

Säkerhetsutmaningar inom UAV-utveckling

Obemannade flygfarkoster får en allt viktigare roll inom både militära och civila områden. Men för att klara säkerheten krävs tekniker som separationskärnor. Paul Parkinson, systemarkitekt hos Wind River redogör här för UAV-läget.

Den ökade utvecklingen av och de skiftande uppdragen för obemannade flygfarkoster (UAV) har påvisat problem rörande informations säkerhet (InfoSec) och kommunikations säkerhet (ComSec) som inte är lösta av åtgärda i traditionella federerade eller nyligen utvecklade integrerade modulära flygelektronik- (IMA)-system. Behov av att ha militära UAV:er i funktion i civila luftstrum som kommunicerar över icke-hemligstämplade linjer till främmande flygledningsystem, samt att hålla kändlig och/eller hemligstämplad information separat utan att öka utrymme, vikt och kraft (SWaP), ställer till med problem för UAV-systemens arkitektur. I denna artikel diskuteras programvaruarkitekturer med flera oberoende säkerhetsnivåer (MILS, multiple independent levels of security) i förhållande till hur de kan tillgodose dessa oberoende krav på konstruktionen av UAV-system.

TILLKOMSTEN AV OBEMANNADE SYSTEM

På senare år har utvecklingen och nyttjandet av UAV:er sett en mycket snabb tillväxt. Dessa obemannade system utnyttjas i många olika roller, från titörtspaning till långvariga operationer på hög höjd (HALE - high-altitude, long-endurance). I många fall har programutvecklingen drivits främst av operativa behov att utnyttja system i miljöer som bedöms vara för farliga eller för fientliga för mänskliga operatörer. Utvecklingen av dessa obemannade system har dock inte klarat de tekniska utmaningarna med att tillgodose dessa operativa krav och samtidigt ge ytterligare faktiska fördelar.

UAV:er behöver inte belämnas med livsuppehållande system för en mänsklig operatör, vilket ofta leder till en fysiskt mindre och lättare utformning än ett bemannat fordon. Många UAV:er kan bidra till ett minskat radaravstånd (RCS), vilket resulterar i

lägre sannolikhet att meddelanden upptäckas (LPI) av fiendestyckor. Detta kan även minska behovet av att använda de smyg-, överljuds- eller hypersoniska möjligheter som för närvarande används för spionplan på hög höjd (såsom U-2, SR-71 och efterföljande flygplan).

UAV:er kan också utföra mycket längre, utökade uppdrag än de som begränsas av att de har en individuell mänsklig operatör. Detta eftersom de ger möjlighet att koordinera funktionen via ett antal skiftarbetande fjärroperatörer. Denna möjlighet ger militära planerare förmåga att ha ökad ströktid på målet, vilket kan vara ovärderligt för ett förtärligt läge på marken.

Dessutom har det den fördelen att militära UAV:er kan användas i krigsskådeplatser som utgör större risk för upptäckt än som skulle vara acceptabelt för bemannade flygplan, inklusive flygspaning för att tillhandahålla ovärderlig information från slagfält. De kan även användas som aktiva lockflyg, tränga sig djupt in på fiendens område i offensiva luftangrepp. Bemannade stridsflygplan eller UAV:er (se fig 1).

EXTREMT MODULÄR FLYGELEKTRONIK

Flygplanssystem blir alltmer komplexa för att implementera mer och mer avancerade funktioner. Därmed förtärligt mjukvaruutvecklingen i dessa system att växa med förväntad hastighet. Under 1980-talet låg exempelvis mjukvaruutvecklingen i flygelektroniksystem i militära snabba jetplan på runt 100 000 källkodlinjer (SLOC), och detta har ökat väsentligt på senare år. Det har beräknats att F-35 Lightning II kommer att ha runt 7 miljoner SLOC.

Den ökande komplexiteten i dessa system har krävt ökad systemprestanda och därmed de fysiska försvårade vad gäller SWaP. Dock finns det trängande behov av att minska systemens fysiska fotav-



Fig 1. Obemannat stridsflygplan (UCAV)

tryck i flygplan och framför allt i UAV:er tack vare deras begränsade fysiska storlek. Så en väsentlig ökad processdensitet är nödvändig.

Detta problem åtgärdas genom antagandet av IMA. Denna arkitektur består av gemensamma beräkningplattformar som kan agera värd till flera applikationer samtidigt (se fig 2). Detta tillvägagångssätt sparar utrymme, kraft och vikt. Om en öppen standardbaserad mjukvaruarkitektur används istället för en stängd egenutvecklad mjukvaruimplementering, som kan vara fallet även med en del COTS-implementeringar, kan det ge ytterligare fördelar i fråga om förbättrad modularitet, samfunktion och mjukvarans överförbarhet.

Mjukvaruarkitekturen ARINC 653 är ett exempel på ett öppet standardbaserat tillvägagångssätt och har blivit överlägset på senare år. Det ger en specifikation för applikationsseparering för IMA-system och är baserad på principerna för robust temporal och spatial partitionering. Dessa är grundläggande krav för att möjliggöra att flera applikationer på olika säkerhetsnivåer körs samtidigt på samma processor. Detta är mycket önskvärd eftersom system av blandad betydelse tidigare måste ha hela mjukvaran säkerhetscertifierad till nivån för den allra viktigaste applikationen, vil-



Fig 3. UAV-uppdrag

ket väsentligt skulle påverka certifieringsåtgärderna och kostnaderna.

ARINC 653 definierar en APEX (application executive), som tillhandahåller ett programeringsgränssnitt för applikationen (API - application programming interface) på 51 rutiner. Detta gör att möjliggöra utveckling av bärbara applikationer på en IMA-plattform, som stöder temporal och spatial partitionering tillsammans med kommunikation mellan applikationer i olika delar via väldefinierade portar. ARINC 653 definierar även ett ramverk för hälsostyrning, som kan användas för att ge ett hierarkiskt ramverk för feldetektering och återställning. Detta ramverk ger en ökad nivå av felolerans, vilket kan utnyttjas för att uppnå förbättrad operativ tillgänglighet.

ARINC 653-standarden har förtidats på senare år genom cyklisk uppeppning av standardiserings- och utvecklingsfarenheter som erhålls genom de säkerhetskritiska IMA-program där den har utnyttjats. Åtgärder för att ta fram riktlinjer för säkerhetscertifiering av IMA-system under RTCA DO-178B3/EUROCAE ED-12B4 har också involverat en upprepningscykel, med erfaren-

heter från veddiga program och nya krav tillbakamåttade in i standardiseringsprocessen. Detta har lett till ett rollbaserat utvecklingsstätt för att underlätta att certifieringsprocessen definieras i de gemensamma riktlinjedokumenterna DO-2975 och ED-1246 framtagna via samarbete i arbetsgrupper gemensamma för RTCA och EUROCAE. Detta rollbaserade utvecklingsstätt, som involverar plattformslieferant, systemintegratör och applikationsutvecklare, kan stötta i ARINC 653-implementeringar.

HEMLIGA UPPDRAG

UAV:er innefattar ett antal system som utför olika funktioner. Dessa inkluderar flygelektroniksystem, uppdragsystem och sensorsystem. Dessa utnyttjar brandväggar och kryptering för säker överföring och mottagning av information mellan nätverk, utan att informationen omvandlas under transporten. Detta kallas kommunikations säkerhet (communications security - ComSec). Det finns exempelvis troligen datakommunikation mellan flygelektroniksystem och uppdragsystem som godkänns lägesinformation, orientering, höjd och så vidare. Information kan dock även be-

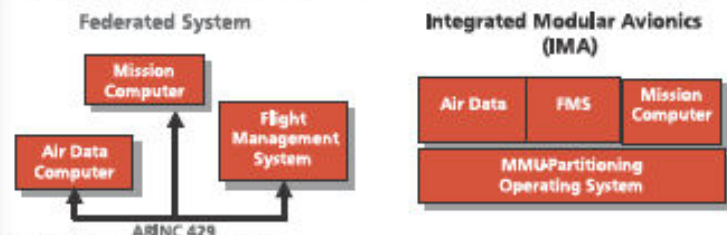


Fig 2. Arkitekturer för flygelektronik

LINUXSPALTEN

Linux tar tid

Går det att leva med en Linuxbaserad dator som huvudsystem? Svaret är naturligtvis ja, men det kostar på – inte pengar men tid.

Att vara, som jag är, övertygad om att öppen källkod är den bästa formen för att sprida programvara är inte svårt. Det är också lätt att försvara den bredd som Linux har när det gäller återanvändning av äldre hårdvara. Dessutom finns det reella möjligheter att förestå bar den dator du har, oavsett konfiguration eller typ fungerar.

Men det tar tid! Tid som du oftast inte har. Sist hände det mig när jag lite slarvigt accepterade alla uppdateringar som uppdateringshanteraren till mitt Linuxsystem föreslog. Felet var mitt eget. Något som inte gör det lättare för mig att leva med företeten.

Kom ihåg att Linux sätts samman av en mängd olika bidragsgivare. Många uppdateringar när aldrig dig beroende på hårdvarumässiga eller programvarumässiga trösklar men ett stort antal uppdateringar blir det ändå. Att det någon gång slinker in en, för dig, oönskad uppdatering är nästan nog oundvikligt. För mig hängde sig systemet, eller snarare fönstermiljön, efter en uppdatering. Jag förpassades snabbt tillbaka den testbaserade Linux jag arbetat med tidigare. Ny hårdvara men samma gränssnitt som jag använde redan 1995 när jag började men Linux! Då var det Slackware som gällde för mig. Fönstermiljön Xfree86 gav mig en massa huvudrymmer men det var en bra skola. Idag 15 år senare känner jag igen symptomen och vet vilka loggar det är meningsfullt att titta i.

Men det tar tid! Här gör den som inte körde Linux tidigt? Denna gång tog det bara några timmar att åter skapa det jag gjorde sist. Är du som jag så har du inte skrivit ner allt på papper. Jodå en del finns där, små lilla skrivna noter i min "svarta bok" som innehåller konfigurationen för hemmets datorer. Dock inte allt. Det saknas detaljer. Sådant som, när jag fixat till något bra, jag bara glömt att skriva ner.

Linux är idag nästan komplett som professionell och privat datormiljö. Jag ångrar inte att



jag för några år sedan tog det slungtriga steget till Linux som huvudsystem från att ha min Linuxmiljö som laboratoriemiljö vid sidan av min riktiga datormiljö som då var Windows.

Jag hittar ständigt nya program och funktioner, senast var det en serie grafikprogram, som kan göra livet lättare för mig. Det är lätt att vara Linux-användare.

Men det tar tid!

Min förhoppning är att, när nu den stabila utgåvan av Debian 5.0 alias "Lenny" släpptes den 14 februari, Debian ska visa vägen för vad vi användare behöver, nämligen stabilitet och frihet från oönskade överraskningar. Jag kör fortfarande Ubuntu. Att migrera till Debian 5.0 får bli ett laboratorieprojekt. Debian har ett rykte att vara mindre användarvänligt än Ubuntu - ett rykte som bara delvis är sanning. Vad som dock är skrivet i sten är att det trots allt tar tid att migrera sitt huvudsystem. Sedan må det vara hur det vill med Linuxbaserad eller Microsoftbaserad programvara. Det jag ämnar i min Linuxmiljö är de upprepade omstarterna som krävs i en Windows-miljö. Ett gissel som jag tror är ett önskat arv från DOS-miljön på 80-talet.

Passa på att ta en titt på den senaste fönstermiljön KDE 4.2 som är full av ögongodis. Glöm inte att allt grafiskt "lull-lull" kostar CPU-cykler och effekt. Den miljömedvetne väljer nog Xfce som skärmgränssnitt. En nerbantad fönstermiljö som passar bra i inbyggda system och gamla bärbara datorer från 90-talet.

Jan Zettergren

Här du en nyhet eller synpunkter för Embedded Linuxspalten, skriv till linux@etn.se



böva omvandlas säkert mellan applikationer, subsystem eller nätverk. Detta kallas informations säkerhet (InfoSec) och kan behövas förutom ComSec.

Låt oss beakta ett hypotetiskt scenario involverande en UAV som har ett topphemligt späningsuppdrag (fig 3). UAV-uppdrags-system kan innehålla hemlig eller topphemlig data, inkluderande planer för flyguppdrag, och det finns risk för att dessa kan avslöjas hemlig, eller topphemligstämplad information till det icke-hemligstämplade civila flyglednings-(ATC)-systemet. Detta är ett verkligt bekymmer eftersom ComSec-implementeringen inte bekräftar sig om den typ av data som utbyts mellan nätverk, endast huruvida nätverk och applikationer kan kommunicera.

UAVn skulle kunna lyfta in om kontrollerat luftburn och tvingas samverka med civil flygledning över okrypterade linjer, fungerande "i svart." UAVns flygplan skulle sedan begära en vektor ur kontrollens luftburn från flygledningen för att utföra sitt späningsuppdrag och därefter skulle all flyginformation vara topphemlig eller "i rött," inkluderande all kommunikation som kommer att vara över kraftigt krypterade linjer.

När späningsuppdraget fullbordats skulle UAVn kunna återgå till kontrollerat luftburn och återigen samverka med civil flygledning. Övergången tillbaka från rött till svart är stenkilt utmanande eftersom den involverar system innehållande topphemlig data kommunicerande över okrypterade linjer till en icke-hemligstämplad flygledning. Planerna för flygningen och loggad data för uppdragets röda del måste behållas för analys/rapportering men får inte "läcka ut" på tillbakavägen till basen.

UAVns uppdragssystem utnyttjar troligen IMA-plattformar på grund av begränsningar för utrymme, vikt, kraft och värme, såsom beskrivits tidigare. I denna miljö är det troligt att det inte är praktiskt att ha separata system för att ge fysisk isolering mellan svart och rött data. Detta bot måste istället åtgärdas genom en InfoSec-implementering som är flernivåskikt (multi level secure - MLS).

FLERA OBEROENDE SÄKERHETSIVÅR

På senare år har samkriteriet (Common Criteria - ISO-15048)7 accepterats brett, till följd av samverkan mellan pionjärarbete inom informations säkerhet under 1990-talet i USA, Kanada, Storbritannien, Frankrike, Tyskland och Nederländerna. Denna standard definierar de kriterier och garantier som krävs för olika typer av system och bot, inkluderande MLS-system innehållande information på flera olika säkerhetsnivåer. I det scenario som beskrevs tidigare, kommer UAVn

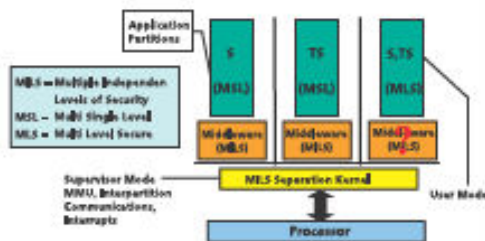


Fig 4. MLS-arkitektur

när den återvänder från sitt späningsuppdrag att ha bibehållit sin topphemliga plan för flygningen och uppdragsloggen. Den kommer att kommunicera över okrypterade linjer till en icke-hemligstämplad flygledning, vilket resulterar i att icke-hemligstämplad och topphemlig data finns i samma system med behov av utvärderingsgarantinivå (evaluation assurance level - EAL) 7.

MLS-arkitekturens utveckling för att klara säkerhetscertifieringens överkomliga kostnad och arbetsbörda som tidigare metodiska implementeringar medfört. Den gör detta genom att dramatiskt minska mängden säkerhetskritisk kod och drastiskt öka översynen av den. MLS-arkitekturen (fig 4) består av en MLS separationskärna som kan ha flera applikationer av olika säkerhetsklass i separata delar.

I MLS-arkitekturen ansvarar separationskärnan för att verkställa separationen av applikationer i individuella delar och att tillse att endast tillåtna flöden sker mellan applikationer. I denna arkitektur definieras tillåtna dataflöden i en säkerhetsdatabas som används av en referensövervakare för att verkställa metoder och detektera försök att kränka dessa. En implementering av MLS-arkitekturen bör också tillse att ingen tyst kommunikation kan ske via hemliga kanaler. Den temporala och spatiala partitioneringen förhindrar kommunikation över hemliga kanaler som skulle kunna ske genom variationer i timing, och resurstillgänglighet, och ytterligare tekniker bör användas för att förebygga hemliga kommunikationskanaler genom nyttjande av processors cache. Dessa möjligheter skulle trygga att UAVn i vårt exempel håller topphemlig uppdragsdata säkert isolerad från en icke-hemligstämplad applikation som kommunicerar med den civila flygledningen.

FRAMTIDEN

Användningen av obemannade system, både UAVer och UCACer, förutsätter att öka snabbt och de kunskaper och erfarenheter som samlas genom tillämpningen av dessa utnyttjas i alltmer utmanande uppdrag. Det är troligt att obemannade system kommer att utnyttjas i framtiden för mycket känsliga späningsuppdrag och

kommer att innehålla hemligstämplad data. Dessa system skulle kunna ha behov av informations säkerhet med ett högt EAL som läggs senare under utvecklingscykeln. Detta skulle vara svårt att lägga ovanpå en del existerande mjukvaruarkitekturen, men giver likheterna mellan respektive arkitekturen båd överföringen från ARINC 653 till MLS erbjuda låg risk. ■ ■ ■

Paul Parkinson, systemarkitekt, Wind River

Referenser

1. Capt. Brian P. Tico, U.S. Air Force, "Unmanned Air Vehicles: The Force Multiplier of the 1990s," Air Power Journal, Spring 1991, <http://www.airpower.maxwell.af.mil/airchronicle/ajpajp91/ajp91v4qtr91.htm>.
2. "ARINC Specification 653P1-2," http://www.arinc.com/chronicle/catalog_detail.cfm?item_id=632.
3. "Software Considerations in Airborne Systems and Equipment Certification," DO-178B, RTCA, December 1992, <http://www.rtc.org/onlinelibrary/product.cfm?id=341>.
4. "Software Considerations in Airborne Systems and Equipment Certification," ED-128, EUROCAE.
5. "Integrated Modular Avionics (IMA) Development: Guidance and Certification Considerations," DO-297, RTCA, <http://www.rtc.org/onlinelibrary/product.cfm?id=382>.
6. "Integrated Modular Avionics (IMA) Development: Guidance and Certification Considerations," ED-124, EUROCAE.
7. "Information Technology - Security Techniques - Evaluation Criteria for IT Security," ISO-15408, http://www.iso.org/iso/catalogue/catalogue_tc/catalogue_detail.htm?number=276.
8. M. Vanfleet, U.S. National Security Agency, "MLS Architecture," Open Group Security Forum, July 2002.

Om författaren: Eur Ing Paul Parkinson är systemarkitekt hos Wind River och arbetar med kunder inom flyg-, rymd- och försvarssektorn i Storbritannien och Norden. Hans professionella intressen inkluderar integrerade modulära flygelektronik (IMA) och ISTAR (Intelligence Surveillance Target Acquisition Reconnaissance)-system. Han bloggar om industrirelaterade frågor på Wind Rivers hemsida på <http://blogs.windriver.com/parkinson>.