Attack model

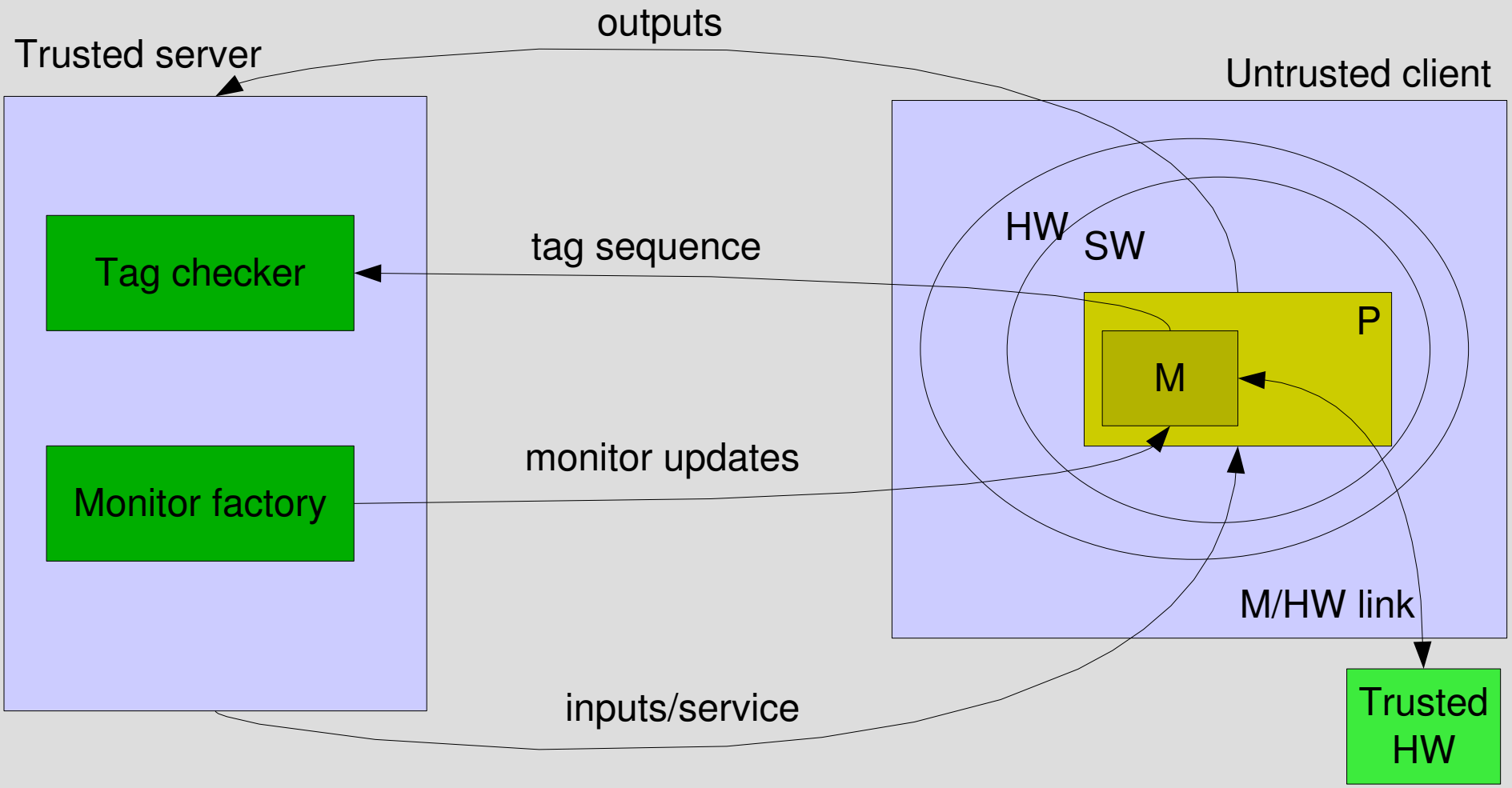
Thomas Herlea
K.U.Leuven – COSIC

RE-TRUST Quarterly Meeting
Trento, 19-20 December 2006

Attack Model Methodology

1. Identify assets (what the entruster cares about)
2. Enumerate attacker goals
3. List attacker capabilities
4. List attacker limitations
5. Describe attacks

Reference Architecture



Assets

- Primary
  - Correct execution of P
  - Complete execution of P
  - Execution limited to P
  - Number of executions of P
- Derivative
  - Confidentiality of cryptographic keys
  - Confidentiality of P+M
  - Integrity of monitor

Attacker – Primary Goal

$$\frac{benefits(P')}{costs(P')} > \frac{benefits(P)}{costs(P)}$$

Attacker – Example Primary Goals

- Increase benefits
  - Eliminate limitations from P
    - No check for expiry date
    - No check for license key
    - Re-enable disabled functionality
  - Change operating parameters of P
  - Run P more times
- Decrease costs
  - Run “lighter” version of P
  - Run P fewer times

Attacker – Derivative Goals

- Reverse engineer P+M
- Fool M about P
- Forge tag sequence
- Forge monitor updates
- Clone P+M's process
- Capture encryption/signing keys

Attacker – Means

Attacker controls:

- Storage media
- Programs
- System libraries
- Operating System: system calls, I/O (entruster, trusted hardware)
- RAM: dynamic attacks
- CPU: tracing, interrupts, virtual memory, timing

Attacker – Limitations

- No faster-than-light communications
- No better than state-of-the-art cryptographic attacks
- Trusted hardware works as specified

Threats to Unprotected Program

- Change program constants
- Change program entry point
- Jump over test instructions
- Call modules from outside the program

Threats to Self-Checkers (1)

- M checks (P+M)'s files, kills (P+M) if files altered
  - Never let control flow reach M
  - Load from tampered files, then restore files
  - Run tampered P, provide original files to M when reading
  - Tamper with P after loading from files
- Conclusion: file checkers are weak

Threats to Self-Checkers (2)

- M checks (P+M)'s process, kills (P+M) if process altered
  - Never let control flow reach M
  - Block killing instructions
  - Snapshot, resurrect killed, try again
  - Wurster's cloning attack
- Conclusion: Wurster's cloning attack must be addressed by RE-TRUST

Threats to Tag Generators

- M generates tags based on measuring P, the entruster continues providing service only if tags OK
  - Forge tags
  - Get tags from untampered clone (Tonella's cloning attack)
  - Replay tags

Threats to M Updates

- M is updated before old M is broken
  - Use memory delta to localize M
  - “Differential Analysis”: Use delta between M versions to reverse engineer M

Threats to Communication with Trusted Hardware

- Forge inputs to THW
- Forge THW responses to SW
- Selectively block SW communication with THW

Unanalyzed Defences

- Run a part of P on THW
- Run M on the THW
- Run monitor factory on M
- Check duration of running M, P
- Use entruster challenges as ingredients of tags

Conclusions

- Deal with Wurster cloning attack (Pioneer?)
- Use trusted hardware for better latency in communication with M
- Use trusted hardware for trusted timestamping
- Use trusted hardware for trusted boot