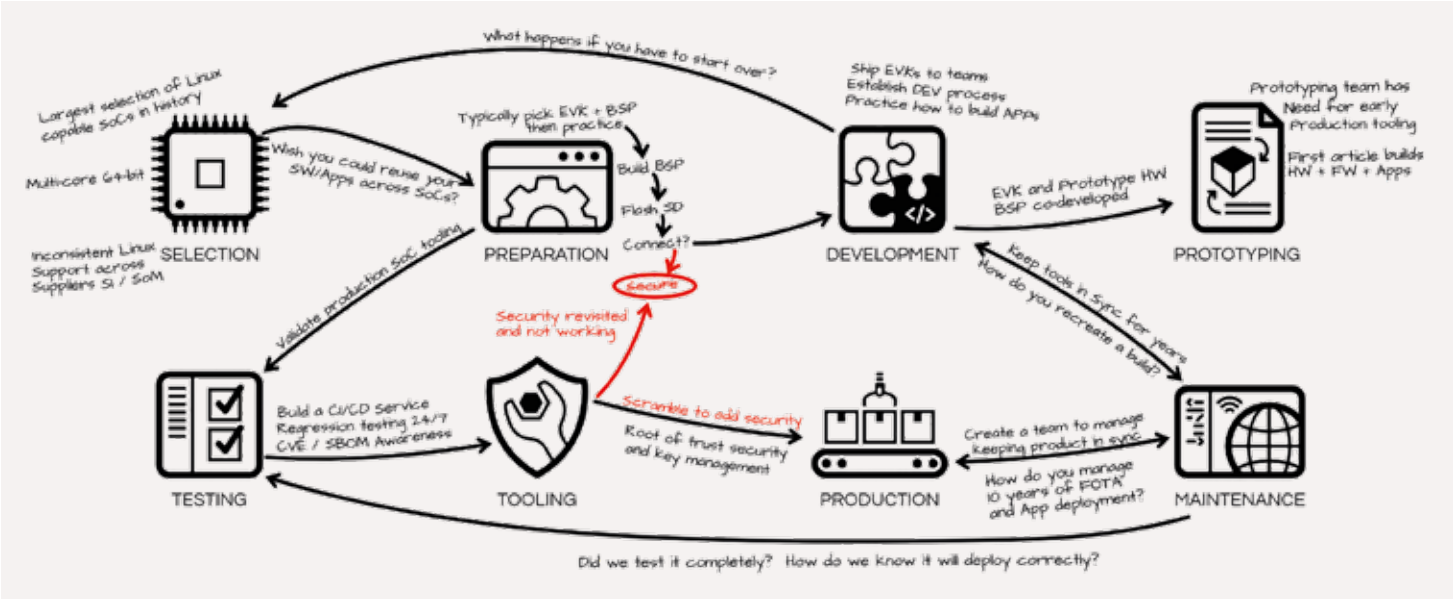


EXPERTARTIKEL

Håll utvecklingens

Tidiga faser i utvecklingsprocessen skapar problem för senare faser genom att ignorera system-funktioner som test, säkerhet och uppdateringar. FoundriesFactory erbjuder en möjlig lösning.

Av John Weil, Foundries

John Weil anslöt till Foundries.io i juni 2022 och är marknadschef. Man kan kalla honom ”systemingenjören i en värld av kisel” som förstått att det egentligen är mjukvaran som förverkligar kapaciteten i ett chip. I Foundries.io kan han ägna sig åt den passionen på heltid – att hjälpa embeddedkunder att suga ut sista droppen ur sina skinande nya SoC:er.

Livscykeln för ett uppkopplat inbyggt system – från utveckling till återvinning – kan sträcka sig över tio år, tjugo år, eller ännu mer. Den består av ett antal distinkta faser som övergår i varandra – åtta stycken enligt vårt sätt att räkna.

Många OEM-tillverkare har problem med att övergångarna mellan faserna skapar fragmentering, både organisatoriskt och tekniskt.

Det rationella är att konstruera produkter så att de är enkla och billiga att underhålla, uppdatera och säkra efter att de tagits i drift. En produkt utgör en enda ekonomisk och fysisk enhet under hela sin livstid. Underhållet måste räknas in i produktens kostnad.

Så när man beräknar kostnader och intäkter för en produkt måste hela livscykeln räknas in och inte bara försäljningspris (och service i förekommande fall). Tyvärr motverkar fragmenteringen detta. Beslut som fattas och åtgärder som vidtas i tidiga faser av utvecklingen får effekter i senare faser (test, produktion och underhåll).

Kollisioner mellan teamen och det de producerar kan inte bara försena lanseringen av en produkt utan också göra den mer utsatt för sårbarheter, öka risken för ekonomiska skadestånd och äventyra att det finns en väl fungerande lösning för uppdateringar.

Ofta tas säkerhetsfrågorna upp på allvar först under senare stadier av produktutvecklingen. Det skapar problem längre fram.

Alla dessa förvecklingar och konflikter kan undvikas. Visserligen är viss organisatorisk fragmentering oundviklig – det är naturligt att utveckling bedrivs separat från underhåll. Men för utveckling av embeddedsystem existerar plattformar som spänner över hela livscykeln.

Den här artikeln kommer att belysa de åtta stegen i en embeddedprodukts livscykel. Vi kommer dessutom att demonstrera värdet av en ny typ av DevSecOps-plattform. Den används genom hela utvecklingsprocessen och ger dig en produkt som uppfyller krav på utveckling, livslång säkerhet, underhåll och fleet management.

Vad de åtta stegen är värda och vad de kostar

Vår ”åttastegsmodell” för en produkts livscykel är mer eller mindre universell för alla typer av uppkopplade inbyggnadssystem, i alla marknadssegment.

Stegen är följande:

1. Kärnkomponent. Bestäm vilken mikroprocessor eller FPGA, mikrokontroller, grafikprocessor eller SoC som hårdvaran ska byggas runt. Valet har djupgående konsekvenser för vilken typ av programvara som produkten kan köra och vilka tillämpningar den kan stödja.

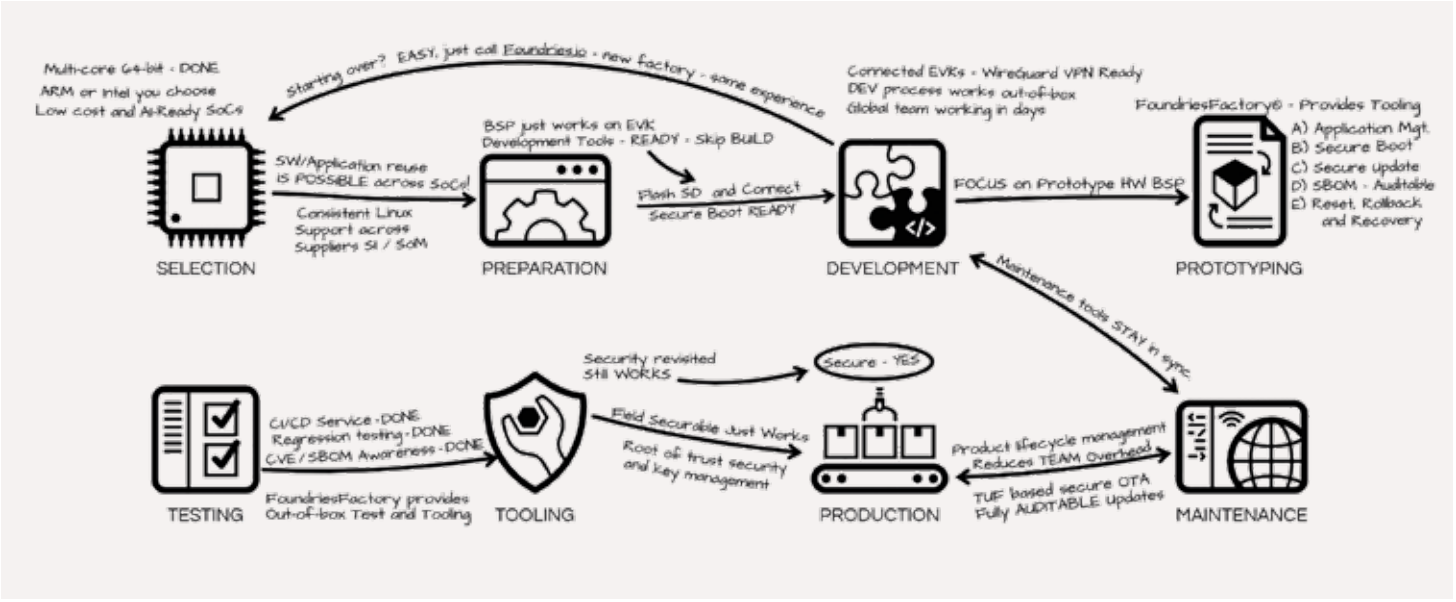
2. Förberedelse. Utvecklarna bygger experimentellt upp en verktygskedja för programvaruutveckling på den komponent som valdes i första steget. Här verifieras att chipleverantörens supportpaket för kortet fungerar, inklusive grundläggande funktioner som uppkoppling, att flasha bootkod och att konfigurera en verifierad bootsekvens.

3. Utveckling. Efter steg 2 finns grundläggande hårdvara. Därmed kan steg 3 utveckla den tillämpning som utgör produktens egentliga värde.

I den här fasen bör utvecklaren vänta sig det oväntade.

Marknadsförhållanden kan förändras, liksom tekniska omständigheter. Det kan leda till krav på förändringar av de verktyg, programvaruplattformar och säkerhetsprocesser som fastställdes i förberedelsefasen, eller till och med i valet av krets under första steget.

4. Prototyp. I de tre första stegen dokumenterades att kärnkomponenten på utvärde-



©2023 - ALL RIGHTS RESERVED. FOUNDRIES.IO

faser i fas

ringskortet kunde leverera merparten av önskad funktionalitet. I prototypfasen byggs nu programvara med komplett funktionalitet på en prototyp som är anpassad även vad gäller formfaktor.

I det här skedet tar produktions- och utvecklingsteamet tillsammans fram en DFM (design för tillverkning). Plattformsutvecklingsteamet måste både upprätthålla en BSP för denna hårdvara och för den ursprungliga utvärderingssatsens BSP.

5. Testning. Testteamet verifierar att tillämpningen fungerar enligt plan. Och ser till att det kommer att gå att testa och validera uppdateringar av firmware och tillämpningsprogram när produkten är i drift.

6. Produktionsverktyg. De system som krävs för att stödja volymproduktion implementeras. I den här fasen kan man i värsta fall upptäcka att de verktyg som användes för att producera prototyper inte stöder volymproduktion.

Observera att det är avgörande att säkerhetsfunktioner som root-of-trust och nyckelhantering finns på plats.

7. Produktion. Det här är det avgörande ögonblicket då OEM-tillverkaren får reda på om konstruktionen – efter allt arbete som lagts ned så långt – kommer att gå att volymproducera.

Det är extra viktigt att säkerställa att produktionsflöden för säkerhet (som upprättande av root-of-trust och inloggning i molntjänster) är kompatibla med den mekanism som används för att uppdatera firmware. Det stödet måste finnas under resten av produktens livstid.

8. Underhåll. Juridiskt ansvar, varumärkesvård och ny lagstiftning är olika anledningar till att tillverkare regelbundet uppdaterar sina produkter efter försäljning. Dels för att minska exponeringen för säkerhetshål och dels för att upprätthålla produktens funktion i linje med köparens förväntningar.

KOSTNADEN FÖR DE FÖRSTA SJU STEGEN, och värdet de genererar, bestäms av interna faktorer, som utvecklarnas skicklighet och engagemang, och hur effektivt de styrs.

Med det åttonde steget är det lite annorlunda. Omvärlden påverkar hur ofta en produkt utsätts för cyberattacker och hur allvarligt utfallet blir. Det är också externa faktorer som avgör hur ofta miljöförändringar – som exempelvis krav på nya inloggningsuppgifter i molnet – gör det nödvändigt att distribuera produktuppdateringar under drift.

Om man misslyckas med att åtgärda en CVE-varning (Common Vulnerabilities and Exposures) eller om produkten råkar ut för cyberangrepp, kan man drabbas av juridiska och tekniska kostnader som kan vara så stora att de i värsta fall utplånar lönsamheten med hela produkten.

Effektiva lösningar för underhåll utgör ett starkt skydd mot den typen av risk. Alltför ofta äventyras dock effektiviteten i det åttonde steget av beslut fattade i tidigare steg.

EN PRODUKTCHEF MÅSTE TYVÄRR alltid vara beredd på att en CVE-varning kan dyka upp när som helst, och efter den en chef som kräver svar:

- Vad gjorde vi för att undvika den här sårbarheten?
- Var åtgärden effektiv?

Detta är det ögonblick då du kan drabbas av insikten att din utvecklingsprocess har satt underhålls- och driftsystemen ur spel.

Fragmenterade utvecklingsprocesser

Att produkten blir extra utsatt för risk under hela sin livstid – det kan bli konsekvensen av en fragmenterad utvecklingsprocess. Den kompetens och de verktyg, strukturer och ledningsprocesser som krävs under utveckling och prototypning är inte desamma som de som krävs vid test, produktion och underhåll.

Under de olika faserna byggs det upp en organisatorisk separation mellan teamen, med olika incitament och olika tidshorisonter. Det leder till att team i värsta fall i slutänden fattar beslut endast utifrån sitt eget snäva perspektiv, snarare än utifrån hela produkten.

Det är till exempel bra om den centrala komponenten får stöd för säkerhet redan i förberedelsefasen (andra steget). För om du skjuter upp det till ett senare skede tar det längre tid och kan det innebära kostsamma och tidskrävande omarbetningar av firmware och mjukvara.

Problemet är att utvecklarna under det tidigare steget värderades och belönades efter hur snabbt de kunde ta fram en fungerande plattform att gå vidare och bygga en tillämpning på. De hade därmed ett incitament att hoppa över säkerheten.

På samma sätt är det med SBOM (software bill-of-materials). En sådan loggar vilka programvarukomponenter som används i olika varianter av produkten och kan göra stor nytta under utvecklingsfasen.

Cisa (US Cybersecurity and Infrastructure ▶▶

Security Agency) och andra Institutioner förespråkar kraftigt att du underhåller en omfattande SBOM. Det börjar även ingå i kravställningen för programvara för myndigheter.

Problemet, återigen, är att en SBOM inte är något som påskyndar produktens utveckling (tredje steget). Snarare bromsas den. Så det finns inget incitament för utvecklingsteamet att skapa den. Därför saknar många produkter från OEM-tillverkare idag en komplett SBOM.

Det gäller generellt att de sju första stegen har utvecklats i riktning mot att ta allt mindre hänsyn till intressen och behov i det åttonde steget.

DET FINNS OFTA EN KULTUR bland produktutvecklare att det finns viktigare saker att ägna sig åt än säkerhet, underhåll och uppdateringar. En DevSecOps-plattform (DevSecOps) ser till att säkerhets- och underhållsaspekter hela tiden finns på dagordningen.

Det kan kosta på ordentligt i både kronor och goodwill att falla offer för en säkerhetsbrist. Dessutom är nya myndighetskrav på cybersäkrade produkter under utveckling. Det har övertalat OEM-tillverkare att börja leta efter verktyg som hjälper dem att introducera stöd för underhåll tidigt i produktutvecklingen.

En lovande approach är den strategi som allmänt kallas DevSecOps (development, security, operations). Plattformar för DevSecOps baseras på en kustomiserbar Linux med brett stöd för olika processorer. Här finns verktyg för att hålla samman utvecklingsflödets alla steg – säkerhet, aktivering av kort, underhåll, fleet management – under en hel produktlivscykel.

GENOM ATT ANVÄNDA en integrerad DevSecOps-plattform överbryggar du etapperna i livscykeln. Till exempel:

- Stöd för SoC:er från de ledande chiptillverkarna innebär att steg 1–3 kan hanteras parallellt och i synk.
- Det går att provköra tillämpningar på både utvärderingskort och egendesignad hårdvara och samtidigt hela tiden fortsätta designa för tillverkning.
- Utveckling för utvärderingskort och förproduktionsflöde kan bedrivas i samma framework för build, secure och deploy som det som används under steg 5–7.
- En SBOM genereras och uppdateras automatiskt för varje instans av produktens firmware och tillämpningar.

Under produktion levereras varje produktionsenhet som lämnar fabriken med sin egen SBOM. Därmed kan en produktchef omedelbart och enkelt svara på frågor om en produkts exponering för en viss CVE-sårbarhet.

- Produktionsverktygen är kompatibla med utvecklingsverktygen. Det betyder att det går att återskapa en tidigare systemimplementering på några minuter, migrera en gammal konstruktion till ny firmware, tillhandahålla en detaljerad uppdateringslogg för produktens hela livslängd och hantera programuppdateringar.
- Alla tidiga utvecklingssteg är helt integrerade med de verktyg som används för fleet management och krypterade programuppgraderingar

OEM-TILLVERKARE som gör produktutveckling på en DevSecOps-plattform – som exempelvis FoundriesFactory från Foundries.io – kan eliminera de säkerhetsrisker och underhållsproblem som uppstår till följd av fragmentering mellan organisatoriska siloer.

En enhetlig plattform med gemensamma verktyg hela vägen från steg 1 till 8, ger större möjlighet att utveckla och driftsätta nya funktioner. Dessutom öppnar det för att addera nya intäktsmöjligheter till en redan installerad bas av uppkopplade produkter.

Om du använder en DevSecOps-plattform kan du alltså förvandla steg 8 från en risk (för dina pengar och ditt rykte) till en intäktsgenererande möjlighet. ■

Tror du att allt står på webben?

Läs Elektroniktidningen!

PRENUMERERA GRATIS

Du får månadsmagasinet genom att fylla i talongen på

www.etn.se/pren

ELEKTRONIK TIDNINGEN