Integritetsverifikation inom Inbyggda System

Uppdragsbeskrivning inför Examensarbete

Axel Blomén Douglas Fjällrud

2026-01-19

Innehållsförteckning

1. Bakgrund

I sitt begynnelsetillstånd, efter 'reset', exekverar en processor oftast en serie olika initialiseringssteg. Ett karaktärsdrag för dessa steg är att se sker i serie, där kommande steg beror på egenskaper som tillhandahålls av föregående steg. Denna process kallas vanligtvis för 'boot', och själva stegen kallas för 'boot-stages'. Den totala sekvensen av stegen kallas för 'boot-chain'.

Dessa stegen utför förberedelser som behövs för att ladda huvudapplikationen. De flesta av dessa steg är givetvis programmerbara för att möjliggöra den mångfald av användningsområden vi ser datorer inom idag. Dessa steg går alltså att ändra på, och som följd är ofta kritiska steg exponerade för attacker.

Det finns en familj snarlika lösningar som alla går ut på att utnyttja förstasteget, 'boot-rom', som ofta är ett statiskt stycke kod med hög tillitsfaktor, för att kontrollera vissa egenskaper av följande 'boot-stages' redan innan de exekveras.

En av de egenskaperna som kan kontrolleras är kryptografiska signaturer, en sorts matematisk avsändargaranti som garanterar att maskinkoden inte manipulerats av obehörig part. Detta görs oftast genom att beväpna det mycket betrodda förstasteget med ett stycke information som används för att genomföra denna typen av kontroll på nästkommande steg. Nästa steg, som nu har kontrollerats och ärver därmed den höga tillitsfaktorn av föregående steg, fortsätter nu på ett konceptuellt liknande vis i behandlingen av nästa steg. Denna processen kallas för 'chain of trust'.

Det ovan nämnda 'stycke information' kallar man i tekniska termer för nycklar eller certifikat, de nycklarna kommer att ugöra det man kallar för 'root of trust'. Vår idé handlar om att utnyttja hårdvaruegenskaper för att lagra de nycklarna på ett mycket säkert sätt, och att sedan konfigurera denna kedja på ett sätt som medför att vi bibehåller tillit hela vägen fram till huvudapplikationen: kärnan.

När väl kärnan kontrollerats så ärver den samma tillitsfaktor som resten av kedjan, vilket innebär att kärn-nativ funktionalitet kan användas för att göra en rad olika säkerhetsverifikationer och kontroller. Ett exempel av detta är integritetsverifikation av 'read-only' filsystem. I vår lösning vill vi även utforska detta samt vad som är rimligt och möjligt.

Vi har valt att arbeta med Codiax Sweden AB, hädanefter refererat till som "näringsidkaren". Det är ett företag som driver verksamhet inom inbyggda system. De har en bred kompetensprofil och utvecklar driftsäkra system inom allt från medicintekniska lösningar till domänspecifik telemetri inom industriella processer.

2. Syfte

Vårt mål är att assistera näringsidkaren i att möta marknadens allt mer uppmärksammade säkerhetskrav. I moderna system är säkerhet ett kontinuerligt arbete snarare än en ursprunglig egenskap. För att säkerställa långsiktig kvalitet krävs därför rigorösa rutiner och metoder för att löpande verifiera, validera och testa produkter i bruk.

3. Målformulering

Krav- och hotbildsunderlag: Sammanställa ett kort kravunderlag som styr vilka konmtroller som behövs och vilka åtgärder som ska tas vid avvikelse.

Prototyp: Implementera en prototyp som kan verifiera integritet och autenticitet för utvalda kritiska komponenter (till exempel uppdateringsartefakter och vitala delar av filsystem/bootkedja), samt hantera felsituationer enligt definierad policy.

Test och utvärdering: Definiera testfall och visa att prototypen upptäcker avsiktliga manipulationer samt uppfyller grundläggande krav på robusthet och underhållbarhet.

Dokumentation: Beskriva designval, gränssnitt, begränsningar och hur lösningen kav förvaltas och samt vidareutvecklas.

4. Problemformulering

- Vilka är de mest relevanta angreppsvägarna för integritets- och autenticitetspåverkan i målmiljön?
- Vilka mekanismer finns på vald plattform (hårdvara, bootloader, OS/build-system) för att stödja integritetsverifiering, och vilka passar bäst givet krav och hotbild?
- Hur bör ett automatiserat verifieringsflöde utformas för uppdateringspaket vid installation samt kontinuerlig kontroll i drift?
- Hur ska systemet agera vid upptäck av avvikelse, och hur påverkas detta av applikationsdomänens krav?
- Hur kan lösningen testas och utvärderas på ett reproducerbart sätt (testfall, mätpunkter, acceptanskriterier)?

5. Motivering

För näringsidkaren: Ger en konkret prototyp och ett arbetssätt för att möta kunddrivna säkerhetskrav och minska risk för oavsiktliga fel och intrång i levererade system.

För oss: Tillämpning av inbyggda system, systemsäkerhet och verifiering i en realistisk industriell kontext. Samt indruktion till Linux i inbyggda system, Yocto, NXP och U-boot.

För omgivande samhälle: Ökad driftsäkerhet och robusthet i uppkoppalde produkter där fel/manipulation kan få stora konsekvenser.

6. Metod

Litteratur- och standardstudie: Sammanställa releventa principer och etablerade angreppsätt för integritet/autenticitet i inbyggda system.

Krav- och hotbildsanalys: Intervjuer med Codiax för att avgränsa mål, antaganden och acceptanskriterier.

Design och arkitektur: Ta fram verifieringsflöde, policy för avvikelse och hur detta integreras i systemets bygg-/start-/driftkedja.

Test och utvärdering: Genomföra planerade testfall (inklusive manipulationsscenarier), dokumentera resultat och begränsningar.

7. Avgränsningar

Vår implementation är initialt plattformsspecifik eftersom olika arkitekturer tillhandahåller olika mekanismer för dessa typer av funktioner. Vi kommer att specialisera vår lösning för Linux.

8. Resurser

Examensarbetet rör primärt dokumentation och verktyg kopplade till vald plattform och build-miljö (till exempel NXP och Yocto). Bootloader (U-Boot) kommer att behöva studeras för att möjliggöra integration. Specialiststöd från Codiax kommer nyttjas vid behov.

9. Tidsplan

Arbetet planeras över ca 15 veckor (22.5 hp): v7-v23 2026, med halvfart v15-v16 på grund av tentaperiod.

- v7-v12: Research och förstudier
- v12-v18: Implementering och testing
- v18-v23: Rapportslut, poster, opponering och presentationsförberedelser.
- v7-v21: Kontinuerligt rapportarbete (inklusive v15-v16)

Viktiga datum (med önskat presentations datum den 1 juni 2026):

- Fullständing rapport till huvudhandledare: senast 11 maj 2026
- Rapport godkänd av handledare till examinator: senast 18 maj 2026
- Rapport version godkänd av examninator + ge till opponenter: senast 25 maj 2026.
- Presentation: 1 juni 2026 (reserv 2 eller 3 juni).
- Posterutställning och mingel 3 juni 2026
- Workshops: 12 feb, 19 feb, 5 mar, 7 maj 2026.

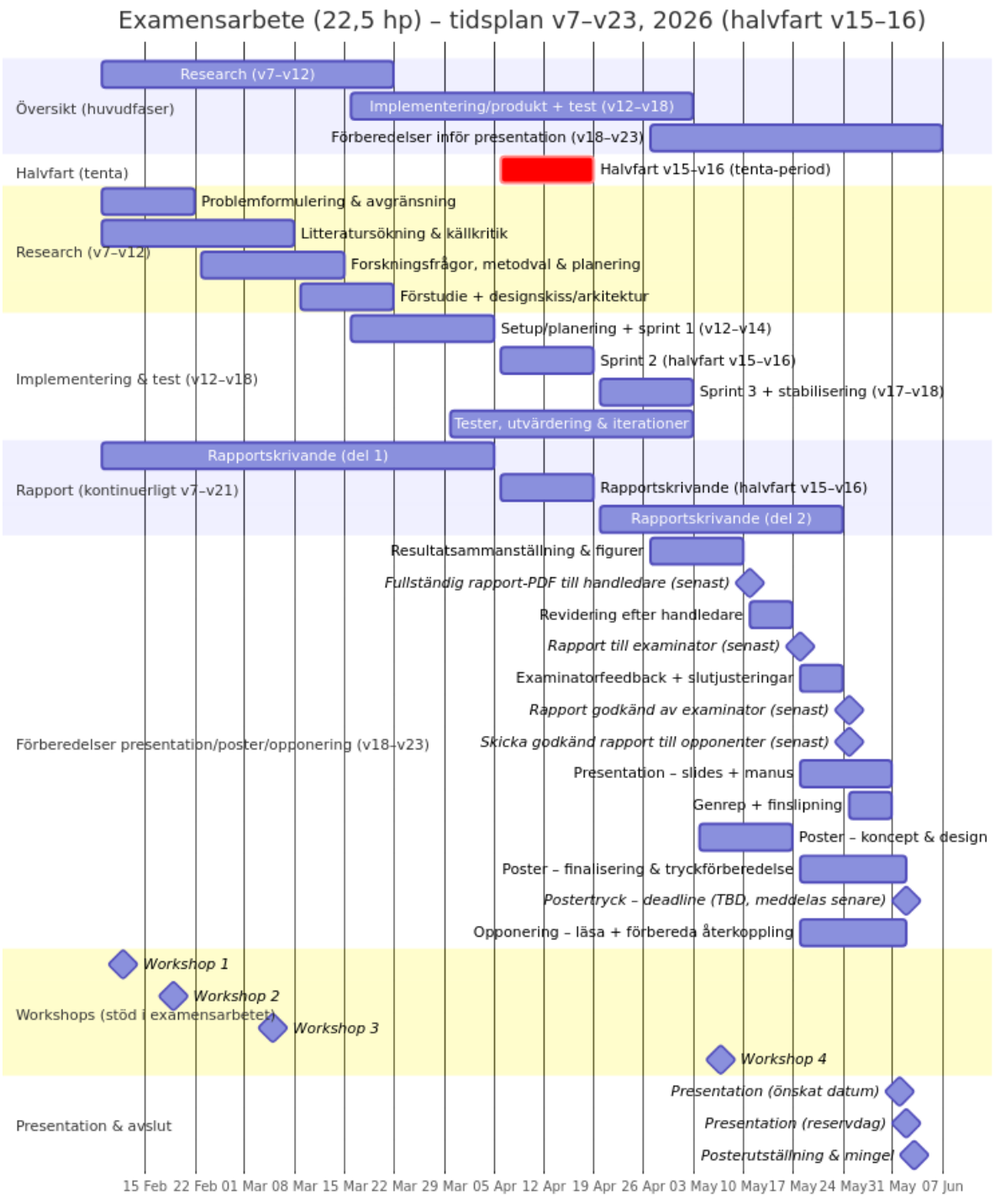

Figure 1: Ganttschema