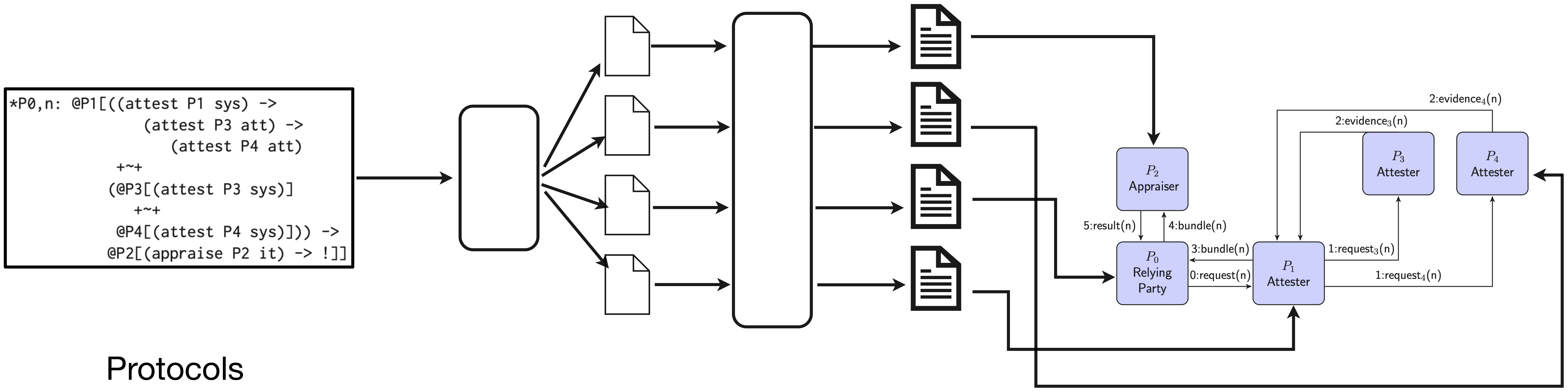
Predictable and Scalable Remote Attestation

Perry Alexander, Adam Petz, Will Thomas, Logan Schmalz, Sarah Johnson
Institute for Information Sciences
The University of Kansas
{palexand,ampetz,30willthomas,loganschmalz,sarahjohnson}@ku.edu

Predictable and Scalable Remote Attestation

- Evidence and Time - A semantics of evidence over time that allows predictions about the effectiveness of attestation evidence in appraising systems
- Flexible Mechanisms at Scale - A semantics for appraisal architectures and its realization as a collection of reusable attestation components and tools for static analysis.
- Empirical Case Studies - Large scale empirical studies of defining, implementing, and running attestation architectures with applications in supply chain and zero trust.

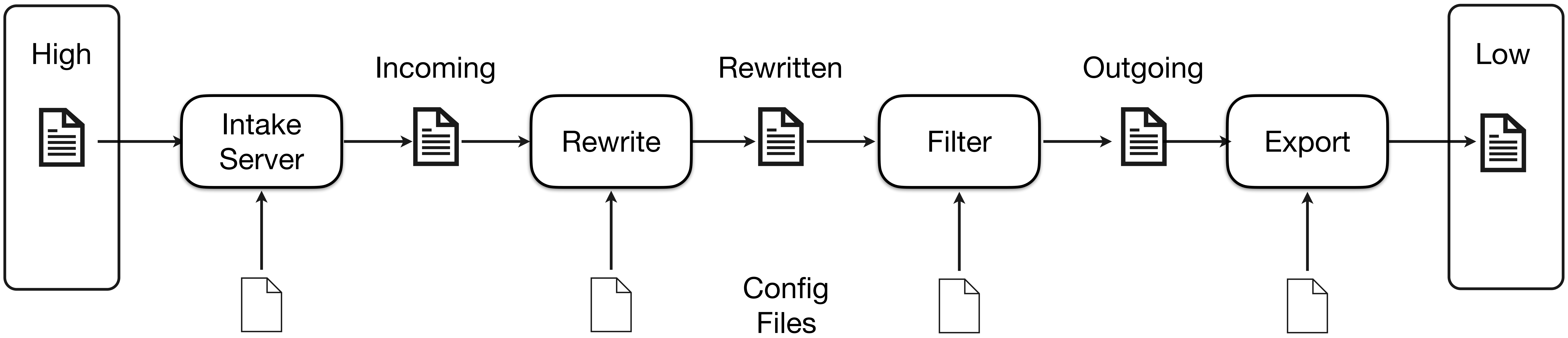
MAESTRO Attestation Infrastructure

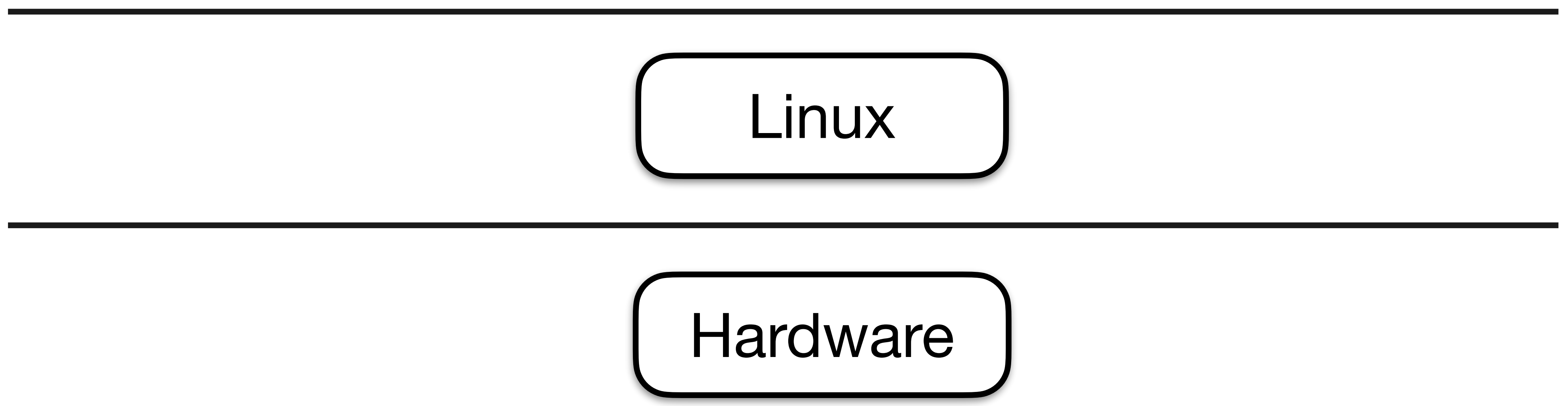


Protocols

- Long running attestations
  - to our knowledge no one has studied long-running experiments on complex attestations
  - evaluating various flexible mechanisms
- Modeling attacks
  - sneaking by the attestation/appraisal system
  - directly attacking the attestation/appraisal system
- Attestation Test Bed
  - controlled evaluation environment
  - mixed architecture - ARM, Intel, IoT, Xen, KVM

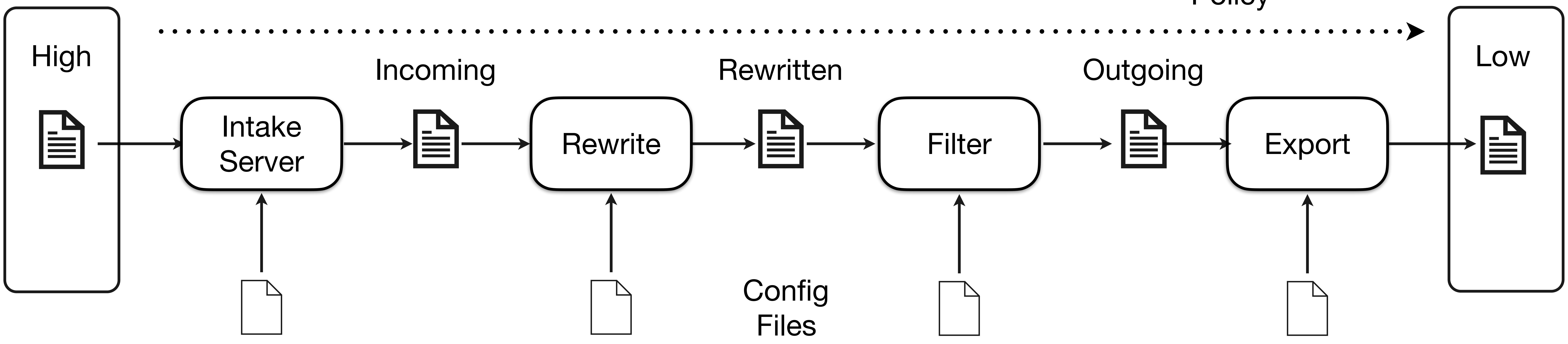
Cross Domain System



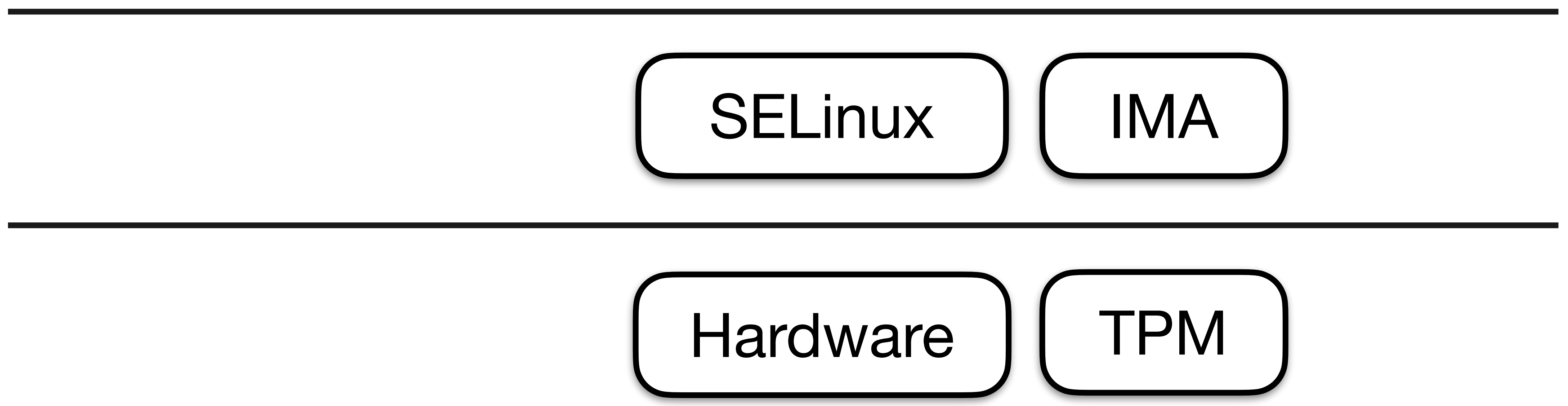


- Moving messages between security domains
  - intake receives a message from the high-side client and writes to incoming buffer
  - rewriter reads from the incoming buffer, applies rewrite rules, and writes to rewritten buffer
  - filter reads from the rewritten buffer, applies address filtering rules, and writes to outgoing buffer
  - export reads from outgoing buffer and outputs to low-side client
- Configuration
  - processes all have configuration files read at initiation
  - SELinux policy is used to enforce flow through the system
- Messages reaching the low-side client must be:
  - rewritten by a properly configured rewriter
  - filtered by a properly configured address filter
  - received from the high-side client
  - in the right order

Cross Domain System Protection



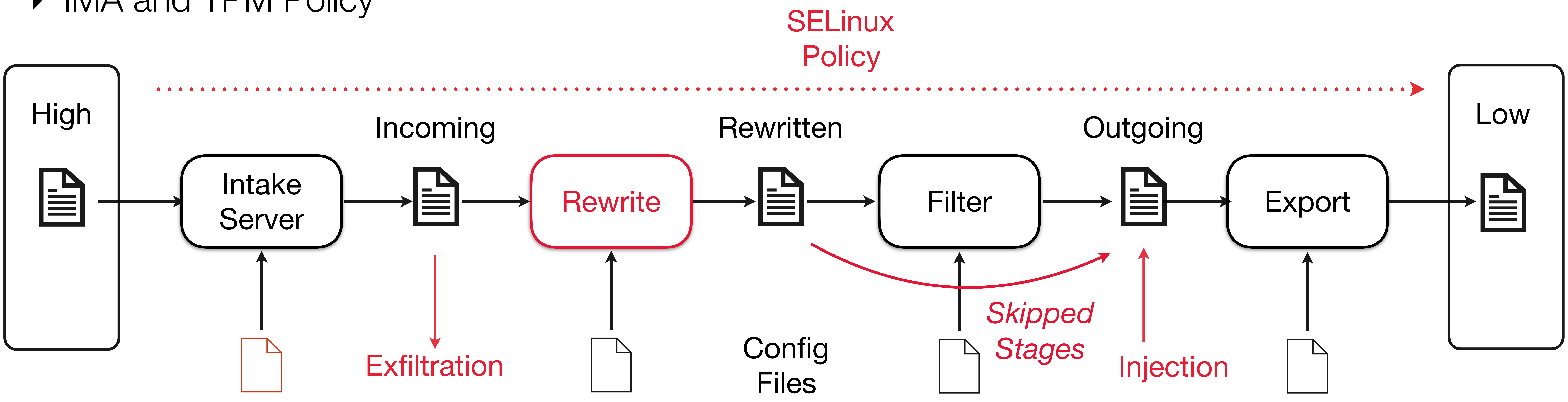
- SELinux & SELinux Policy
  - mandatory access control
  - enforces flow through pipeline
- Integrity Measurement Architecture (IMA)
  - ensures correct binaries start
  - records boot order by extending TPM PCR
- TPM
  - root-of-trust for storage and reporting
  - memorializes boot, seals keys to state



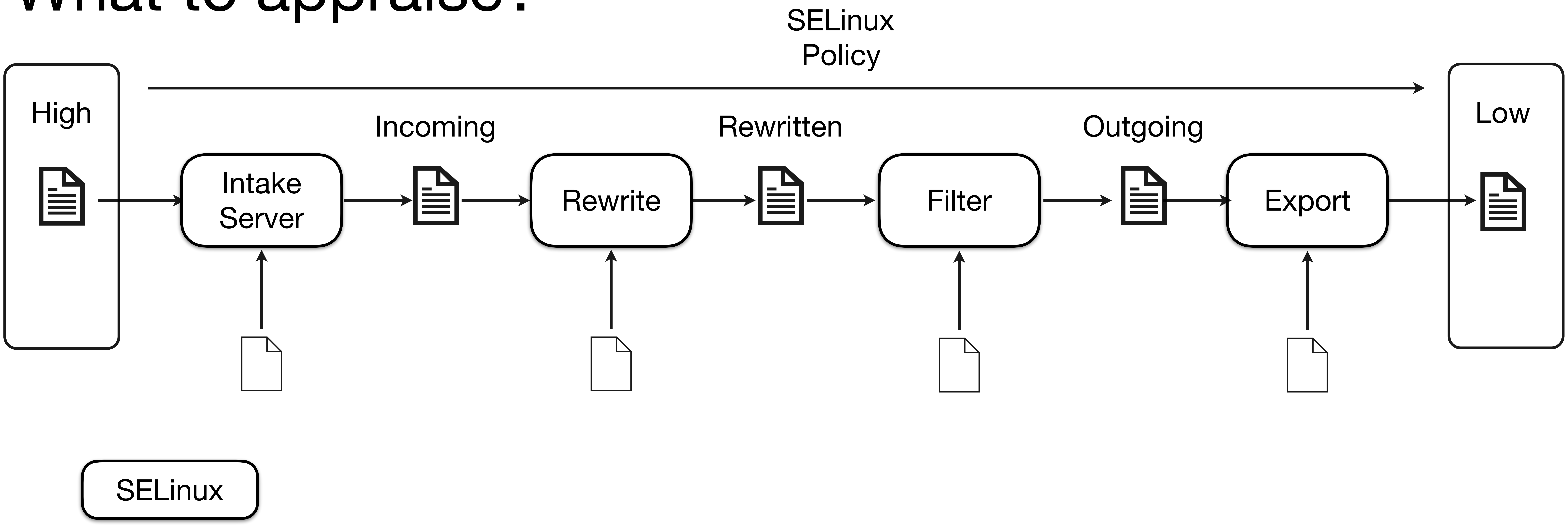
- Enforces basic CDS requirements
  - boot into a good state
  - general access control
  - message flow

What might an adversary target?

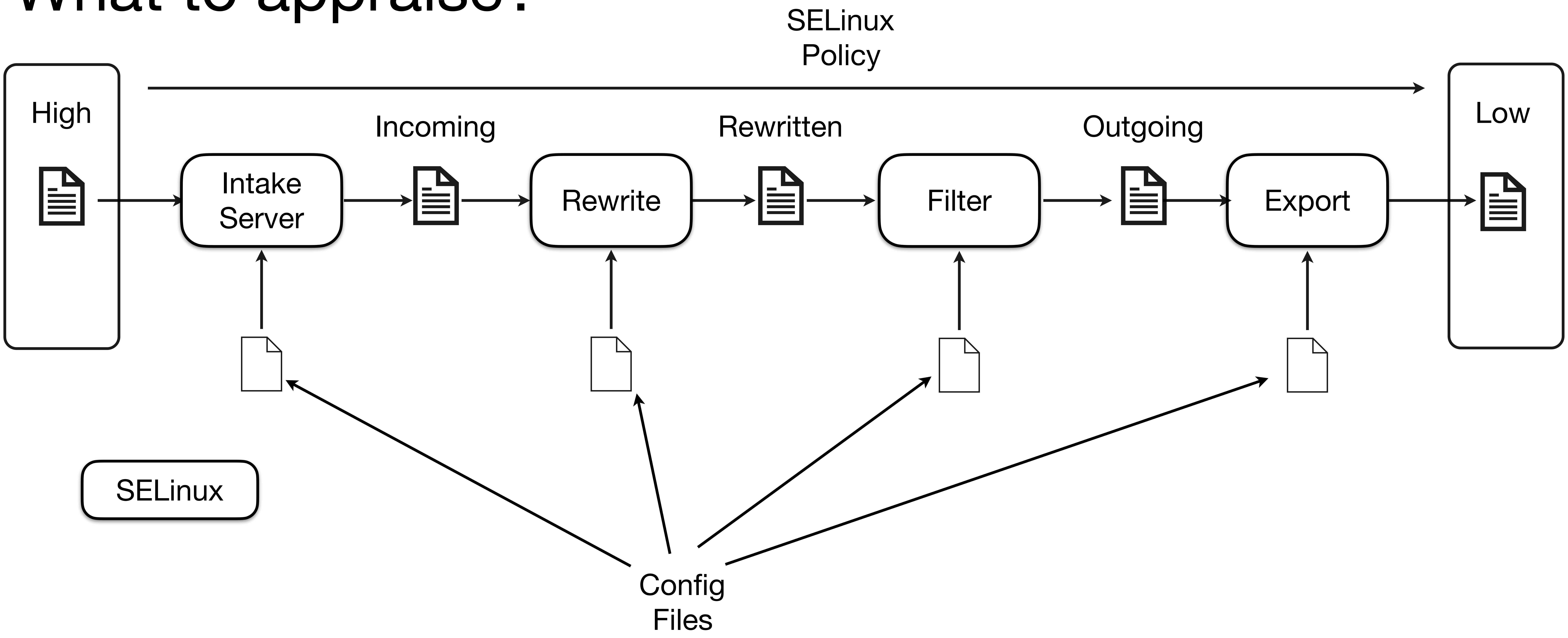
- Configuration files for pipeline binaries
- Pipeline binaries themselves
- Communication paths and buffers
- SELinux Policy
- IMA and TPM Policy



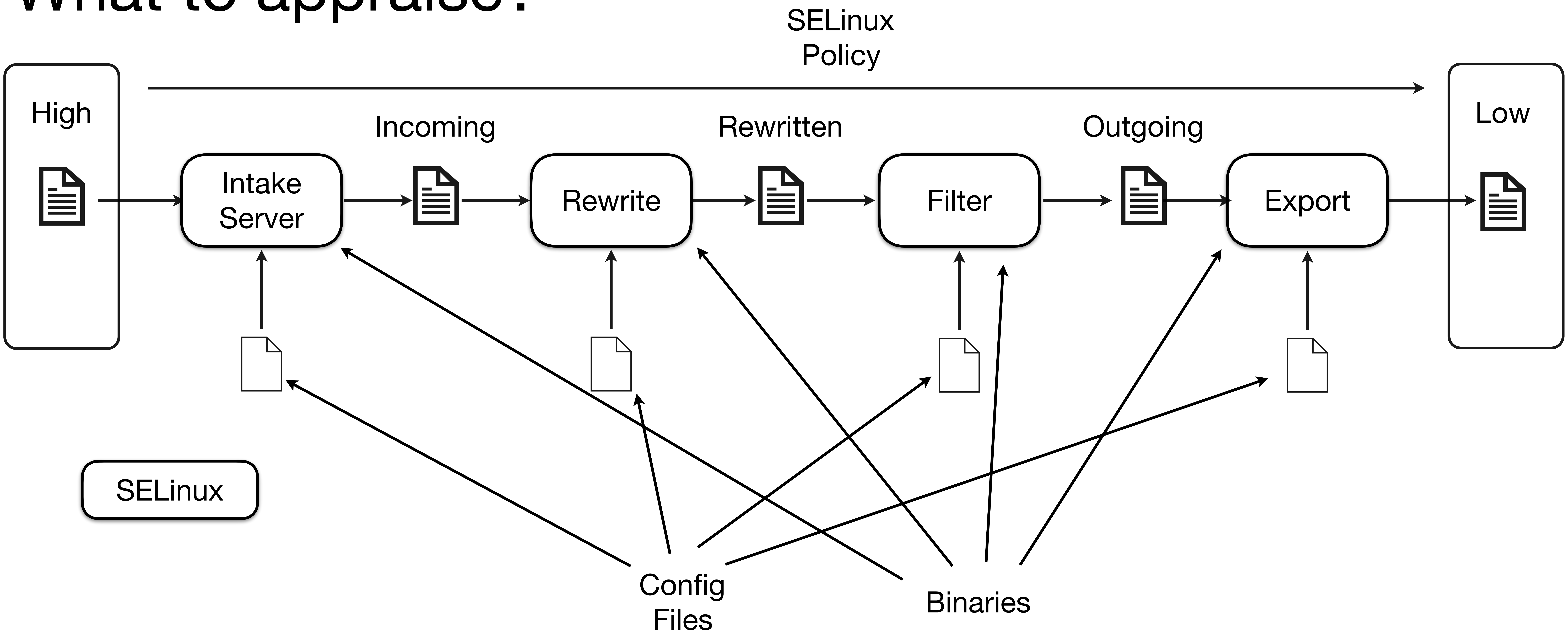
What to appraise?



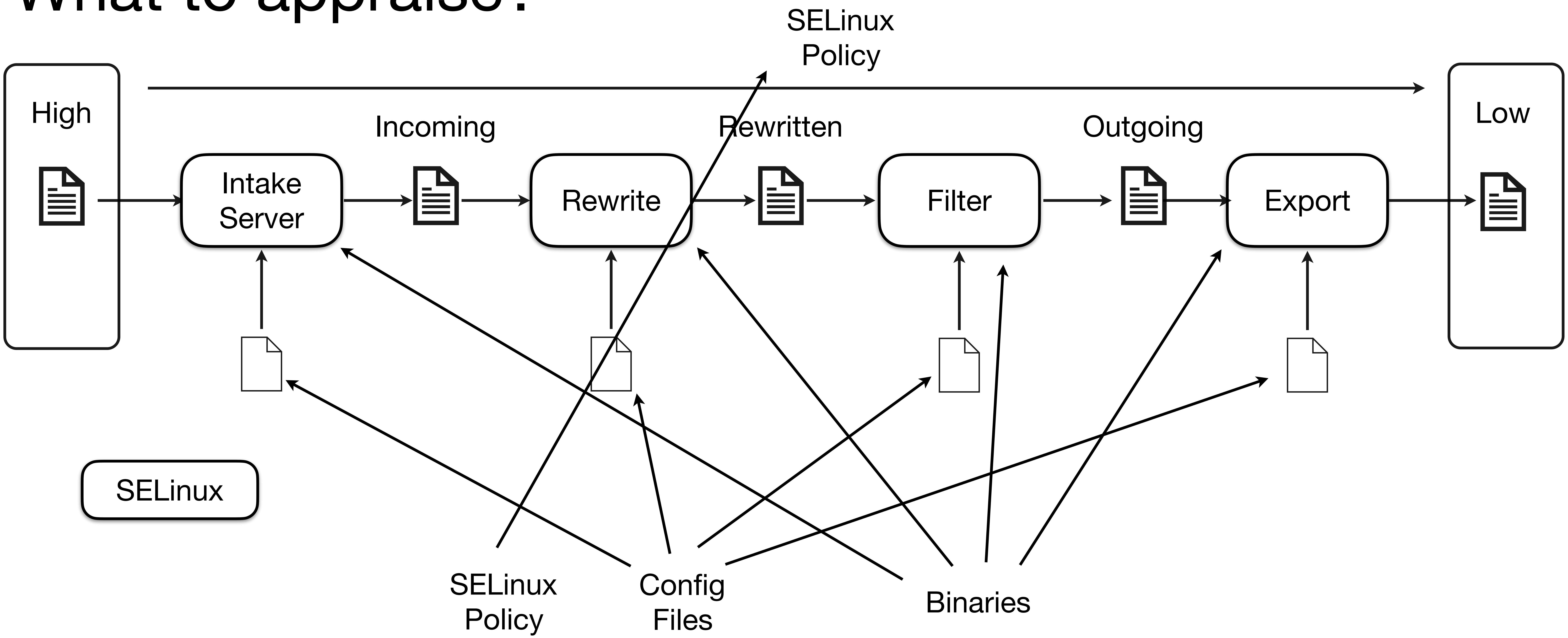
What to appraise?



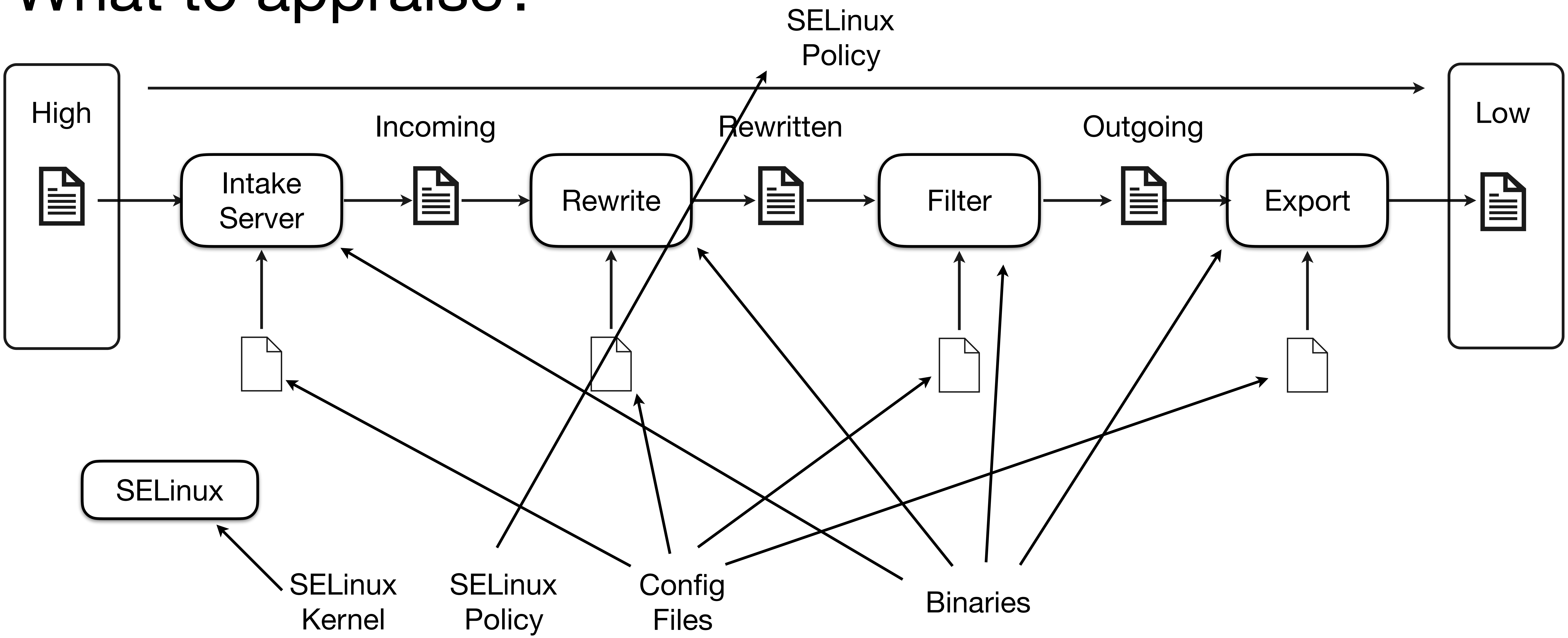
What to appraise?



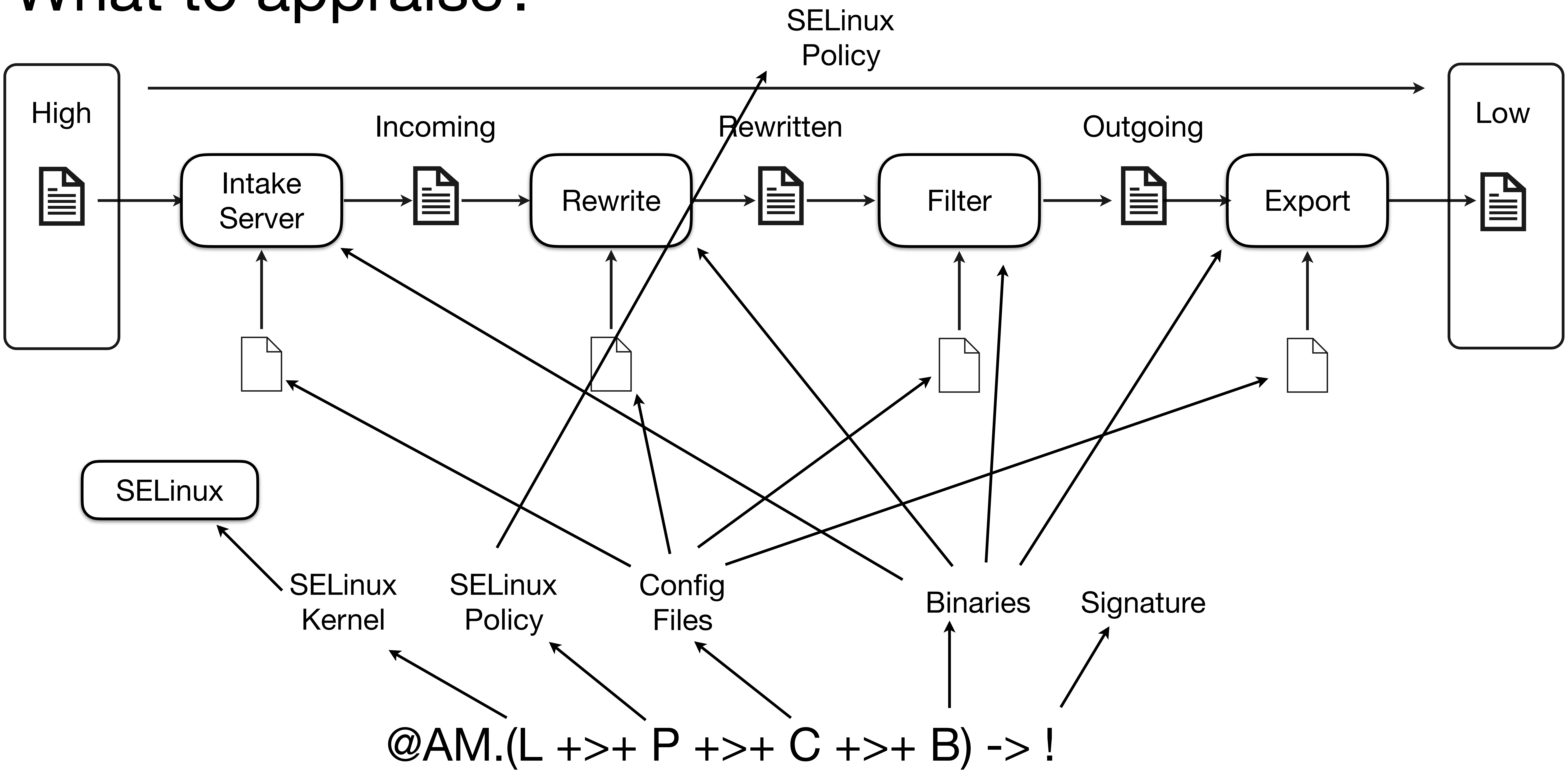
What to appraise?



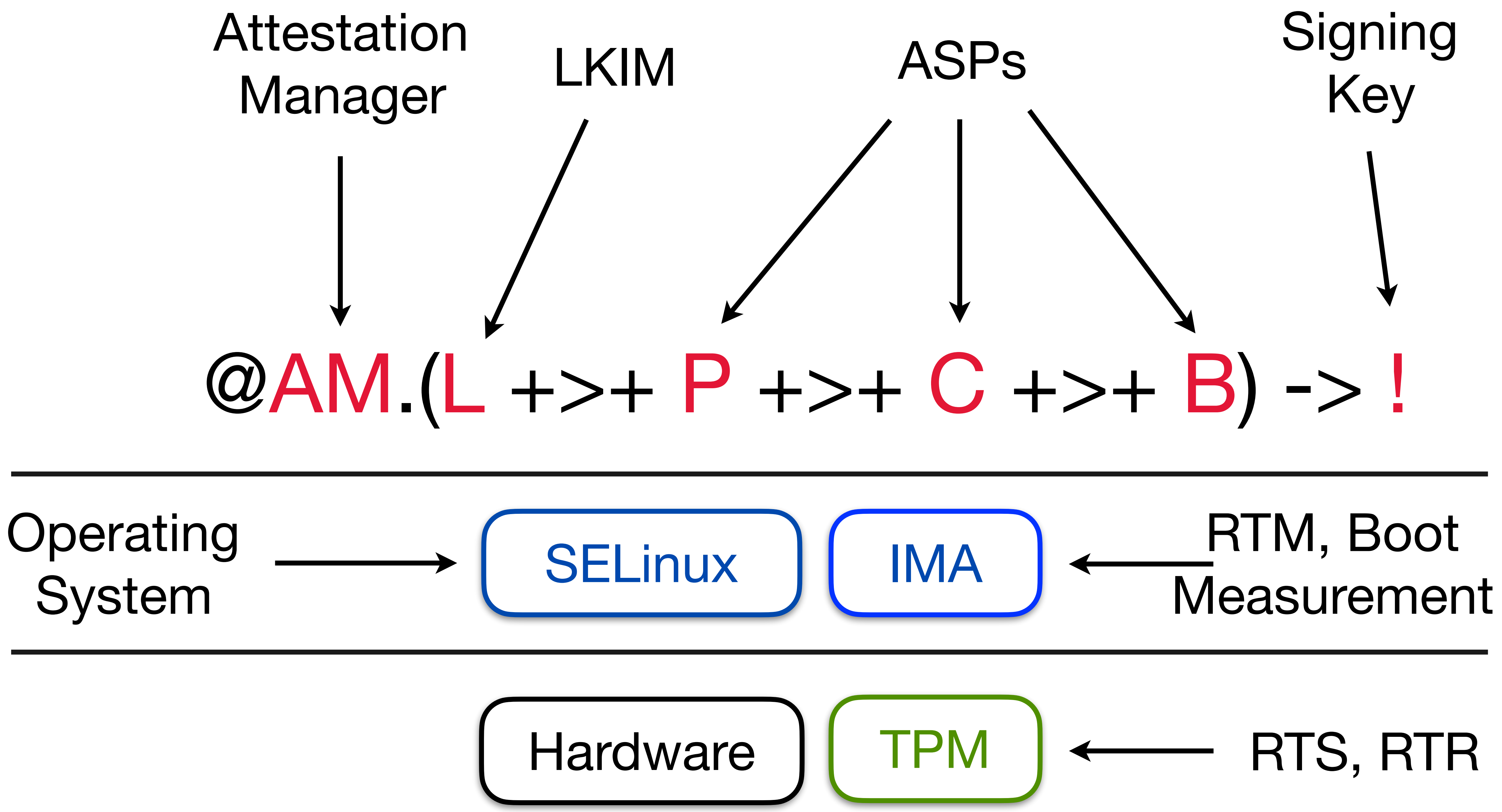
What to appraise?



What to appraise?



What dependencies might an adversary target?

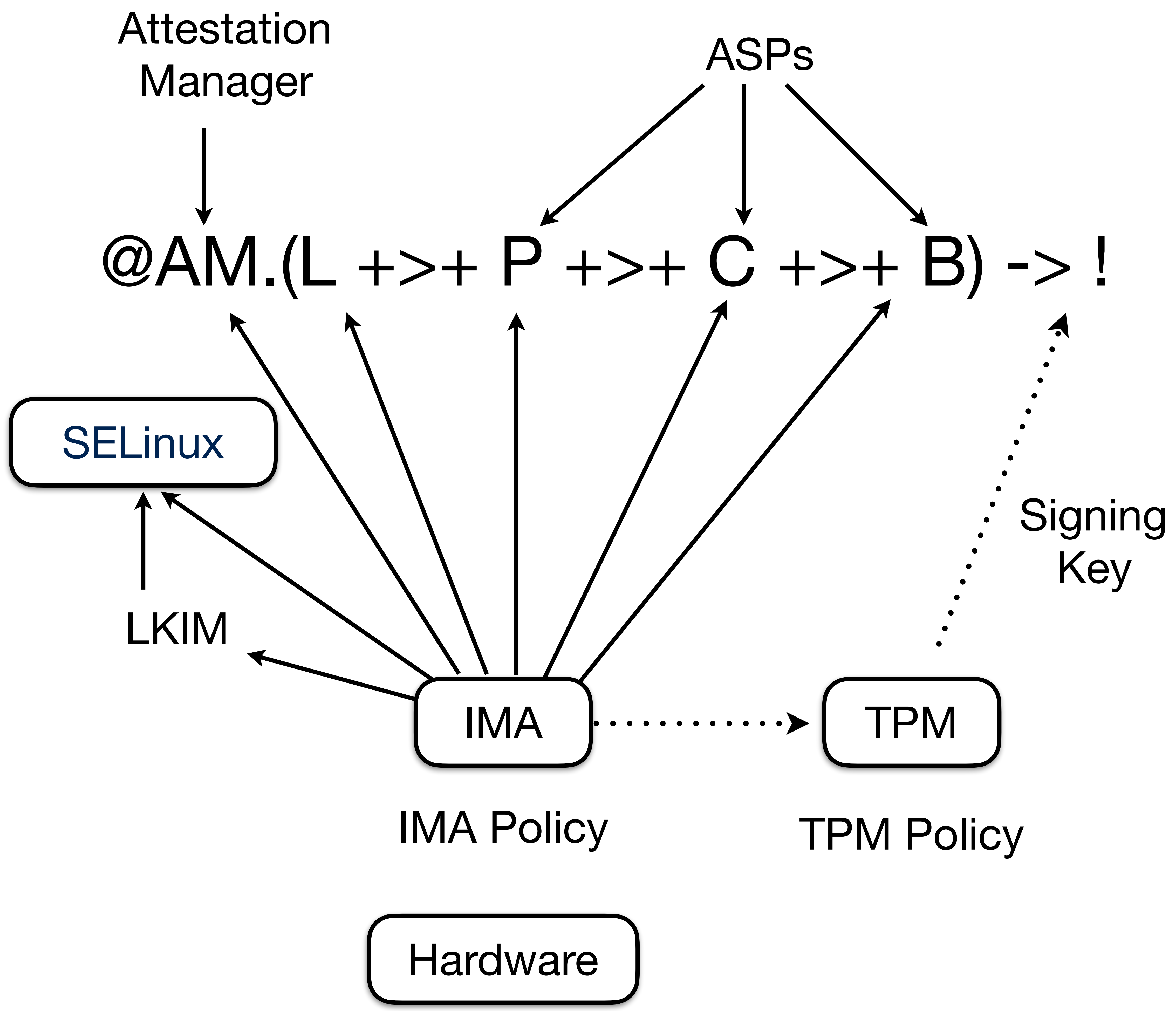


- Deeper subsystems are harder to attack
  - TPM
  - SELinux + IMA
  - Attestation infrastructure
  - Cross Domain System

- Attestation Manager
  - runs attestation protocol
  - signs evidence
  - formally verified
- Attestation Service Providers (ASPs)
  - perform measurements
  - perform appraisals
  - invokes LKIM
- Operating System
  - provides services
  - enforces SELinux policy
- Trusted Platform Module (TPM)
  - root-of-trust for storage
  - root-of-trust for reporting
  - enforces TPM policy
- Hardware
  - Trusted *a priori*

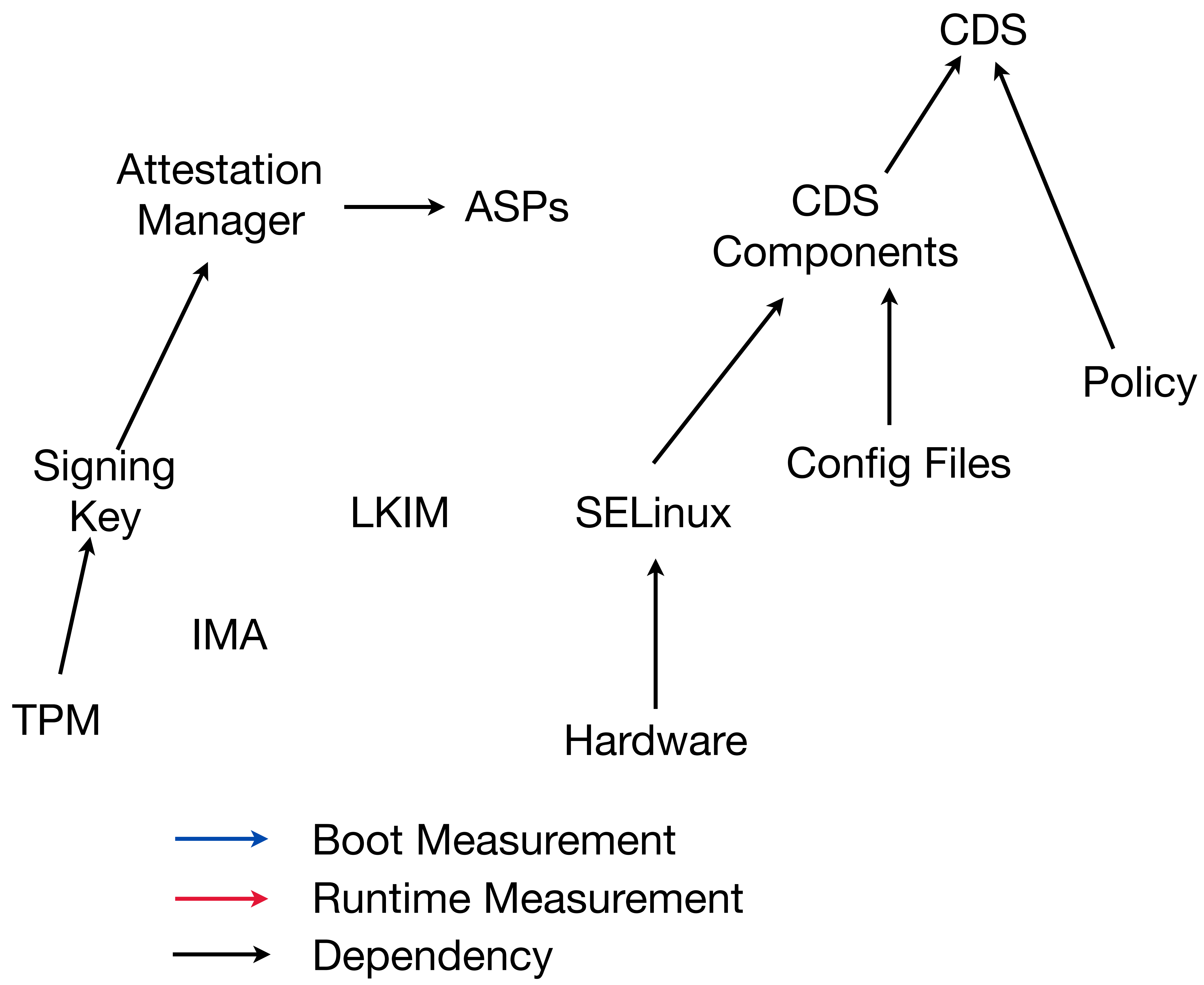
What to appraise and protect?

- Integrity Measurement Architecture (IMA)
  - measures AM and ASPs
  - measures LKIM
  - measures SELinux
  - writes to TPM PCR
- Linux Kernel Integrity Measurer (LKIM)
  - measures SELinux runtime state
  - ASP triggers and gathers measurement
- TPM
  - generates AM's signing key
  - binds signing key to an encrypted credential
  - seals signing key to IMA policy and IMA enable
  - key never leaves the TPM



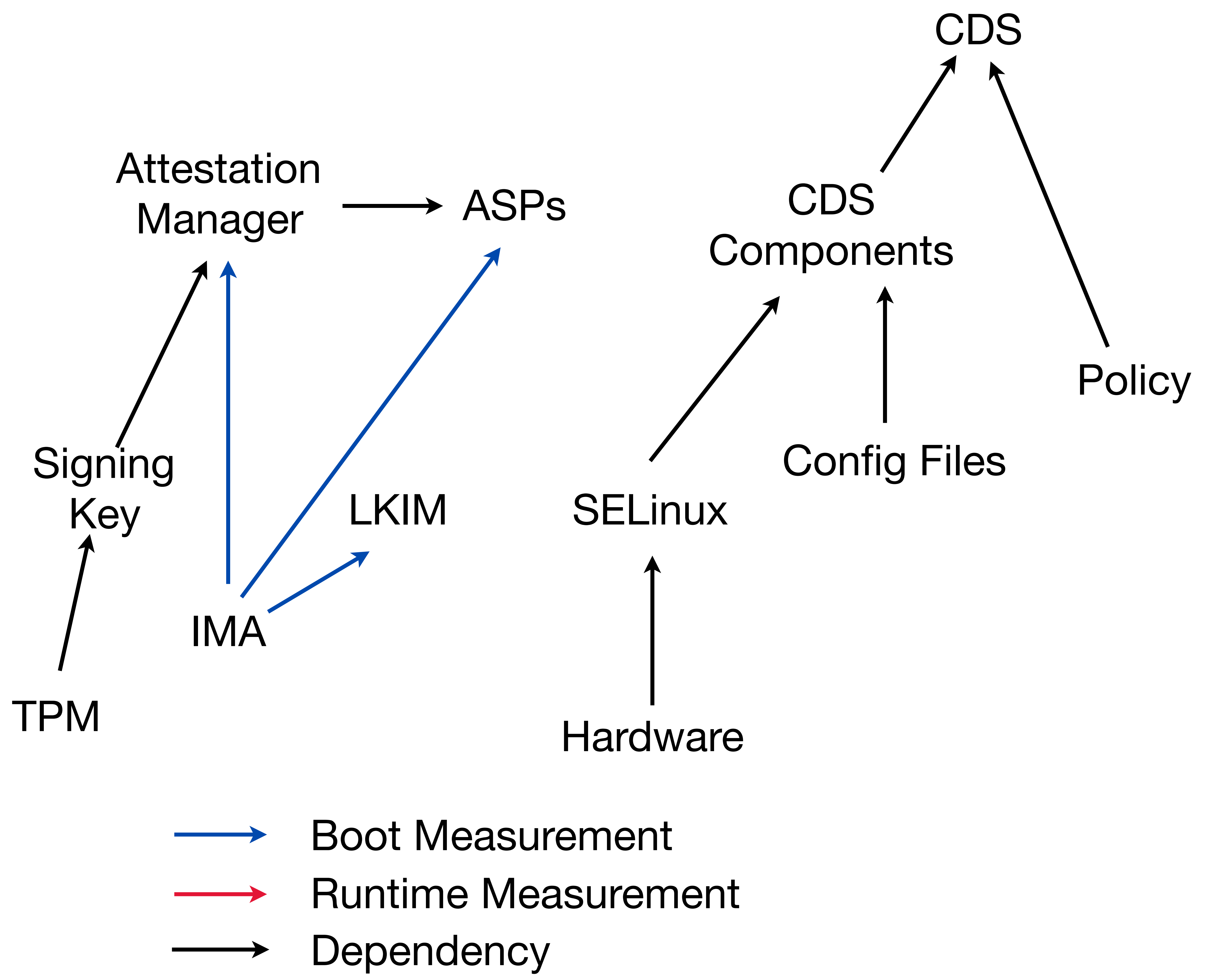
Defining A Protocol

- Establish roots-of-trust
  - measurement - IMA
  - storage - TPM
  - reporting - TPM
- Dependencies are measured first
  - SELinux before AM, CDS
  - AM and ASPs before CDS
  - Configurations before CDS components
- Deeper systems are harder to attack
  - Hardware, TPM, IMA
  - SELinux, LKIM
  - attestation subsystem
  - application software
- Measurement Frequency



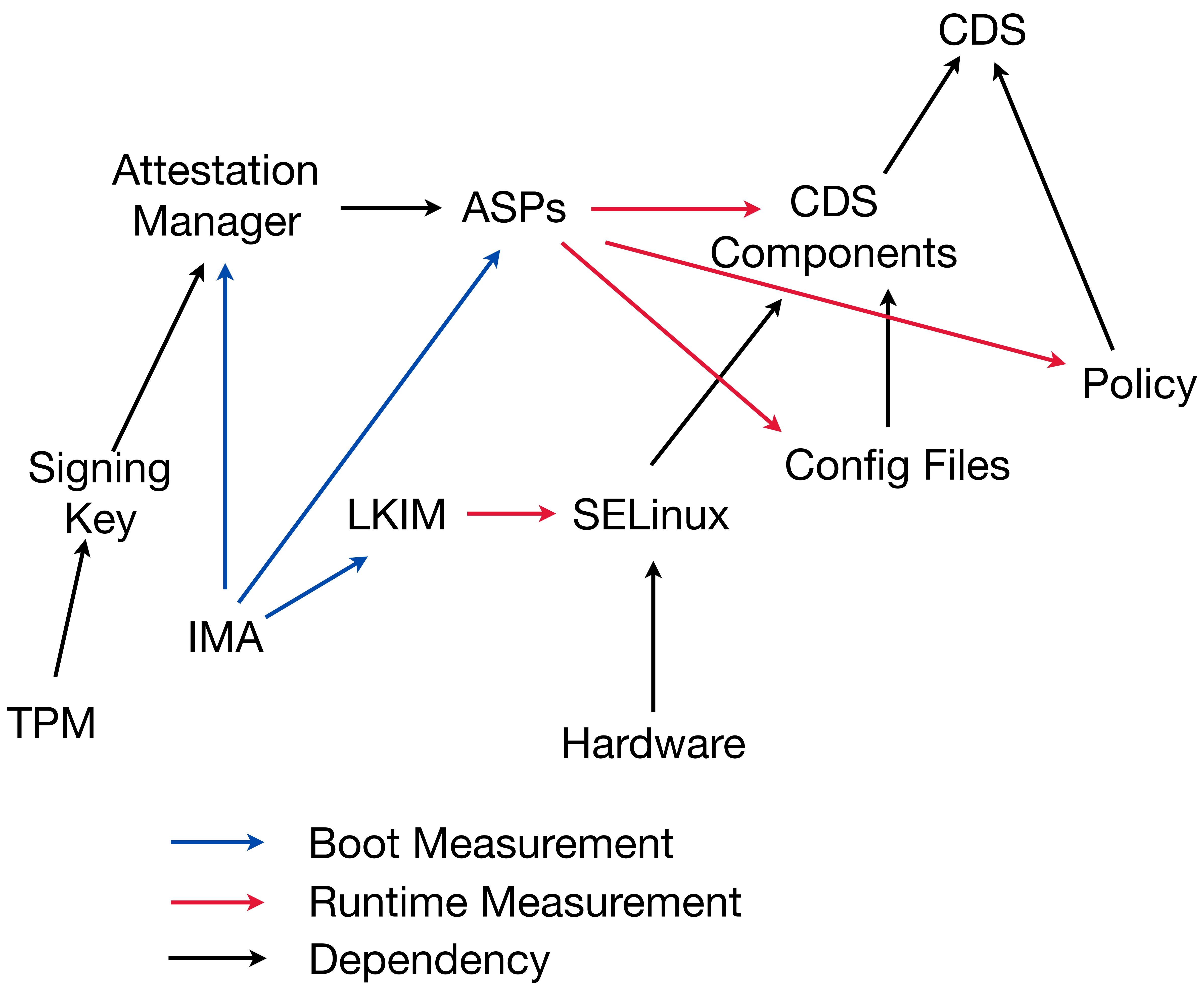
Defining A Protocol

- Establish roots-of-trust
  - measurement - IMA
  - storage - TPM
  - reporting - TPM
- Dependencies are measured first
  - SELinux before AM, CDS
  - AM and ASPs before CDS
  - Configurations before CDS components
- Deeper systems are harder to attack
  - Hardware, TPM, IMA
  - SELinux, LKIM
  - attestation subsystem
  - application software
- Measurement Frequency



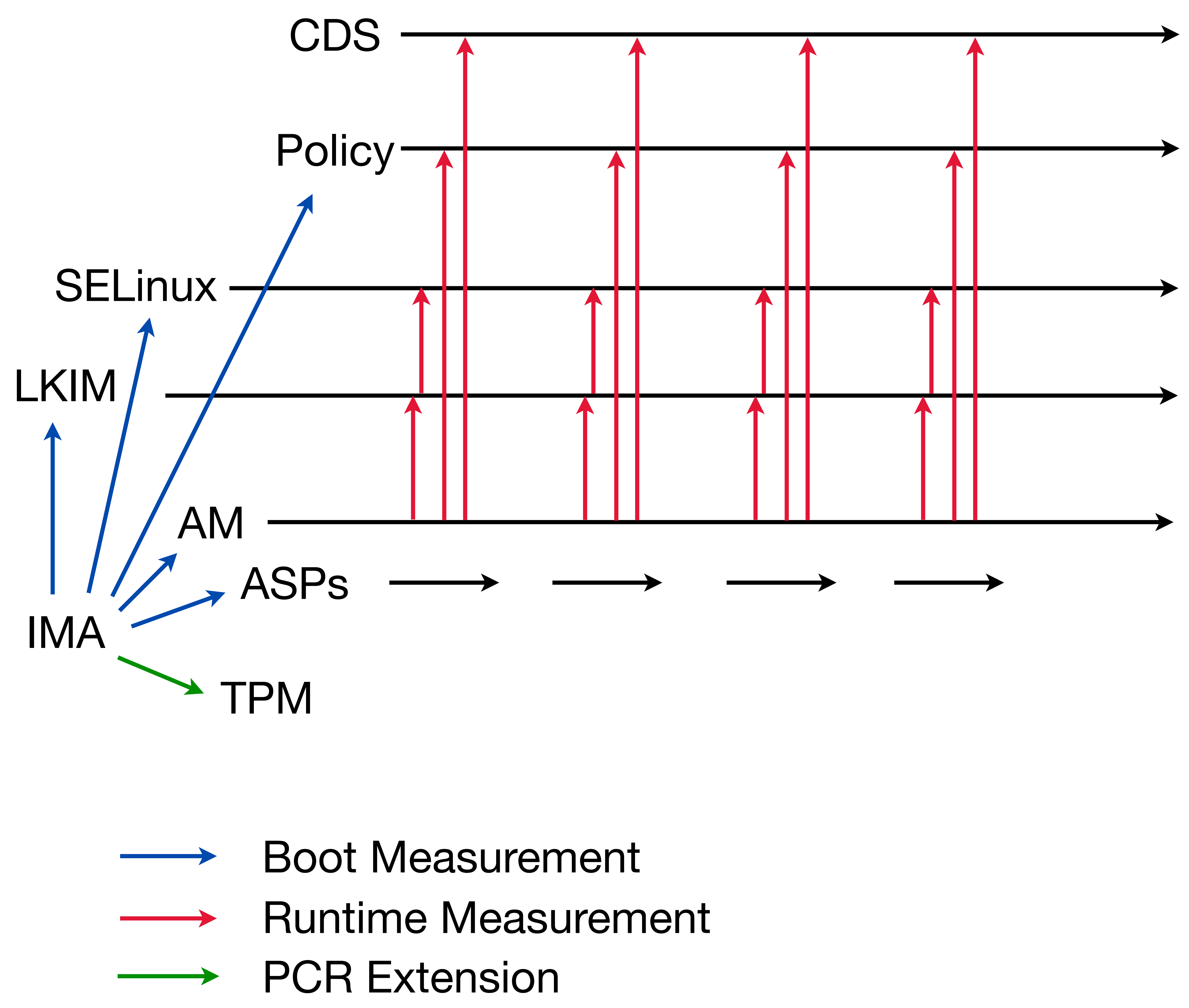
Defining A Protocol

- Establish roots-of-trust
  - measurement - IMA
  - storage - TPM
  - reporting - TPM
- Dependencies are measured first
  - SELinux before AM, CDS
  - AM and ASPs before CDS
  - Configurations before CDS components
- Deeper systems are harder to attack
  - Hardware, TPM, IMA
  - SELinux, LKIM
  - attestation subsystem
  - application software
- Measurement Frequency



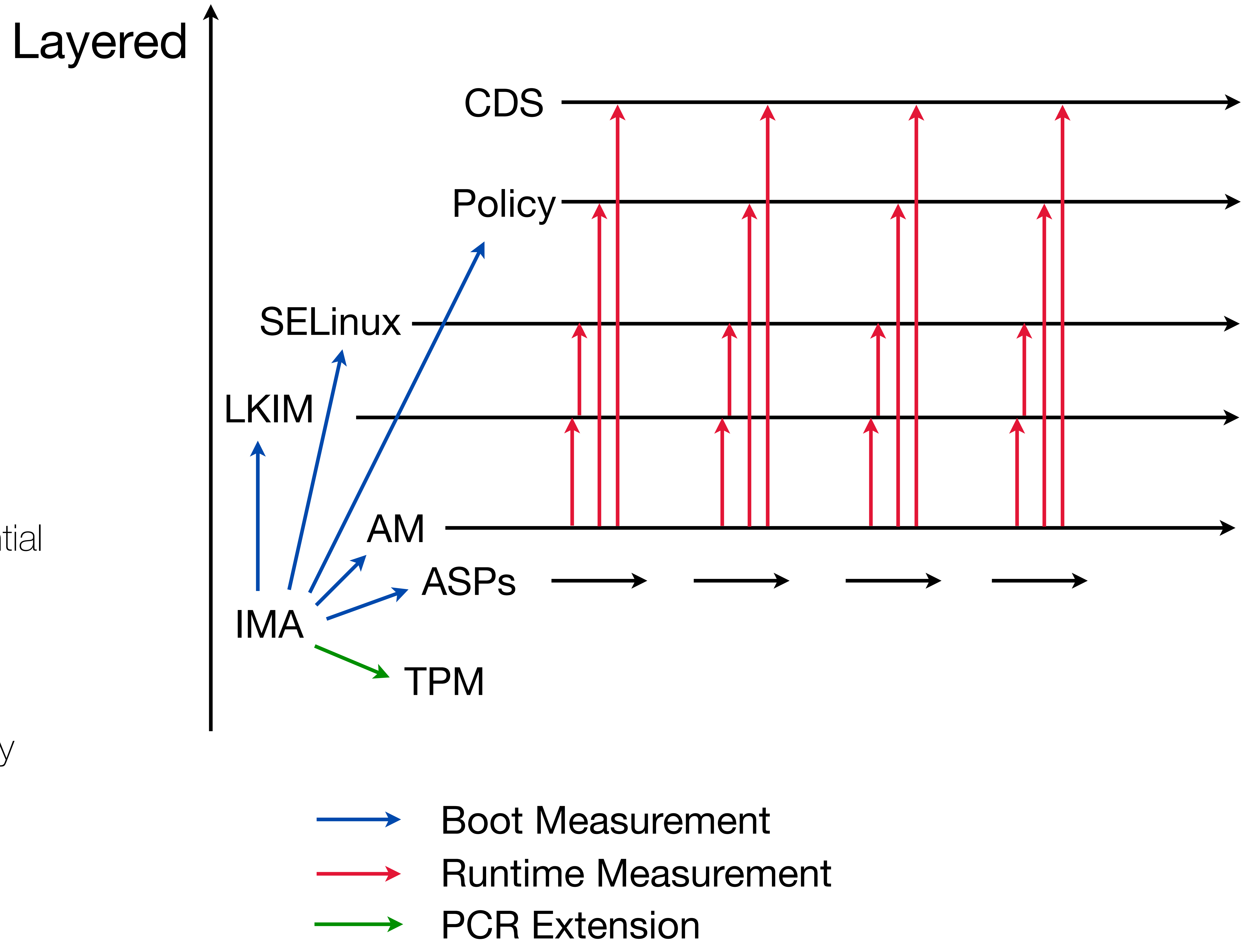
Layered Runtime Attestation

- Boot to an initial measured state
  - establish running AMs with bound keys
  - IMA measures booting system to TPM
  - IMA policy establishes measurement targets
  - TPM memorializes boot state
  - AM key is available on good IMA result
- Remeasure at runtime
  - define protocols to measure systems
  - create ASPs to measure components
  - AMs coordinate attestation protocols
  - AM access using an SELinux protected credential
- TPM IMA PCR is the trust bridge
  - boot measured by IMA into PCR
  - signing key sealed by IMA PCR
  - AM cannot generate good evidence without key
- Layering builds trust bottom up
  - dependencies measured first
  - bundled evidence reflects measurement order



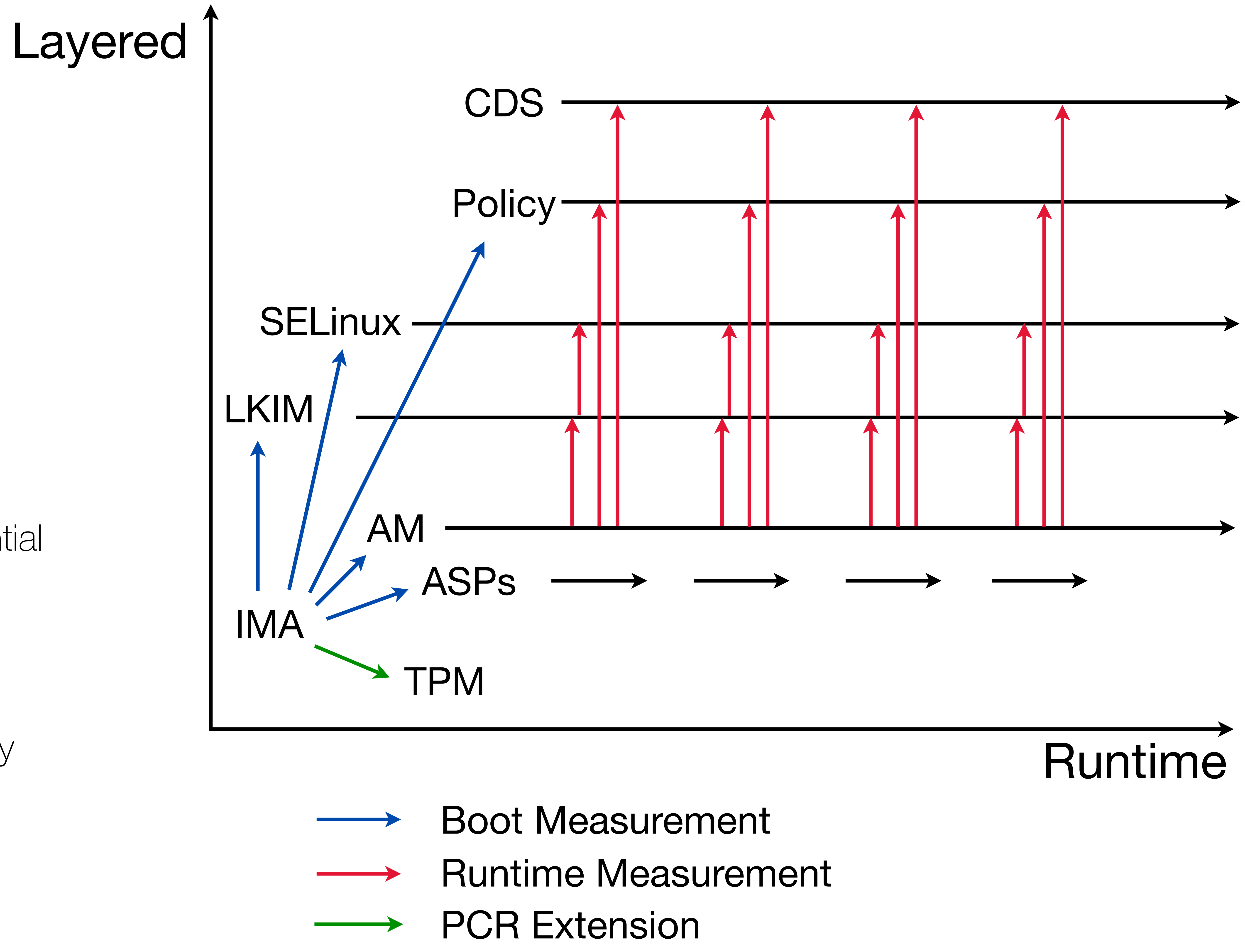
Layered Runtime Attestation

- Boot to an initial measured state
  - establish running AMs with bound keys
  - IMA measures booting system to TPM
  - IMA policy establishes measurement targets
  - TPM memorializes boot state
  - AM key is available on good IMA result
- Remeasure at runtime
  - define protocols to measure systems
  - create ASPs to measure components
  - AMs coordinate attestation protocols
  - AM access using an SELinux protected credential
- TPM IMA PCR is the trust bridge
  - boot measured by IMA into PCR
  - signing key sealed by IMA PCR
  - AM cannot generate good evidence without key
- Layering builds trust bottom up
  - dependencies measured first
  - bundled evidence reflects measurement order



Layered Runtime Attestation

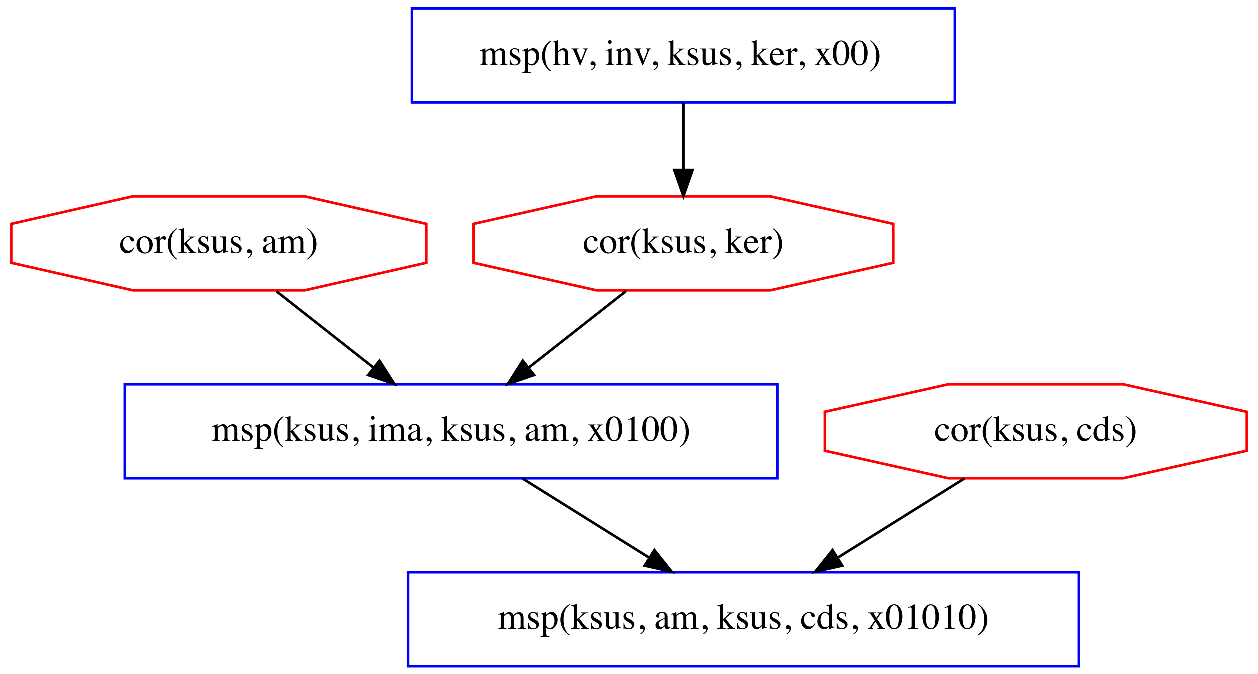
- Boot to an initial measured state
  - establish running AMs with bound keys
  - IMA measures booting system to TPM
  - IMA policy establishes measurement targets
  - TPM memorializes boot state
  - AM key is available on good IMA result
- Remeasure at runtime
  - define protocols to measure systems
  - create ASPs to measure components
  - AMs coordinate attestation protocols
  - AM access using an SELinux protected credential
- TPM IMA PCR is the trust bridge
  - boot measured by IMA into PCR
  - signing key sealed by IMA PCR
  - AM cannot generate good evidence without key
- Layering builds trust bottom up
  - dependencies measured first
  - bundled evidence reflects measurement order

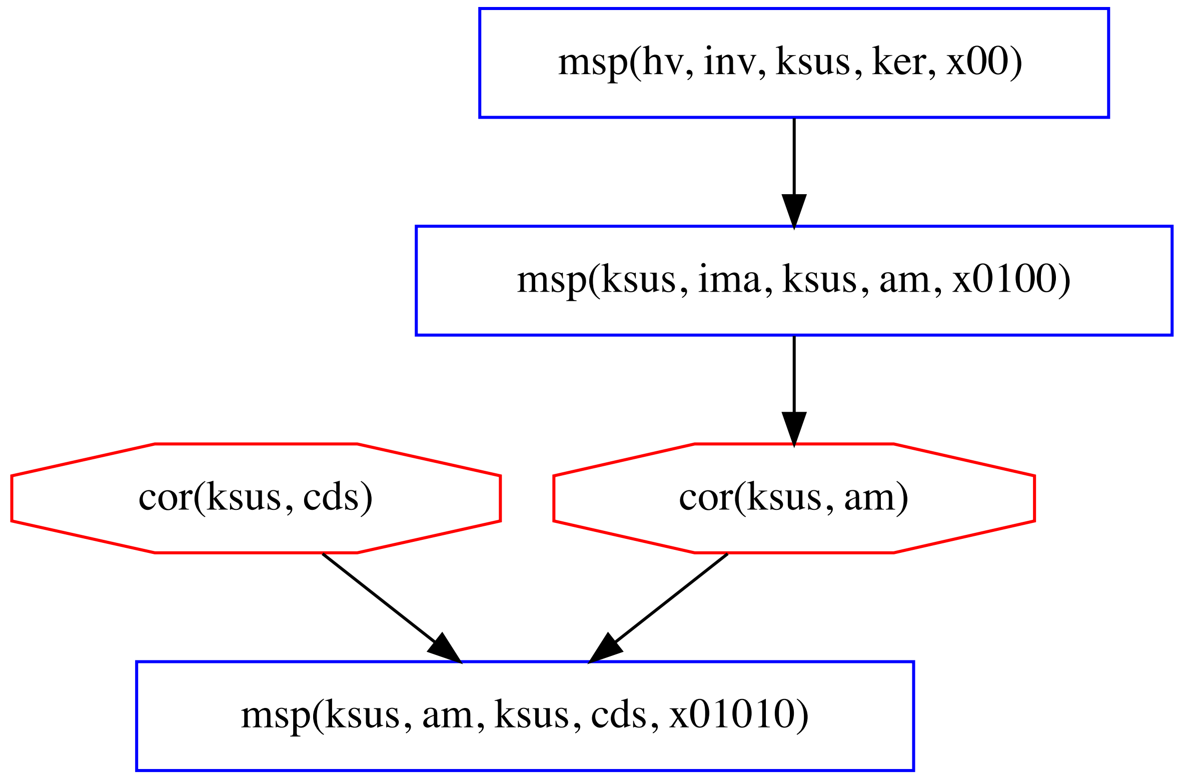


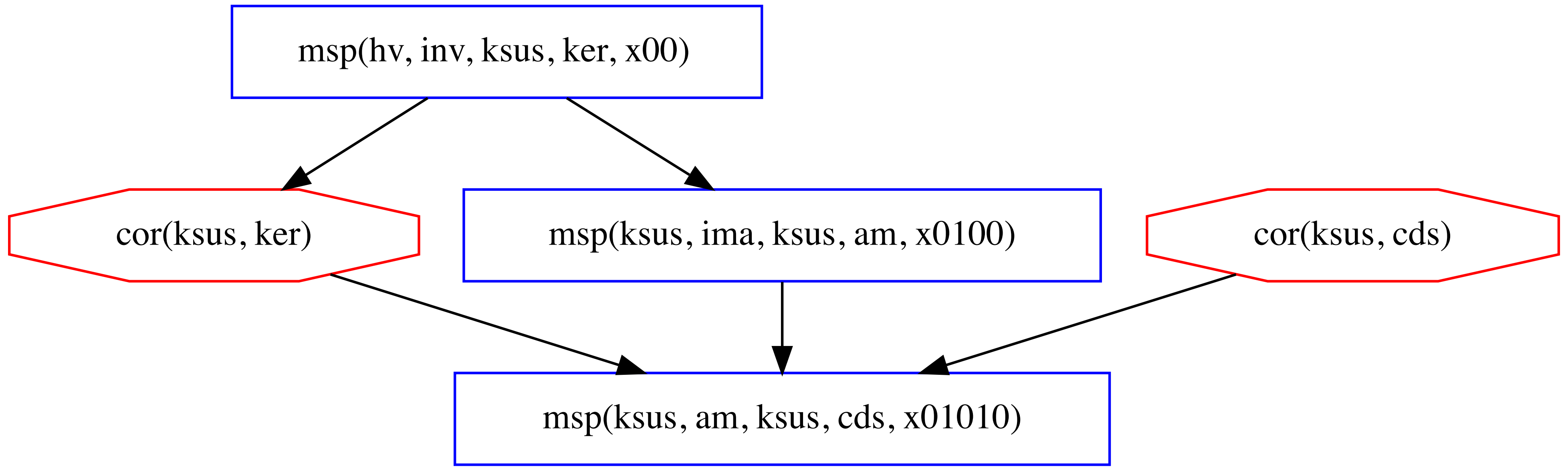
Attack Generation and Testing

- Generate attacks from CHASE outputs
  - CHASE generates all models allowed by a constraint set
  - specialized to generate all allowed attack graphs for a Copland protocol
  - use attack graphs for generating actual attacks on implementations
- Implementing Tradeoff Studies
  - deep vs shallow attestation implementations
  - caching measurements of deep components
  - tradeoff costs and time vs attack detection
- Protocol Ordering
  - formally comparing protocols continuing
  - refinement of the “stronger” concept with utility of evidence
  - heuristics implemented in automated lint-like tools

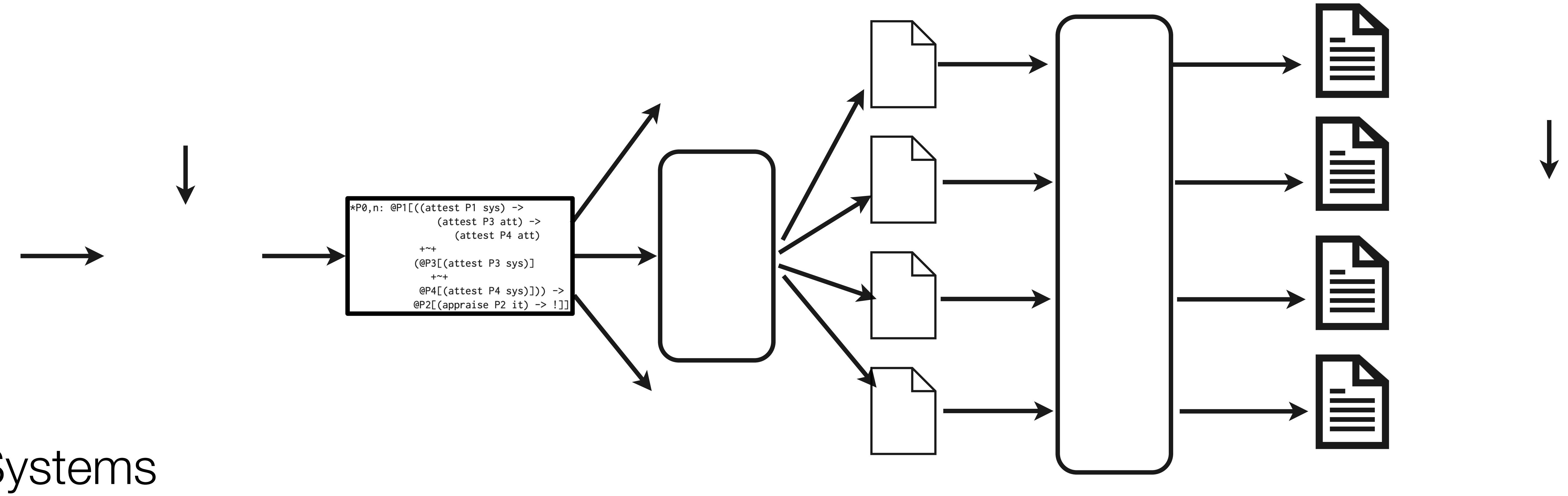
Attack graphs define event orderings in successful attacks







Next Up…



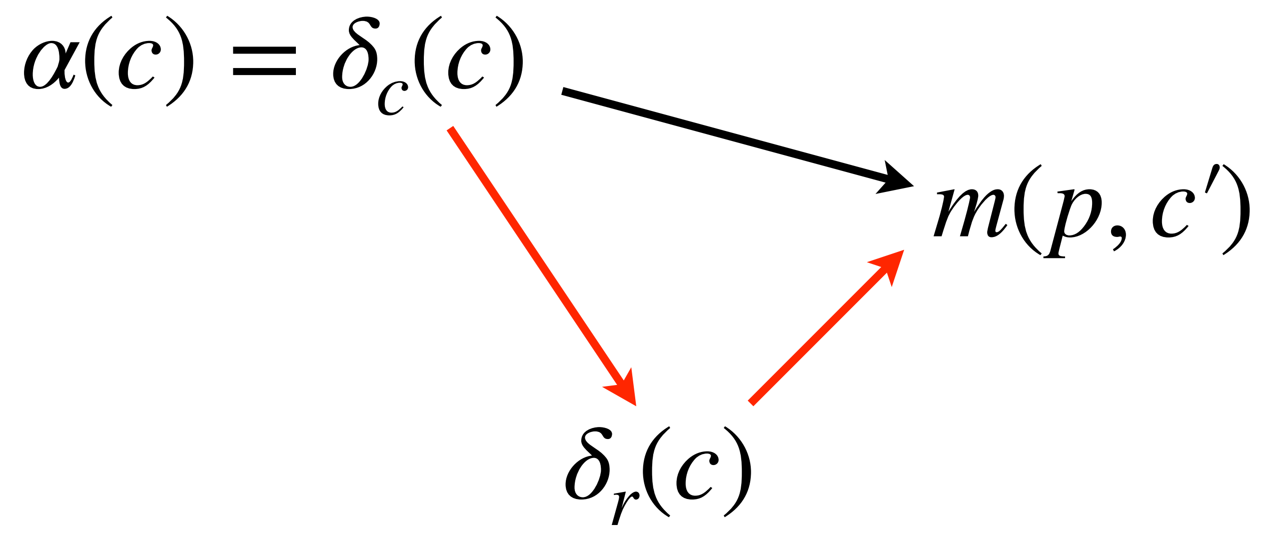
- Protocols From Systems
  - move the user from protocol authoring to system modeling
  - generate protocols from system models
  - include adversary models
- Put Evidence Semantics to Work
  - linter to provide protocol writing guidance
  - type analysis to predict protocol behavior
  - understanding protocol orderings
- Continued Empirical Case Studies
  - account for the adversary in test cases
  - execute long-running case studies
  - trusted AM boot integration
- Users and Publications
  - KCNSC beginning evaluations and training
  - Automated Software Engineering (ASE'25)

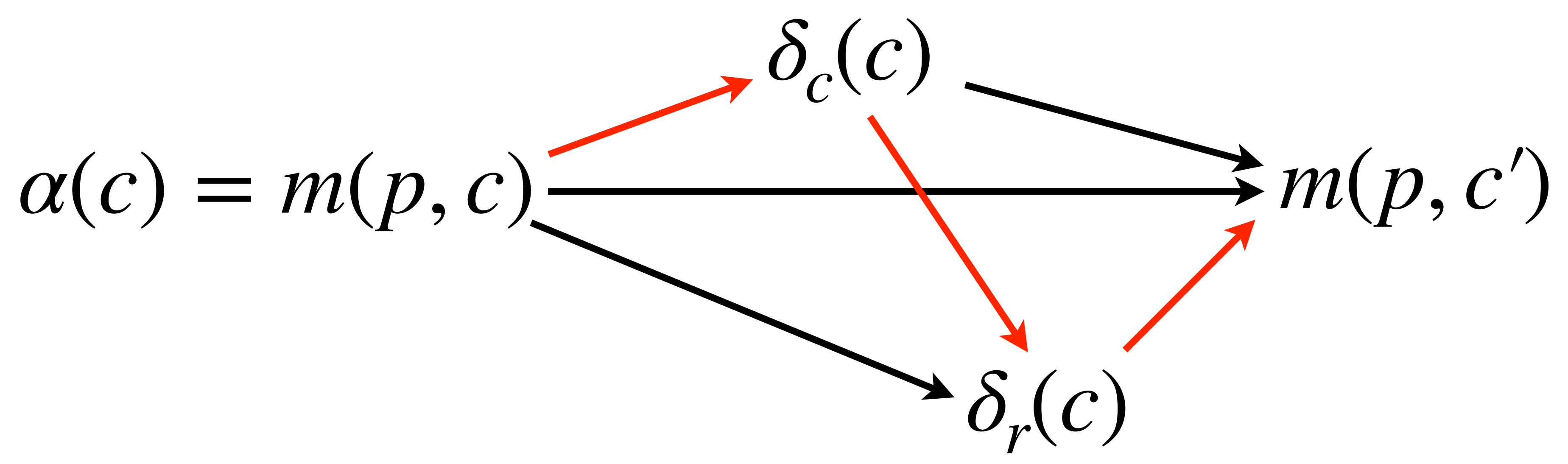
People

- Dr Perry Alexander - PI
  - palexand@ku.edu
- Dr Adam Petz - Research Staff
  - ampetz@ku.edu
- Sarah Johnson - PhD Student
  - sarahjohnson@ku.edu
- Will Thomas - PhD Student
  - 30wthomas@ku.edu
- Logan Schmalz - PhD Student
  - loganschmalz@ku.edu

Evaluating Protocols

- Protocol soundness & sufficiency
  - executability for a given attestation system
  - policy enforcement for a given attestation system
  - soundness = executability + policy enforcement
  - sufficiency defined by protocol ordering
- Ordering Protocols $P_0 \leq P_1$
  - adversary executes the easiest successful attack
  - attestation goal is making the adversary work harder
  - $P_0 \leq P_1$ If the easiest attack allowed by P_1 is at least as hard as the easiest attack allowed by P_0
  - formally defined $P_0 \leq P_1$ verified it is partial order
- Attacks are harder when
  - constrained by measurement timing
  - more required attack events to execute
  - increasing precision and freshness of measurements





Adversary goal is establishing trust in something that should not be trusted