Bezpeèná koexistencia softPLC a Linuxu (1)

Kombinova´ Linux a softPLC do jedného systému dáva zmysel. Linux ponúka mno´stvo vlastností, ktoré by pod¾a oèakávania mali podporova´ moderné PLC. Súèasné softPLC implementácie na báze Linuxu stavajú PLC nad jadro Linuxu, tak´e ich funkèná spo¾ahlivos´ závisí od presnosti jadra. Kvôli svojej ve¾kosti nemô´e by´ jadro Linuxu dostatoène validované a korektne osvedèené. Tento fakt bránil aplikovate¾nosti konceptu do bezpeènostných PLC systémov. Prístup popísaný v tomto èlánku kladie Linux a softPLC na rovnakú úroveò jeden ved¾a druhého na vrch malého mikrojadra tak, aby dva subsystémy mohli koexistova´ bezpeène bez toho, aby jeden od druhého závisel. V tomto smere je báza dôveryhodného kódu PLC redukovaná o nieko¾ko rádov a umo´ní sa tým jeho certifikácia pod¾a platných noriem pre bezpeènostné systémy.

Vývoj softPLC

Programovate¾né logické automaty (PLC) sú základom dnešnej priemyselnej automatizácie. Vynájdené v 60-tych rokoch minulého storoèia sa postupne vyvinuli z náhrady za relé obvody ku komplexným systémom, u ktorých sa dnes oèakávajú také vlastnosti ako grafické pou´ívate¾ské rozhranie a pripojenie na internet. Tieto funkcie sú ¾ahko dostupné v operaènom systéme Linux, myšlienka kombinácie Linuxu a PLC implementácie na báze softvéru (softPLC) je preto opodstatnená.

Ka´dopádne, PLC systémy sú èasto pou´ívané v bezpeènostných aplikáciách. Ich porucha by mohla spôsobi´znaèné škody na majetku alebo dokonca zrani´èi zabi´obsluhujúci personál. Tieto systémy preto musia by´ove¾a spo¾ahlivejšie ako priemerné pracovné stanice. Táto spo¾ahlivos´sa musí overi´pred inštaláciou prostredníctvom testovania alebo formálnej verifikácie. Zatia¾ èo štandardizované PLC programovacie jazyky sa hodia na vyèerpávajúce testovanie alebo validáciu, to isté sa nedá poveda´o platforme vykonávajúcej kód (softPLC). Dôkladné testovanie takejto platformy je nákladné a jeho finanèná nároènos´sa zvyšuje proporcionálne s mno´stvom kódu, ktorý je potrebné preskúma´. Báza dôveryhodného kódu softPLC obsahuje všetok kód, ktorý má potenciál ovplyvni´korektnú funkciu softPLC. Ak sa Linux pou´íva ako báza na podporu softPLC, mno´stvo kódu sa zväèší o pribli´ne jeden milión riadkov (rozsah porovnate¾ný s kódom jadra Linuxu), èím sa prakticky znemo´ní dôkladná validácia alebo verifikácia.

Tento èlánok sa sna´í predstavi´spôsob, ako umo´ni´bezpeènú a kooperatívnu koexistenciu softPLC a Linuxu na jednom zariadení. Na rozdiel od predchádzajúcich prístupov Linux nie je pou´ívaný ako báza pre PLC, Linux a PLC skôr existujú popri sebe na základe minimálneho mno´stva dôveryhodného kódu. Prístup popisovaný v tomto èlánku predchádza závislostiam medzi oboma komponentmi, ktoré by si inak vy´adovali, aby sa jeden na druhého spoliehal, ´e bude správne pracova´. Najskôr sa èlánok bude venova´nieko¾kým súèasným softPLC implementáciám na báze Linuxu a bude analyzova´ich závislosti. Potom sa bude prezentova´prístup na báze mikrojadra a na záver sa uká´u prvotné výsledky prebiehajúceho projektu, ktorý aplikuje prezentovaný prístup. Projekt sa týka integrácie Linuxu a známeho softPLC CoDeSys do jedného systému.

Súèasné softPLC na báze Linuxu

Z technického h¾adiska je softPLC program vykonávajúci špecializovaný kód, podobne ako napr. Java Virtual Machine (JVM), avšak na rozdiel od JVM, musí PLC vykonáva´svoj kód v èasovej postupnosti. Preto ka´dé softPLC potrebuje ma´pod sebou vrstvu operaèného systému, ktorý zabezpeèí vykonávanie v reálnom èase. Štandardný Linux to doká´e iba v obmedzenom rozsahu. Doterajšie softPLC na báze Linuxu teda pou´ívali doteraz buï jedno z mo´ných rozšírení reálneho èasu jadra alebo si vystaèili s obmedzeným výkonom v reálnom èase.

SoftPLC na báze rozšírenia reálneho èasu jadra Linuxu

Takéto rozšírenia jadra Linuxu ako napr. RTAI alebo RTLinux pracujú na základe integrácie špecifického programového rozhrania reálneho èasu do jadra Linuxu. Oddelenie tohto rozhrania od Linuxu je iba formálne a nepredpokladá sa, aby kód reálneho èasu vyvolal funkcie z jadra Linuxu. Neexistuje však pravidlo ako presadi´toto pravidlo. Z technickej stránky je ka´dá aplikácia reálneho èasu schopná vyvola´akúko¾vek funkciu jadra (èo väèšinou vedie k nedefinovanému správaniu). Všetky èinnosti reálneho èasu, v tomto prípade softPLC, sa vykonávajú ako privilegovaný kód v tom istom adresnom priestore ako aj jadro Linuxu. Linux aj softPLC majú nekontrolovaný prístup k tomu istému kódu, dátam a V/V portom – nie sú priestorovo oddelené. Plánovaè reálneho èasu spúš´a Linux ako proces s najni´šou prioritou. Linux má teda mo´nos´nieèo vykona´iba v okamihoch, keï ´iaden z procesov reálneho èasu nie je pripravený. To znamená, ´e pri prístupe k výpoètovému èasu je Linux vystavený na milos´od softPLC. Pokia¾ sa softPLC nikdy nezastaví, Linux nedostane príle´itos´na èinnos´. Linux tak doèasne závisí od softPLC (nie však opaène).

Zatia¾ èo jadro Linuxu doká´e ovplyvni´PLC, aplikaèné programy Linuxu to nevedia, pokia¾ nie sú vykonávané s právami administrátora. Za predpokladu vylúèenia akýchko¾vek bezpeènostných dier z jadra Linuxu, softPLC nepotrebuje dôverova´aplikáciám Linuxu. Báza dôveryhodného kódu preto obsahuje v tejto konfigurácii jadro Linuxu a softPLC.

SoftPLC v priestore pou´ívate¾a

Èisté nemodifikované jadro Linuxu u´obsahuje isté vlastnosti, ktoré poskytujú obmedzenú funkcionalitu reálneho èasu. Ak softPLC aplikácia mô´e akceptova´odchýlky èasovania v rozsahu nieko¾kých milisekúnd, mô´e by´implementovaná ako pou´ívate¾ský proces Linuxu. Taktie´niektoré spomenuté rozšírenia reálneho èasu Linuxu ponúkajú volite¾nú podporu, aby sa umo´nil prístup rozhrania reálneho èasu k programom v pou´ívate¾skom priestore (napr. LXRT). Toto by sa dalo pou´i´na implementáciu softPLC do pou´ívate¾ského priestoru bez toho, aby bolo potrebné urobi´ústupky v èinnosti reálneho èasu.

Oba prístupy vedú k tomu, ´e Linux a ani softPLC nezdie¾ajú ten istý adresný priestor a PLC nemô´e ¾ahko ohrozi´jadro Linuxu. Na druhej strane, PLC je závislé od Linuxu, preto´e od neho dostáva vo¾nú pamä´a V/V prostriedky pod¾a vlastných po´iadaviek. Z tohto vyplýva, ´e obaja nie sú stále priestorovo oddelení.

Ak PLC existuje ako štandardný pou´ívate¾ský proces Linuxu, potrebuje ma´pridelený výpoètový èas od jadra Linuxu. Na druhej strane, pokia¾ sa vyu´ívajú rozšírenia reálneho èasu v pou´ívate¾skom priestore, Linux dostane priestor na vykonávanie len v prípade, keï všetky procesy reálneho èasu sú blokované. Teda v prípade oboch metód neexistuje ´iadne doèasné oddelenie PLC a Linuxu. Buï je

softPLC v milosti jadra Linuxu alebo opaène. Znova, celkové mno´ stvo dôveryhodného kódu, èi´ e kódu potrebného na zriadenie funkcionality softPLC, zahàòa kód v jadre Linuxu a kód softPLC.

Prístup: oddelenie zdrojov

V oboch prístupoch spomenutých doteraz jadro Linuxu riadi priestorové a s výnimkou LXRT aj doèasné zdroje systému. SoftPLC zásadne závisí od tohto jadra – musí predpoklada´ , ´ e je úplne bezchybné. Pri zvá´ ení absolútnej ve¾kosti jadra je málo pravdepodobné naplnenie tohto predpokladu a kontrola bázy dôveryhodného kódu pod¾a štandardov je nereálna. Preto takýto systém nemô´ e by´ pou´ itý v bezpeènostných aplikáciách.

Ka´ dopádne, poh¾ad na typickú softPLC implementáciu uká´ e, ´ e sa vy´ aduje podpora len malej èasti funkcionality jadra. Všetko, èo je potrebné je základné runtime prostredie, t.j. prístup k pamäti, vstupy/výstupy a procesný èas. Mnohé z pokrokových funkcií jadra Linuxu nie sú od PLC vôbec vy´ adované. Jednako, pokia¾ sú tieto funkcie implementované v jadre, zodpovedajúci kód musí by´ dôveryhodný.

V záujme redukcie dôveryhodného kódu tento prístup implementuje iba tie funkcie na úrovni jadra, ktoré sú technicky nerealizovate¾né iným spôsobom (t.j. vy´ adujú si inštrukcie s privilegovanými právami) alebo tie vy´ adujúce si zriadenie bezpeèného runtime prostredia pre programy na úrovni pou´ ívate¾a. Všetky ostatné funkcie, ku ktorých vykonávaniu v privilegovanom móde inklinujú všetky monolitické jadrá ako Linux, sa mô´ u realizova´ aj programami na úrovni pou´ ívate¾a. Jadro navrhnuté pod¾a týchto vzorov je zvyèajne oznaèené ako mikrojadro. Takéto mikrojadro poskytuje len základné mechanizmy, ktoré umo´ òujú rozdeli´ systémové doèasné a priestorové zdroje do individuálnych podskupín. Tieto podskupiny mo´ no pova´ ova´ za virtuálne stroje, z ktorých ka´ dý mô´ e hosti´ kompletný operaèný systém ako napr. Linux spolu s jeho všetkými aplikáciami.

Na rozdiel od RTAI a RTLinuxu, tento prístup v princípe umo´ òuje existova´ ¾ubovo¾nému poètu virtuálnych strojov na jednom zariadení, t.j. nezávisle mô´ e by´ v èinnosti na jednom zariadení viac ako jedno softPLC. Akýko¾vek kód vykonávajúci sa v rámci virtuálneho stroja musí be´ a´ v pou´ ívate¾skom móde. Preto vo väèšine prípadov potrebuje operaèný systém isté prispôsobenie. Aplikácie v pou´ ívate¾skom priestore však vôbec nemusia rozozna´ rozdiel. Pokia¾ má oklieštenú vlastnú mno´ inu zdrojov, nie je operaèný systém be´ iaci na virtuálnom stroji schopný obmedzi´ iný operaèný systém na inom virtuálnom stroji. Systém je rozdelený na oddiely, ktoré sú navzájom úplne nezávislé. Ak to aplikujeme na prípad softPLC v kombinácii s Linuxom, mô´ eme ka´ dému z nich prideli´ iný oddiel a zabezpeèi´ tak ich vzájomnú nezávislos´ – softPLC nepotrebuje dôverova´ Linuxu a naopak. Oba subsystémy koexistujú bok po boku (v protiklade s koncepciou jedného nad druhým). Obaja sa delia o bázu dôveryhodného kódu skladajúceho sa z mikrojadra (jediný softvérový komponent be´ iaci v privilegovanom móde) a z vrstvy systémového softvéru, softvérového modulu be´ iaceho v móde pou´ ívate¾a nad mikrojadrom. Druhý z nich implementuje zásady správy oddielov na báze mechanizmov poskytnutých mikrojadrom. V nasledujúcom odseku bude popísané delenie zdrojov detailnejšie.

Priestorové delenie

Na to, aby mohol be´ a´ hos´ ujúci operaèný systém v oddieli, musí mikrojadro poskytnú´ prístup k èasti systémového hardvéru, t.j. k pamäti, k pamä´ ovo alebo portovo namapovaným V/V registrom a k prerušeniam. Takisto musí hos´ ujúcemu operaènému systému poskytnú´ mechanizmy na pridelenie a odobratie prístupu k týmto zdrojom zo strany aplikácií be´ iacich nad mikrojadrom. Toto sa musí dosiahnu´ najflexibilnejšie ako to je len mo´ né, aby bolo mo´ né hosti´ všetky typy operaèných systémov.

Ak sa má zabezpeèi´ nezávislos´ medzi hos´ ujúcimi systémami, musia by´ zdroje pridelené prostredníctvom dôveryhodnej strany (jadro). Aby sa to podarilo, mikrojadro priradí ku ka´ dému oddielu mno´ inu virtuálnych adresných priestorov, ktoré vystupujú ako zásobníky pre zdroje. Pomocou týchto metód je mikrojadro schopné garantova´ , ´ e ´ iadny oddiel nebude zasahova´ do druhého.

Toto sa však týka len zdrojov v pou´ ívate¾skom priestore. Mikrojadro samo o sebe potrebuje pamä´ na správu tabuliek stránok, zásobníkov, radov vybavených procesov a procesov èakajúcich na vykonanie. Tieto zdroje musia by´ taktie´ oddelené, inak by mohol iný oddiel spôsobi´ tzv. útok odopretia slu´ by, èo vedie k ve¾kej spotrebe pamäte jadra, napr. namapovaním jednej stránky ku všetkým dostupným virtuálnym adresám v rámci adresného priestoru

Priestorové delenie pri popise nášho prístupu je vhodné pre štandard ARINC 653 – pridelenie zdrojov je vykonané staticky pod¾a pevnej konfigurácie. Štandard vy´ aduje najtvrdšie obmedzenia nastavenia systému. Konfigurácia zásobníkov zdrojov je implementovaná vrstvou systémového softvéru. Okrem vlastností delenia ponúka táto vrstva aj komunikaèné slu´ by ako napr. zdie¾anie pamäte a notifikaèné mechanizmy medzi oddielmi. Toto poskytuje bezpeènú komunikáciu medzi hos´ ujúcimi systémami naskrz celým oddielom.

Doèasné delenie

Väèšina mikrojadier nebola pôvodne navrhnutá s cie¾om prevádzky v reálnom èase. Preto mnohé z nich zabezpeèujú priestorové delenie zdrojov, ako je popísané vyššie. Avšak len niektoré disponujú potrebnými vlastnos´ ami umo´ òujúcimi deterministické pridelenie doèasných zdrojov (t.j. procesného èasu).

Cie¾om virtualizaèného prostredia je vytvori´ ka´ dému operaènému systému pracujúcemu v rámci oddielu ilúziu, ´ e má pre svoje vlastné potreby vyhradený konštantný proporcionálny podiel celkového procesného èasu. To by naznaèovalo, ´ e vykonávací èas pridelený k jednotlivým oddielom lineárne narastá. V praxi však mô´ u by´ procesory v rámci oddielov iba èasovo multiplexné, t.j. ka´ dý oddiel má èasový úsek, poèas ktorého je aktívny. Idealizovaný lineárny nárast vykonávacieho èasu je aproximovaný funkciou rampy.

Kvalita tejto aproximácie sa zlepšuje s nespojitos´ ou èasových úsekov, t.j. absolútne trvanie èasových úsekov je kratšie. Len´ e z toho vyplývajúce èasté prepínanie vedie k znaènému nárastu ré´ ie. SoftPLC je klasický príklad èasovo spúš´ aných systémov. Musí by´ vyvolávané periodicky v pevných èasových okamihoch. Pokia¾ tieto vyvolania nie sú synchronizované s prepnutiami medzi oddielmi, softPLC bude èeli´ nepredvídate¾ným zdr´ aniam. Tie mô´ u zasiahnu´ softPLC kdeko¾vek poèas jeho vykonávacieho cyklu, t.j. mô´ u oneskori´ jeho èinnos´ alebo jeho výpoèet a tým predå´ i´ koneèný termín na splnenie pracovnej úlohy.

Je zrejmé, ´ e by bolo vhodné, aby dochádzalo k prepnutiam medzi oddielmi èo najrýchlejšie, aby sa zdr´ ania udr´ ali èo najkratšie. Ako však u´ bolo spomenuté, vedie to k neúmernému nárastu ré´ ie. Iná, mo´ no lepšia cesta, je synchronizova´ prepínanie medzi oddielmi s cyklom softPLC. Virtuálny stroj hos´ ujúci softPLC musí by´ aktivovaný v èase, keï má by´ v èinnosti softPLC, ktoré musí získa´ to¾ko èasu, aby bolo schopné dokonèi´ svoju prácu. SoftPLC hos´ ované na virtuálnom stroji vie poskytnú´ rovnaké charakteristiky ako nevirtualizované softPLC.

Ïalšie pokraèovanie seriálu nájdete v nasledujúcich èíslach ATP Journalu

www.sysgo.com

-bb-