XAVFSIZ TARMOQ TRAFIGINI TAHLILLASH TIZIMI ARXITEKTURASI TADQIQI

Abdujapparova Mubarak Baltabaevna

Muhammad al-Xorazmiy nomidagi Toshkent Axborot Texnologiyalari Universiteti

“Telekommunikatsiya injiniringi” kafedrasining mudiri, PhD, dotsenti

Muradova Alevtina Aleksandrovna

Muhammad al-Xorazmiy nomidagi Toshkent Axborot Texnologiyalari Universiteti

“Telekommunikatsiya injiniringi” kafedrasining PhD, dotsenti

Shoysayev Ozodxo’ja Anvarxo’ja o’g’li

Muhammad al-Xorazmiy nomidagi Toshkent Axborot Texnologiyalari Universiteti

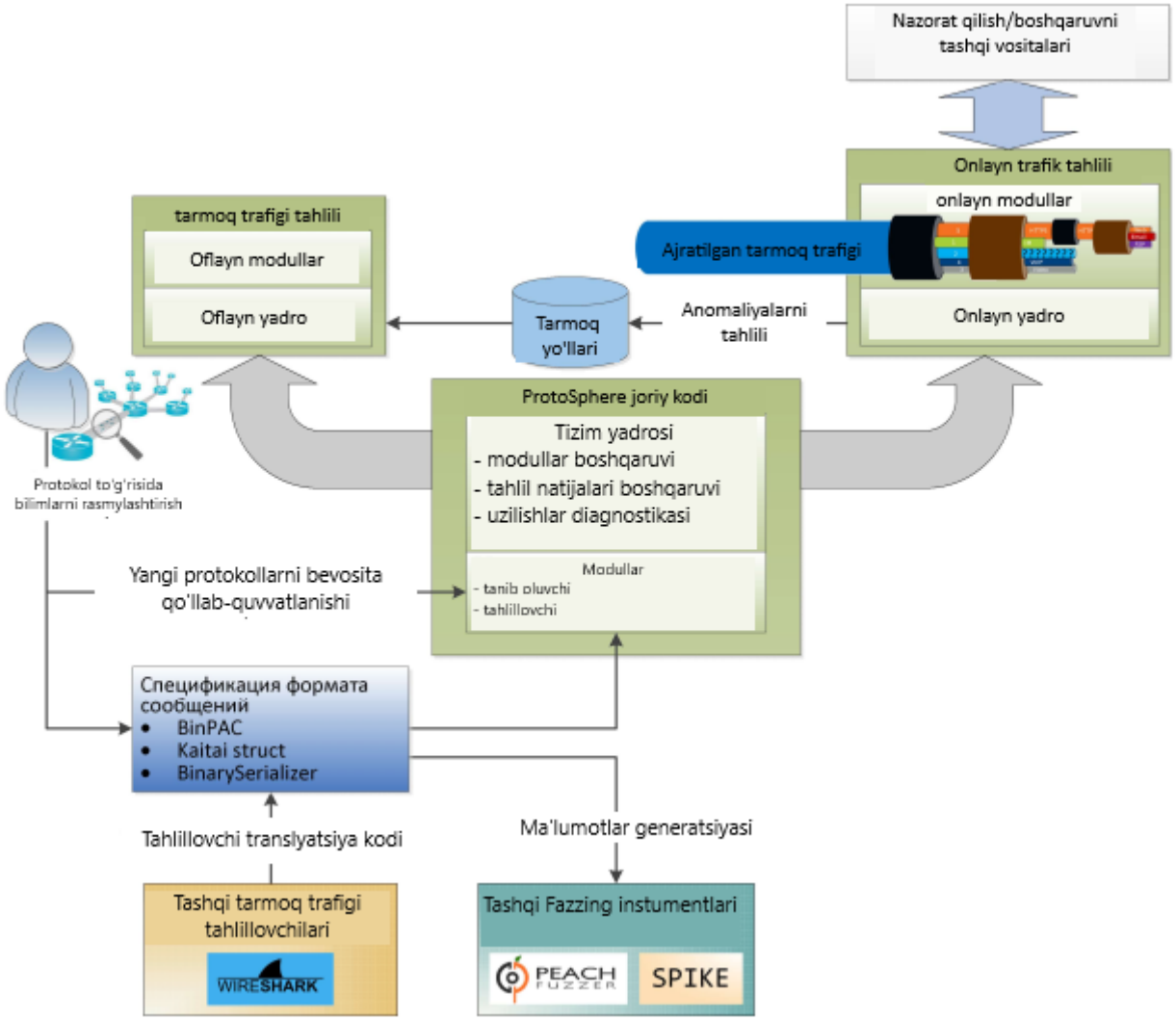
“Telekommunikatsiya injiniringi” mutaxasisligi magirtranti

(shoysayevozodxoja21@gmail.com)

Tizim arxitekturasi ishlab chiqilgan bo'lib, u onlayn va oflayn rejimda protokol tahlil qiluvchi manba kodlarining yagona to'plamidan foydalanish imkonini beradi. Taklif etilgan ma'lumotlarni taqdim etish modeli dinamik kutubxonaga kompilyatsiya qilingan tizim yadrosi sifatida amalga oshiriladi. Ma'lumotlarni tahlil qilish va tanib olish modullarining ishi yadro API-ga asoslangan. Ikkita vosita ishlab chiqildi - mos ravishda onlayn va oflayn tahlil qilish uchun – va ikkalasi ham bitta infratuzilmadan foydalanadi (1-rasm). O'rnatilgan parametrlarga qarab, yadro va tahlil qilish modullari oflayn analizator yoki onlayn analizator uchun yig'iladi. Ushbu printsip asosida qurilgan tizim tahlil qilish va aniqlash modullarini ishlab chiqishda (disk raskadrovka) va onlayn rejimda ishlashda ushbu modullardan keyingi foydalanishda (bir xil manba kodi ishlatiladi) oflayn tahlildan to'liq foydalanish imkonini beradi.

Yadroning oflayn va onlayn versiyalari o'rtasidagi asosiy farq xotirani boshqarish mexanizmidir: cheksiz ma'lumotlar oqimi ustida ishlashda xotirani o'z vaqtida chiqarishni ta'minlash kerak. Tarmoq izlarini tahlil qilish vositasi, o'z navbatida, ishlab chiqilgan grafik interfeysga muhtoj bo'lib, uning imkoniyatlari

muammolarning muayyan sinflarini hal qilishda foydalanuvchi harakatlarini avtomatlashtirishga qaratilgan, masalan: tarmoq ulanishlarini mahalliylashtirish va detallashtirish, protokollarni teskari muhandislik (shu jumladan disk raskadrovka). tahlilchilar), tahlil qilish tartibi paketlarini interaktiv nazorat qilish va boshqalar [1].



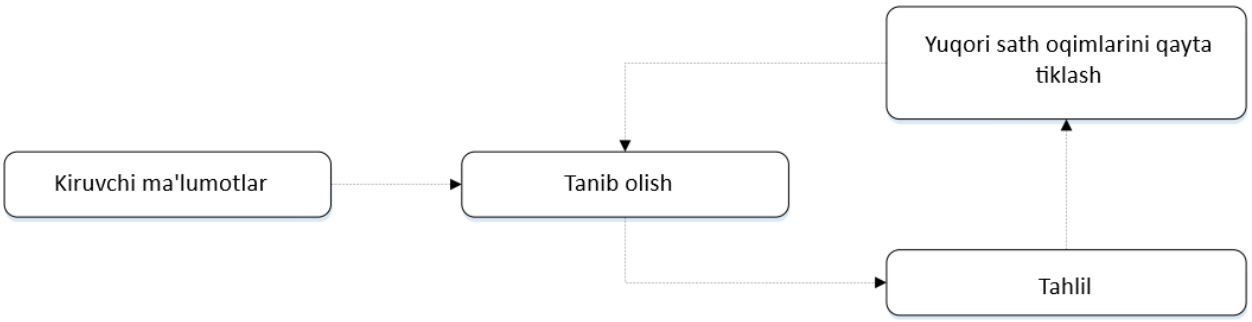
1-rasm. Tizimning tarkibiy qismlari.

Tizim ko'pgina tarmoq trafigi analizatorlari kabi modulli: har bir protokol uchun alohida modul yaratiladi, unda ushbu protokol bilan ishlash funksiyasi lokalizatsiya qilinadi. Asboblarning har biri o'ziga xos tahlil qilish modullaridan foydalanadi. Turli quyi tizimlar (asboblar) uchun tahlilchilarni kompilyatsiya qilishda bir xil manba kodidan foydalanish muhim ahamiyatga ega. Ishlab chiqilgan tizim doirasida tahlil qilish modullari mustaqil hisoblanadi: yangi

tahlillarni qo'shish mavjudlariga o'zgartirish kiritishni talab qilmaydi. Bu mustaqillik rezolyutsiyadan foydalanishga asoslangan tahlil bog'lash mexanizmi orqali erishiladi.

Qo'llab-quvvatlanadigan protokollar bazasi nafaqat tahlil kodini qo'lda ishlab chiqish, balki Wireshark tizimi tahlilchilarini avtomatlashtirilgan uzatish (portlash) orqali ham kengaytiriladi. Taklif etilayotgan tizim va Wireshark tizimining API funksiyalari uchun xaritalash belgilanadi, manba kodi oraliq ko'rinishga tarjima qilinadi, uning asosida uzatilgan kod yaratiladi. Port vositasi quyidagi bo'limda batafsil tavsiflangan [2].

Tahlil qilish tartibi. Tahlil ma'lumotlarni aniqlash va tahlil qilish operatsiyalarini ketma-ket bajarishdan iborat. Ba'zi protokollar uchun semantikaga muvofiq oqimlar tiklanadi, ularning ma'lumotlari ham keyinchalik tahlil qilinadi. Shunday qilib, ma'lumotlarni taqdim etish darajasining izchil o'sishi ta'minlanadi (2-rasm). Natijada, foydalanuvchi yuqori darajadagi ma'lumotlarda ishlaydi, bu esa uning tahlil sifatini oshirishi mumkin.



2-rasm. Ma'lumotlarni taqdim etish darajasini doimiy ravishda oshirish.

Yadro darajasida tahlil holati uchta stek bilan tavsiflanadi:

- bloklar to'plami;
- kontekstlar to'plami (element - kontekst juftligi va kalitlar guruhi);
- tarmoq tugunlari to'plami (element - jo'natuvchi va qabul qiluvchi juftligi).

3-rasmda blokni tizim yadrosi tomonidan tahlil qilishning soddalashtirilgan algoritmi ko'rsatilgan.

*1. Bloklar stekiga **B** blokni qo'shish*

*2. **B** uchun **T** tahlil qilish turini aniqlash*

*3. Agar **T** uchun **ContextExistence** bayrog'i o'rnatilgan bo'lsa, unda,*

*4. Kontekst stekiga **C** kontekst qo'shish*

*5. Agar **T** uchun **L** log-mavzusi nazarda tutilgan bo'lsa, unda,*

*6. Mavzular jurnali stekiga **L** mavzu qo'shish*

*7. **T** ga muvofiq **B** ma'lumotlar uchun tahlilchini chaqirish*

*8. Agar **T** uchun **L** log-mavzusi nazarda tutilgan bo'lsa, unda,*

9. Jurnal mavzulari stekidan mavzuni ko'chirish

*10. Agar **T** uchun **ContextExistence** bayrog'i o'rnatilgan bo'lsa, unda*

11. Kontekst stekidan kontekstni ko'chirish

12. Blokni bloklar stekidan chiqarish

3-rasm. Tizim yadrosi tomonidan blokni tahlil qilish algoritmi.

Kontekstni qo'shish (4-qatorda) asosiy kalitlar guruhida T turdagi kontekst mavjud: agar bo'lmasa, u yaratilishi kerak. Blok stekidan ajratish daraxtini qurish uchun foydalaniladi: stekning yuqori qismi tahlil qiluvchi tomonidan yaratilgan bloklarning ota-onasiga aylanadi.

Tanib olish jarayoni birinchi muvaffaqiyatga qadar ketma-ket qo'llaniladi. Qaysi blok tahlil qilinayotganiga qarab, tan oluvchilarning turli guruhlari qo'llaniladi [3]:

1) Oqim-blok:

- yig'ish turi ma'lum bo'lgan oqimni tanib oluvchilar;
- universal tanib oluvchilar;

2) Tarkibli yoki yakka blok:

- maydonni taniy oluvchilar;
- universal tanib oluvchilar.

Kontekst stek bloklarni guruhlash uchun ishlatiladi (kontekst daraxti qurilgan): stekning cho'qqisi hozirda sarlavhasi tahlil qilinayotgan protokolni tavsiflaydi. Kontekstni va stekning cho'qqisidagi kalit guruhini faol deb hisoblanadi. Kontekst blokni tahlil qilishdan oldin darhol stekga suriladi, agar tahlil turi kontekstga ega bo'lsa (bayroq o'rnatilgan)ContextExistence) va tahlil

qilish tugagach, stekdan chiqariladi. Oldin Tahlilning boshida turi aniqlanmagan xayoliy kontekst va bo'sh kalitli kalit guruhi yaratiladi va stekga joylashtiriladi: buning yordamida faol kontekst va kalit guruh har bir vaqtning o'zida aniqlanadi. Soxta kalit guruhi birinchi haqiqiy kontekstning asosiga aylanadi. Stakka kontekst qo'shishdan oldin, faol kalitlar guruhida tegishli turdagi voris konteksti mavjudligi tekshiriladi. Agar shunday bo'lsa, unda yangi kontekst yaratilmaydi va mavjud kontekst stekga suriladi. Kontekst yaratilganda kengaytma – ulanish holatini tavsiflovchi maxsus atributlar ishga tushiriladi. SSL protokoli kontekst kengaytmasida, xususan, keyingi foydalanish uchun foydalanuvchi tomonidan tizimga yuklangan shifrlash kalitini saqlanadi.

Agar blokni tahlil qilishdan oldin kontekst stekga surilsa, tahlil qilish jarayonida kalitlar guruhi aniqlanadi. Istisno nol o'lchamdagi kalit bilan bog'liq GroupKeySize: kontekstda tuzilishi mumkin faqat bitta guruh (xususan, HTTP protokoli uchun kalit o'lchami nolga teng).

NetworkTree yaratish uchun tarmoq tugunlari to'plami ishlatiladi. Agar protokol jo'natuvchi va qabul qiluvchining mavjudligini nazarda tutsa, sarlavha tahlil qilinganda, kerakli tarmoq tugunlari stekga joylashtiriladi.

Jurnal tahlil qilish xatolarini qayd etish uchun mo'ljallangan. Undagi har bir yozuv vaqt tamg'asi, matnli xabar va tahlili ushbu yozuvni yaratishga olib kelgan blok bilan tavsiflanadi. Jurnaldagi yozuvlar mavzu bo'yicha ham guruhlangan: mavzuga ulanish bloklarni tahlil qilish turi yordamida amalga oshiriladi. Jurnal mavzular to'plamini saqlaydi: tahlil boshlanishidan oldin birlamchi mavzu unda joylashtiriladi. Agar blokni tahlil qilish turi mavzuni talab qilsa, u holda tegishli mavzu tahlil qilishdan oldin darhol stekga suriladi va tahlildan keyin stekdan chiqariladi. Agar tur uchun mavzu aniqlanmagan bo'lsa, tegishli yozuvlar stekning yuqori qismidagi mavzu bilan bog'lanadi. Odatda, har bir tahlil moduli uchun bitta mavzu belgilanadi. Barcha oqim bloklari tahlil qilingandan so'ng, trafik faylini tahlil qilish tugallanadi. Onlayn rejimda tahlil foydalanuvchining buyrug'i bilan to'xtaydi.

Yadro API. Tizim API-si tahlilchiga turlarni (tahlillarni tavsiflash) va

rezolyutsiyani ro'yxatdan o'tkazish, oqim bloklarini, bitta va birikma fragmentlarni yaratish va oqimni tiklashni boshqarish imkonini beradi. Xususan, qayta tartiblangan paketlardan oqimni to'g'ri tiklash uchun ikkita oqim blokini bittaga birlashtirish funksiyasi taqdim etiladi [4].

*Tahlil qilish. 1-j*advalda tahlillarni ishlab chiqishda qo'llaniladigan asosiy funksiyalarni ko'rsatadi. E'tibor bering, tahlil qilish modullari bir-biridan butunlay mustaqildir.

1-jadval. Tahlil qilish modullarini ishlab chiqish uchun API.

Funksiya nomi	**Tavsif**
Blok operatsiyalari	
processSingle	Bitta blokni yaratish va tahlil qilish
processComposite	Composit-blokni yaratish va tahlil qilish
createStream	Oqim-blokni yaratish
completeStream	Oqim-blokni tahlil qilish
streamAppend	Oqim-blok uchun blok ma'lumotlarini qo'shish
Holat boshqaruvini tahlil qilish	
contextExtension	Faol kontekst kengaytmasi ma'lumotlarini olish
activateKeyGroup	Joriy kontekstda berilgan kalit bilan belgilangan guruhni faollashtirish
keyGroupExtension	Faol kalit guruhi kengaytmasi ma’lumotlarini olish
setSrcDst	Yuboruvchi va qabul qiluvchini faollashtirish
Tahlil qiluvchilarni ro'yxatga olish	
regType	Ro'yxatdan o'tish turi
regRecognizer	Tanib olishni ro'yxatdan o'tkazish
getType	Boshqa modulda ro'yxatdan o'tgan turni olish
Xabarlar jurnali	
log	Jurnal xabarini yozish

Guruhni shakllantirish kaliti sarlavhani tahlil qilishda tuziladi:

activateKeyGroup funksiyasi guruhni faollashtirish uchun foydalaniladi. Tahlil qilish jarayonida guruhlar (keyGroupExtension) va faol kontekst (contextExtension) kengaytmalari ma'lumotlari o'zgartiriladi.

Maydon rezolyutsiyasini ro'yxatdan o'tkazishda tegishli maydon nomini, shuningdek, asos blokni tahlil qilish turini ko'rsatish kerak. Shunday qilib, turli modullarda joylashgan tanib oluvchilar bir xil maydonga murojaat qilishlari mumkin. Ushbu bog'lash usuli tufayli mavjud tahlil modullarining kodiga o'zgartirish kiritmasdan funksionallikni kengaytirish mumkin bo'ladi.

Trafik tahlili natijalarini berilgan formatda taqdim etish. Qurilish interfeysi (2-jadval) dekodlangan ma'lumotlarni foydalanuvchi tomonidan belgilangan formatda saqlash uchun mo'ljallangan. Format qurish modulini yaratish orqali aniqlanadi. Odatda, bunday modullar onlayn tahlilni o'tkazishda qo'llaniladi. Xususan, tiklangan PNG, JPEG, HTML fayllari, HTTP oqimlari va boshqalarni saqlash mumkin. Qurilish modullarini ulashning ikki yo'li mavjud [5]:

- bloklarni tahlil qilish turi bo'yicha;
- tarmoq tugunlari steki bo'yicha.

Ro'yxatdan o'tishda qurilish moduli tizim yadrosiga unga o'tishi kerak bo'lgan bloklar turini aytadi. Tarmoq tugunlarini stek bilan bog'lashda blok turi bilan bir qatorda jo'natuvchilar va qabul qiluvchilar stekining tarkibi tekshiriladi: faqat mos keladigan bo'lsa, blok tegishli modulga yuboriladi.

2-jadval. Qurilish modullarini ishlab chiqish uchun API.

Funksiya nomi	**Tavsif**
createBuffer	Bufer yaratish
completeBuffer	Buferni faylga saqlash
bufferAppend	Belgilangan formatga muvofiq buferga blok ma'lumotlarini qo'shish

Wireshark-dan tahlillarni ko'chirish uchun vosita. Qo'llab-quvvatlanadigan tizim protokollari bazasini ko'paytirish uchun Wireshark analizator tahlilchilarini portlash uchun vosita ishlab chiqildi. Tahlil modulini ko'chirish jarayoni uch

bosqichdan iborat:

1. tahlil manba kodini oraliq ko’rinishga tarjima qilish;

2. oraliq vakillikning modifikatsiyasi;

3. o'zgartirilgan oraliq vakillik asosida kod generatsiyasi.

Wireshark tahlil modullari uchun kod C tilida yozilgan. Ta'riflangan tizim tomonidan ishlatiladigan tahlillarni ishlab chiqish C++ tili doirasida amalga oshiriladi. C va C++ tillari ko'p jihatdan o'xshashdir. Instrumentni tanlashga asos bo’lib, uni ishlatishning qulayligi, shuningdek, C++ tilida dasturiy interfeysning mavjudligi bilan bog'liq.

Birinchi bosqich - Wireshark tahlil modulining manba kodidan foydalangan holda abstrakt sintaktik daraxt (ASD) yaratish. Ikkinchi bosqichda qurilgan ASD ni modifikatsiya qilish amalga oshiriladi [6]:

1. foydalanilgan turlar va tur nomlarini, agar ular boshqacha bo'lsa, almashtirish;

2. ASD ning modul kodiga mos keladigan qismini ajratish (qo'shilgan fayllarga mos keladigan ASD qismini ajratish);

3. keyingi bosqichlar uchun zarur bo'lgan ma'lumotlarni to'plash;

4. bir interfeysning funksiya chaqiruvlarini boshqa interfeysning funksiya chaqiruvlari bilan almashtirish, o‘zgaruvchilar va funksiya parametrlarini almashtirish;

5. keraksiz ko'rsatmalarni olib tashlash, amalga oshirilmagan konvertatsiyalarni tekshirish.

Uchinchi bosqich ASD yordamida maqsadli kodni yaratishdan iborat. Shu bilan birga, kodning ba'zi bo'limlari (xususan, maqsadli tizimga xos funksiyalar) to'g'ridan-to'g'ri, ASD ga qo'shilmasdan yaratilishi mumkin.

Tarjima qilishdan oldin, turdagi ma'lumotlarni saqlash uchun modul kodi oldindan qayta ishlanadi (parsing). Wireshark kutubxonasining kiritilgan sarlavha fayllari va modulning o'zi o'rtasidagi chegara ASD darajasida kiritilgan fayllar kodini modul kodidan yanada ajratish uchun maxsus pragma direktivasi bilan belgilanadi. Keyin modul kodi standart tizim protsessori tomonidan qayta

ishlanadi, undan so'ng Elsa vositasi oldindan ishlangan kodni ASD ga tarjima qilish uchun ishlatiladi.

Wireshark tahlil maydonlarining turlari haqida ma'lumot to'plash uchun ASD-da hf_register_info turidagi tuzilmalar massivining ishga tushirish ro'yxati ishlatiladi.

Jurnalga kiritilgan protokollar haqida ma'lumot olish uchun funksiyalar tahlil qilinadi proto_register_protoname va proto_reg_handoff_protoname, bu yerda protoname protokol nomi. Quyidagi funksiyalarga qo'ng'iroqlar mahalliylashtirilgan:

1. proto_register_protocol – protokolni ro'yxatga olish;
2. create_dissector_handle – tahlil tavsifini yaratish;
3. register_dissector – tahlilga nom berish;
4. find_dissector – nomi bilan tavsiflovchini qaytarish;
5. dissector_add_uint – tahlilni bog'lash jadvalida bog'lash.

Keyinchalik, Wireshark tahlil yadrosi funksiyalariga qo'ng'iroqlar tahlilni amalga oshiradigan funksiyalarda ishlab chiqilgan tizimning tahlil yadrosi funksiyalariga qo'ng'iroqlar bilan almashtiriladi.

Wireshark tahlilchilari uchta yoki to'rtta parametrni oladi:

- bufer ob'ektiga ko'rsatgich;

- paketning sarlavhalarida avval ajratilgan xizmat maydonlari to'plamining qiymatlarini o'z ichiga olgan packet_info tuzilishiga ko'rsatgich;

- ma'lumotlarni qo'shadigan tahlil qilish daraxtining pastki daraxtiga ko'rsatgich;

- Ixtiyoriy foydalanuvchi parametri: uning qiymati chaqiruvchi va chaqiruvchi tahlilchilar o'rtasidagi kelishuv bilan belgilanadi.

Ishlab chiqilgan tizimning tahlil funksiyasi 2 parametrni oladi: xotira buferiga ko'rsatgich va buferning o'lchami.

O'zgartiriladigan funksiyalarni guruhlarga bo'lish mumkin:

1. ma'lumotlarga kirish funksiyalari;
2. tahlil daraxtiga maydonlar qo'shish funksiyalari;

3. TCP segmentlari orqali uzatiladigan ma'lumotlarni tahlil qilish funksiyalari.

Ma'lumotlarga kirish funksiyalari bayt tartibini o'zgartirish funksiyalari yordamida to'g'ridan-to'g'ri ma'lumotlarga kirishga aylantiriladi.

Tahlil daraxtiga maydonlar qo'shish funksiyalari mos tahlil turiga ega oddiy yoki murakkab fragment bloklarini qayta ishlash funksiyalari bilan almashtiriladi.

Tcp_dissect_pdus funksiyasi TCP oqimini chegaralari segment chegaralariga to'g'ri kelmasligi mumkin bo'lgan xabarlarga bo'lish uchun mo'ljallangan. Rivojlangan tizimda TCP oqimlarini yig'ish TCP moduli tomonidan amalga oshirilganligi sababli, yuqoriroq protokollarning paketli tahlillarida faqat oqim ma'lumotlarini xabarlarga bo'linishini amalga oshirish kerak. Shuning uchun, tcp_dissect_pdus funksiyasi chaqiruvi ushbu tahlilni bajaradigan ko'rsatmalar ketma-ketligi bilan almashtiriladi. Yangi manba kodi Elsa vositasi yordamida ASD dan yaratiladi.

FOYDALANILGAN ADABIYOTLAR RO’YHATI

[1] Mike Cloppert. An Overview Of Protocol Reverse-Engineering. https://digitalforensics.sans.org/blog/2012/07/03/an-overview-of-protocol-reverse-engineering

[2] IETF RFC 791. J. Postel. Internet Protocol, September 1981

[3] Antonios Atlasis. Fragmentation (Overlapping) Attacks One Year Later, Troopers 13 – IPv6 Security Summit, 2013

[4] IETF RFC 793. J. Postel. Transmission Control Protocol, September 1981

[5] Judy Novak, Steve Sturges. Target-Based TCP Stream Reassembly, 2007

[6] Jon C. R. Bennett, Craig Partridge, Nicholas Shectman. Packet reordering is not pathological network behavior // IEEE/ACM Transactions on Networking (TON) archive, Volume 7 Issue 6, Dec. 1999, Pages 789-798