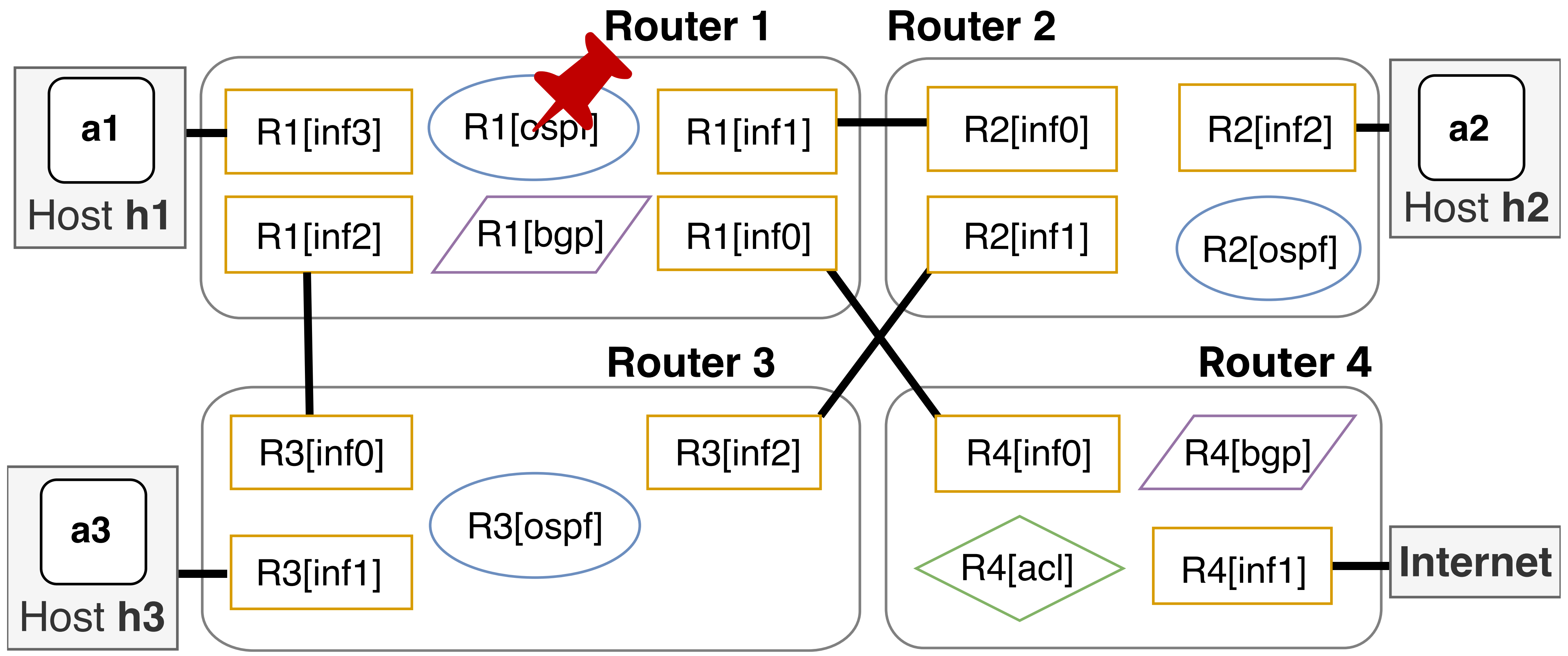
- ***Observation 1***: The order and range in which operators access configuration blocks affect the likelihood of consequence

# Assessing Likelihood of Consequence: Finding Probability Assets are Impacted

- ***Observation 1***: The order and range in which operators access configuration blocks affect the likelihood of consequence

Example: RootCauseBlock = R1[ospf]

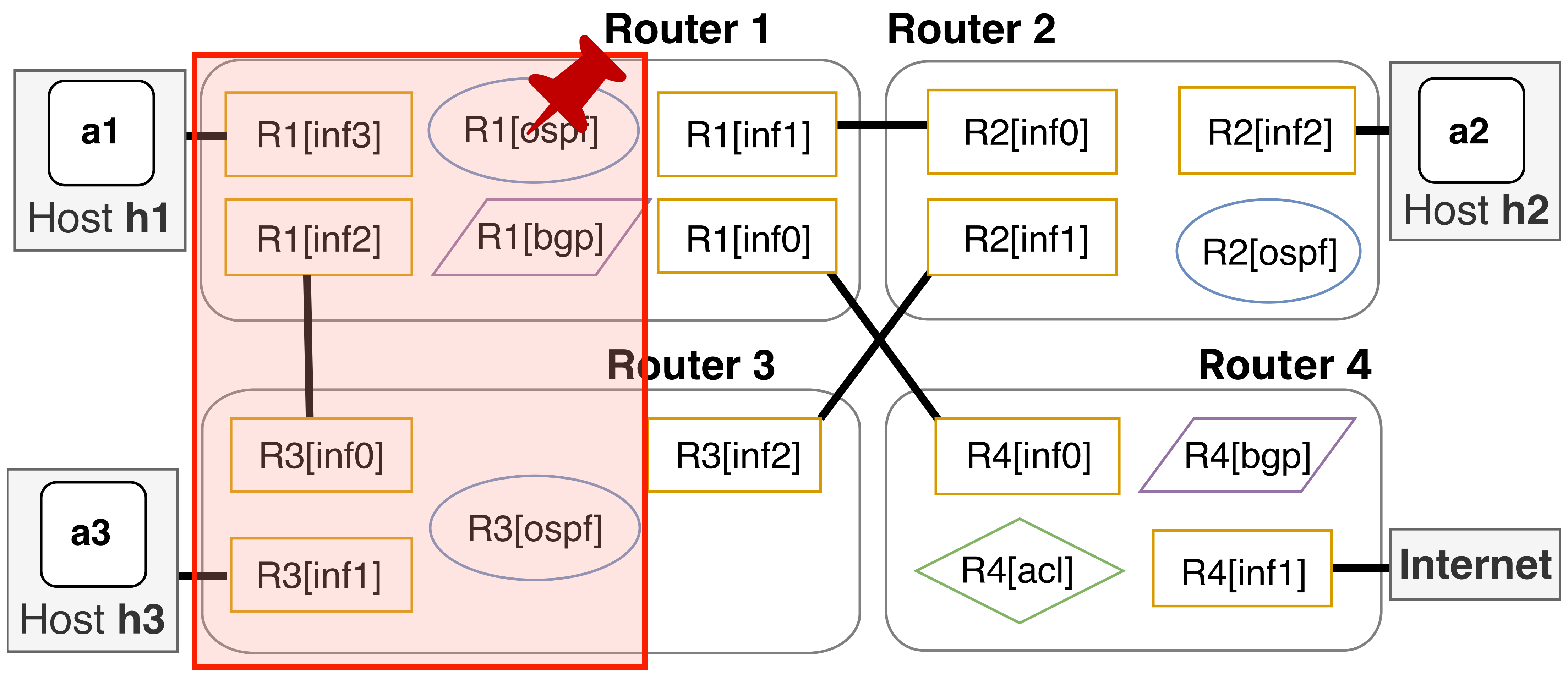




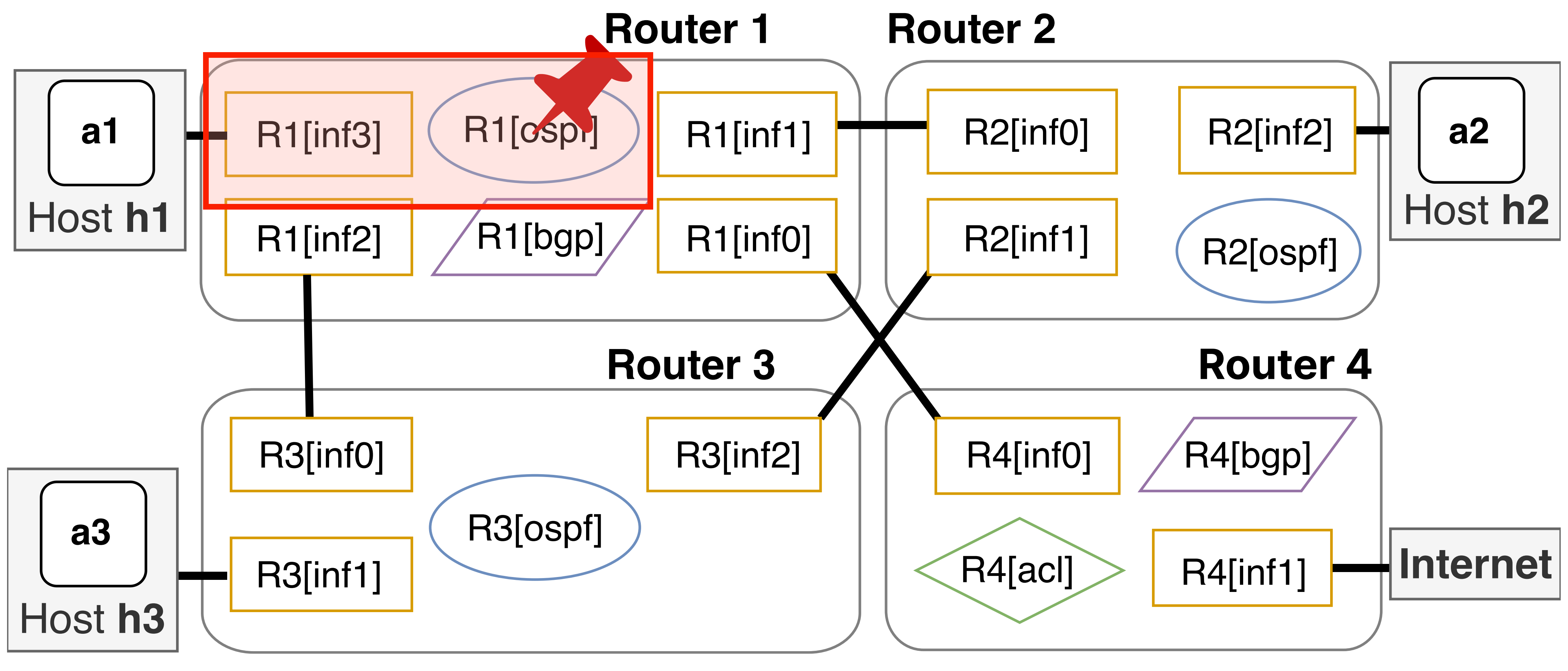
# Assessing Likelihood of Consequence: Finding Probability Assets are Impacted

- ***Observation 1***: The order and range in which operators access configuration blocks affect the likelihood of consequence

Example: RootCauseBlock = R1[ospf]



A Novice Technician: Access many blocks before fixing the problem



An Expert Technician: Quickly identifying the problem and solving

# Finding Probability Assets are Impacted

- ***Key Idea from Observation***: How operators diagnose and resolve a ticket can be abstracted as a ***preference order*** ($\geq_{pref}$) defined on ***configuration blocks*** ($b$) in the RDG subgraph (h$_{src}$, h$_{dst}$).
- An expert technician tends to identify root cause easily and put RootCauseBlock at higher(earlier) position in the total order.

# Finding Probability Assets are Impacted

- **Observation 2**: The root cause can only be (educated) guessed before resolved.
- ***Key Idea from Observation***: Risk Assessing Model takes as input the root cause estimation $\mathbb{P}(c|ticket)$, computed using historical stats

# Step Back to $\mathbb{P}(S|T)$

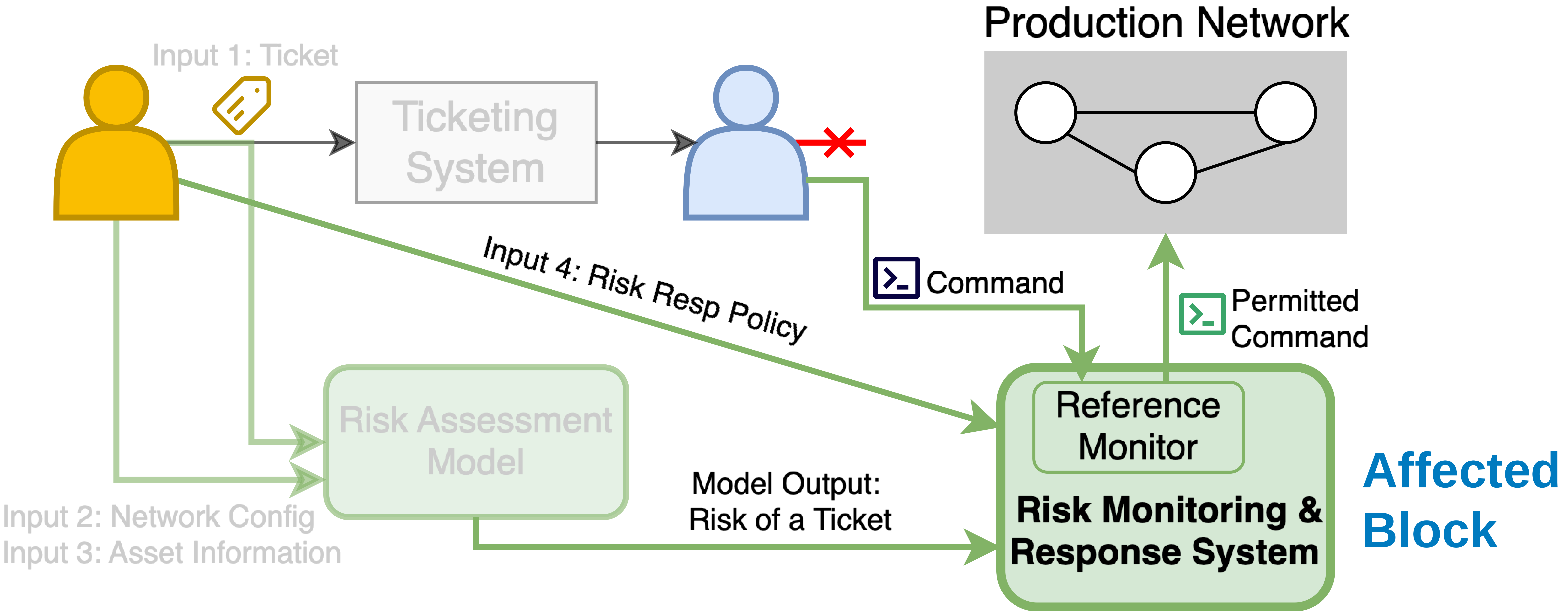
- Recall our risk definition:

$$Risk(T) \coloneqq \sum_{S \in Assets} \mathbb{P}(S|T) * S.value$$

- Risk Dependency Graph, preference order, and root cause estimation, together contribute to $\mathbb{P}(S|T)$, details in our paper.

# Heimdall’s Risk Management Workflow

- Keep monitoring risks as technician performing actions



# Risk Response Policy

- Determine if a command is allowed based on risk response policy
  - We use ticket risk to decide if a command should be allowed to affect the block
  - Risk of granting access to $b$ $\coloneqq$ sum of risks for all $\mathrm{b}' \geq_{pref} b$

# Risk Response Policy

- Determine if a command is allowed based on risk response policy
  - We use ticket risk to decide if a command should be allowed to affect the block
  - Risk of granting access to $b \coloneqq$ sum of risks for all $\mathrm{b}' \geq_{pref} b$

RRP defines the risk threshold and corresponding actions

| **Level** | Low | Medium | High |
|---|---|---|---|
| **Accumulated Risk% of Accessible Blocks** | 30% | 60% | 100% |
| **Action** | Allow | Alert | Stop |

# Risk Response Policy

- Determine if a command is allowed based on risk response policy
  - We use ticket risk to decide if a command should be allowed to affect the block
  - Risk of granting access to $b$ $\coloneqq$ sum of risks for all $b' \geq_{pref} b$

RRP defines the risk threshold and corresponding actions

| **Level** | Low | Medium | High |
|---|---|---|---|
| **Accumulated Risk% of Accessible Blocks** | 30% | 60% | 100% |
| **Action** | Allow | Alert | Stop |

RRP defines the access granting batches following preference order

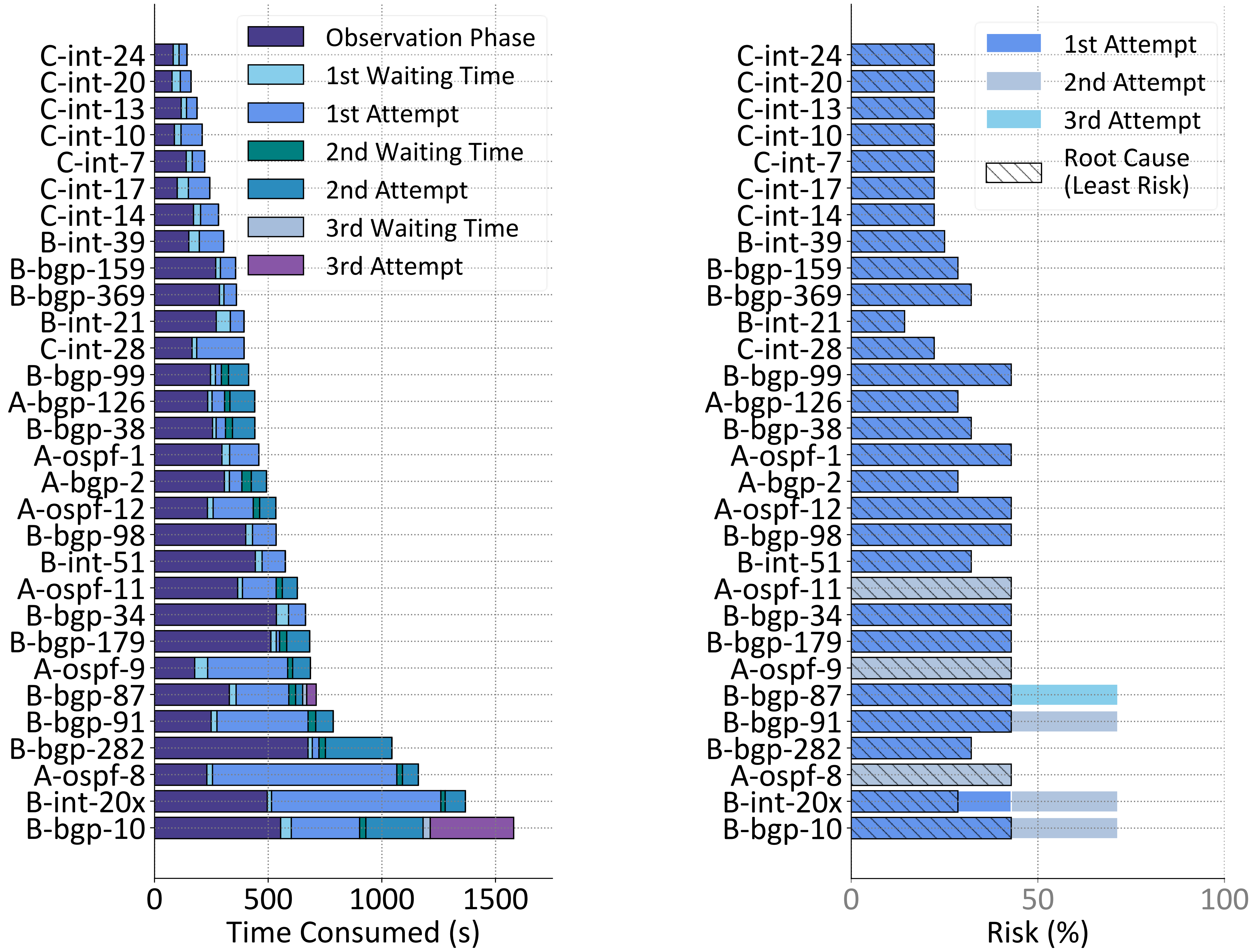
| **Step** | 1st Batch | 2nd Batch | … |
|---|---|---|---|
| **Blocks** | {R1[inf0], R1[inf1]} | {R1[ospf],R3[ospf]} | {…} |

# Outline

- Inputs
- Workflow
- Evaluation
- Conclusion

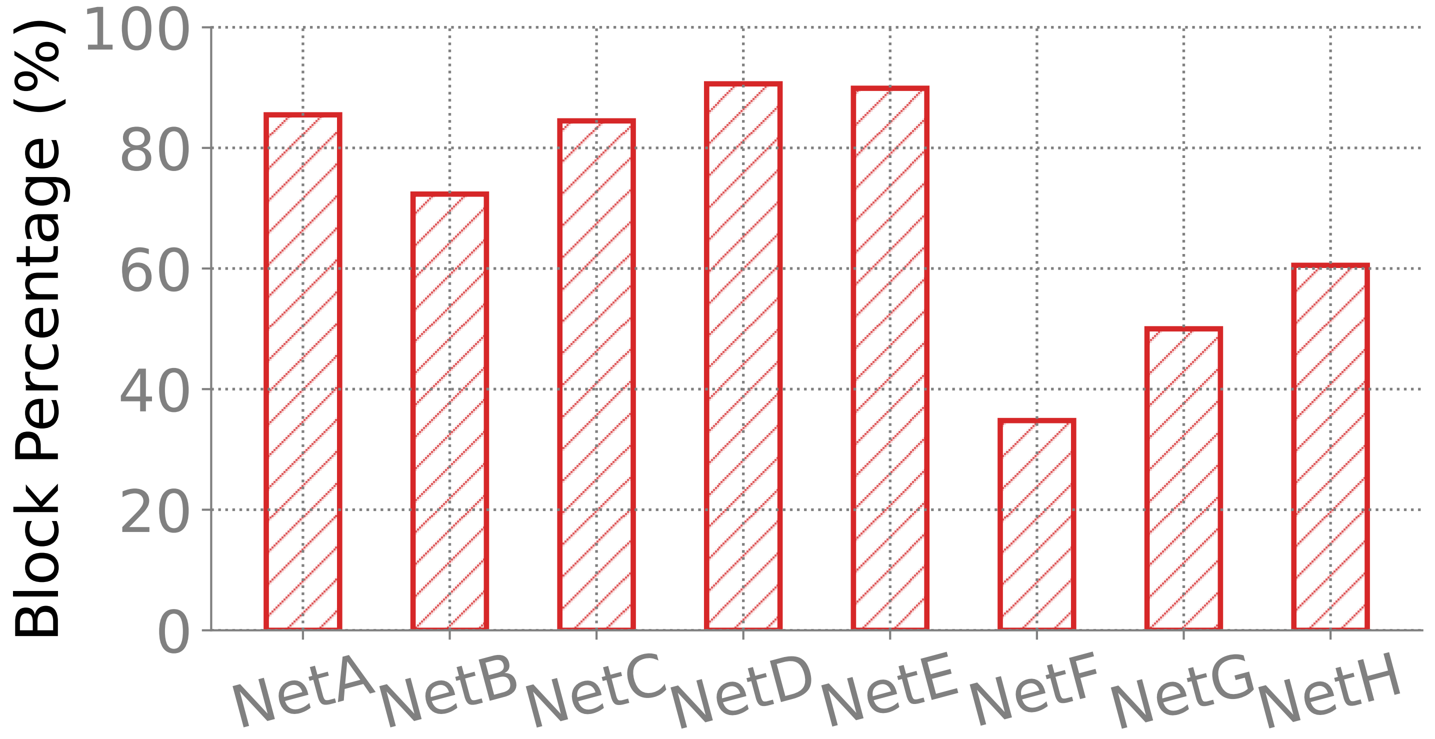
# Expert Validation

- 10 experts, 99 round of experiments with 41hours.
- **Heimdall is efficient for practical usage:** Average completion time is 550s, with 7.4% time overhead in operating time.
- **Heimdall reduces outsourcing risks effectively:** 92% tasks are solved without extra risks than the root cause block only.



# Risk Assessment Effectiveness

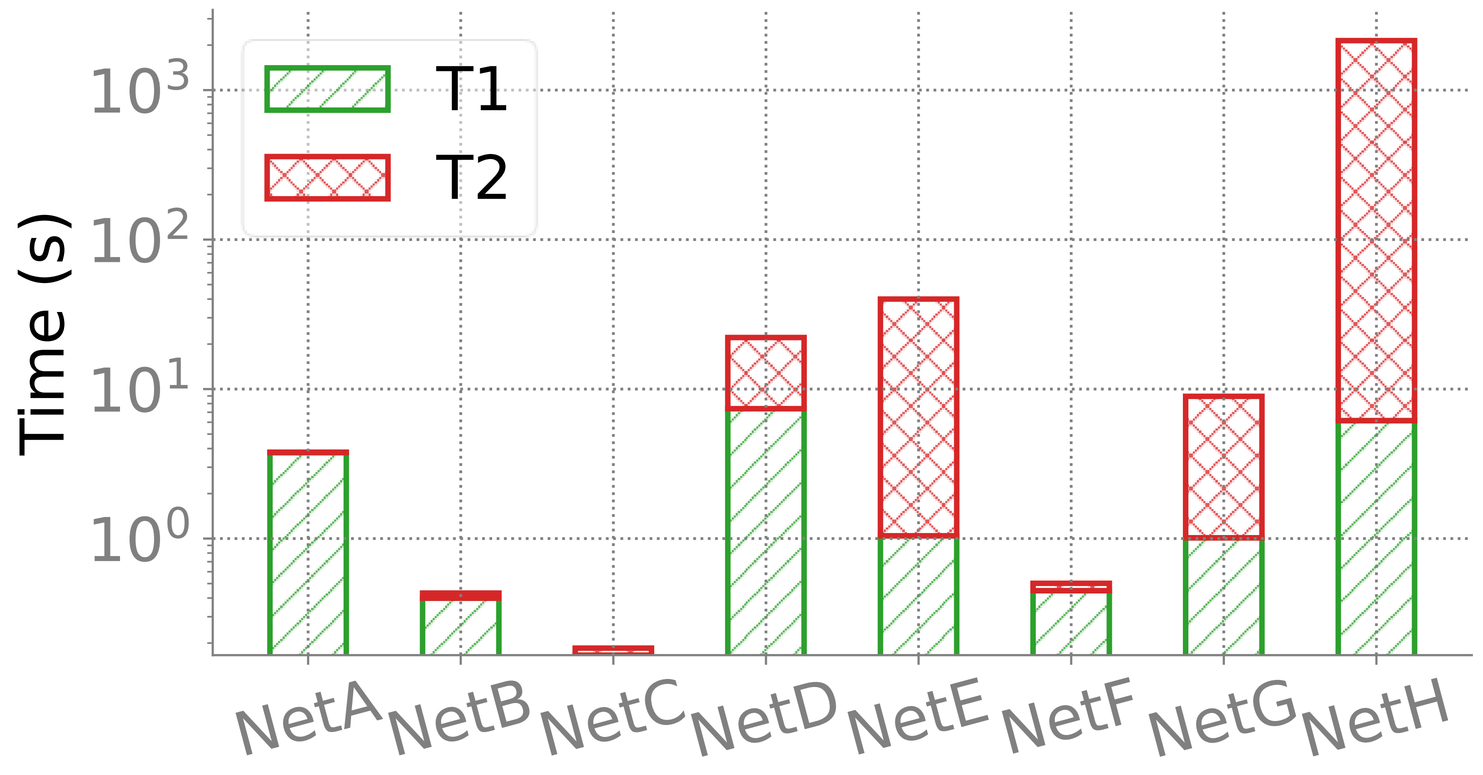
- Strawman approach by linking blocks sharing the same token generates up to >80% false-positive dependency links.
- Comprehensive study of affecting factor of risks, including block type, access order, granularity, etc., is discussed in our paper.



Accuracy comparison of token-matching and RDG model

# System Components Performance

- The Reference Monitor processes commands at negligible expense of 4-5ms, <0.5% of command execution latency.
- 86% of tasks incurs less than 10% overhead on the risk-aware privilege granting workflow.
- All computation time is proactive, before solving a ticket.



Time of computing the RDG for each network

# ***Heimdall***: Towards Risk-Aware Network Management Outsourcing

- It’s possible to manage risks when outsourcing network management
- Challenge: figuring out the inputs
  - What risk is associated with fixing a ticket?
  - Changes to what configuration blocks pose greatest risk?
  - How to automatically enforce risk-based policies?
- Read our paper for details.
- We invite you to use our risk assessment techniques in your work!