Realization Router, Crypto, Real Time Clock Applications on NetFPGA Open Source Hardware Platform With VHDL (Very High-level Design Language) Codes

Kursat Gol
Electric-Electronic Engineering, Kahramanmaras Sutcu Imam University
Bati Cevre Yolu 46040 K.Maras, Turkey
E-mail: yl_kgol@ksu.edu.tr

Metin Salihmuhsin
Electric-Electronic Engineering, Kahramanmaras Sutcu Imam University
Bati Cevre Yolu 46040 K.Maras, Turkey
E-mail: msalihmuhsin@ksu.edu.tr

Abstract

In this paper, we aim to define right sniffing method to logging on database user character on local-global PC networks. We tested two method like PCAP and TCP socket programming, claim that TCP socket programming one is suitable than PCAP. We applied data authentication test with on board ATSHA204. Timestamp labels were tagged to logs on database via M41T62 real-time clock chip.

Keywords: NetFPGA-1G-CML, URL Address Sniffing, Crypto Authentication

NetFPGA Açık Kaynak Donanım Platformunda Yönlendirici, Şifreleme ve Gerçek Zaman Saati Uygulamalarının VHDL (Very High-level Design Language) Kodlarıyla Gerçekleştirilmesi

Özet

Bu çalışmamızda NetFPGA-1G-CML geliştirme kartı ile yerel-genel bilgisayar ağlarında kullanıcı karakteristiklerini belirleyip, veritabanı üzerinde kayıt altına alınması hedeflenmiştir. Bu işlemi gerçekleştirebilmek için iki metod test edilmiş ve TCP socket programming metodunun, PCAP metoduna göre üstünlükleri gösterilmiştir. Kart üzerindeki ATSHA204 şifreleme entegresi kullanılılarak doğrulama algoritması ile veri doğrulaması yapılmıştır. Kayıtlar M41T62 gerçek zaman saati entegresi ile zaman etiketi eklenerek tutulmuştur.

Anahtar Kelimeler: NetFPGA-1G-CML, URL Adres Yakalama, Veri Şifreleme

1. GİRİŞ

İletişimin her geçen gün önem kazandığı günümüzde, iletişim konusuyla ilgili firmaların, üniversitelerin yaptığı çalışmalara, araştırmalara ve testlere ilgi artmaktadır. Artan bu ilgiyi karşılayabilmek amacıyla Amerika'da Stanford Üniversitesi'nde tamamen açık kaynak kodlu olmak üzere NetFPGA (Networking Field Programmable Gate Array) platformu geliştirilmiştir. Başta FPGA entegre üreticisi Xilinx olmak

üzere, ağ cihazları üreticisi Cisco, Stanford Üniversitesi ve Cambridge Üniversitesi de bu projeye destek vermişlerdir.

Bizim yaptığımız bu çalışmada NetFPGA platformunun üçüncü nesil üyesi olarak üretilen NetFPGA-1G-CML (Networking Field Programmable Gate Array-1 Gigabit-Computer Measurement Laboratory) geliştirme kartı kullanılmıştır. Ayrıca kartın üzerinde bulunan ATSHA204 ve M41T62 entegreleriyle uygulama yapılarak kartın özelliklerinin tanıtılması hedeflenmiştir. Aynı zamanda ofis, ev veya okul gibi yerel bilgisayar ağlarında kablo üzerinden bağlantı sağlayan kullanıcıların giriş yaptıkları internet adresleri gerçek zamanlı olarak listelenmiştir. Kullanıcıların kimlikleri internete giriş yaptıkları bilgisayarlarının IP (Internet Protokol) adresleri yardımıyla tanımlanmıştır. Çalışmamızda bu verilerin tutulması ve işlenebilmesi için Fedora işletim sistemi üzerinde çalışan MySQL (My Structural Query Language) tabanlı MariaDB (Maria DataBase) veritabanı uygulaması kullanılmıştır. Elde edilen veriler görsel olarak hazırlanan GUI (Graphical User Interface) ile ağ yöneticisine sunulmuştur. GUI hazırlanırken linux tabanlı Fedora işletim sistemi üzerinde GTK (GIMP (GNU (Gnu's Not Unix) Image Manipulation Program) Tool Kit) kütüphanesi tercih edilmiştir. Çalışmamızın sonunda, NetFPGA-1G-CML kartının kontrolü için PCI (Peripheral Component Interconnect) veriyolunu kullandığımız kontrol bilgisayarının performans değerleri, dünya genelindeki araştırmalarda serbest olarak kullanılabilen, ağ yönetim aracı Cacti ile izlenmiştir.

Naous, Gibb, Bolouki ve McKeown yapmış oldukları çalışmada birinci nesil olarak üretilen NetFPGA-1G kartı üzerinde akademik ve eğitim amaçlı uygulamalar gerçekleştirmişlerdir. Bahsi geçen çalışmalarında Verilog programlama diliyle, kart üzerindeki Xilinx Virtex-II Pro entegresini donanımsal olarak programlayarak yeniden kullanılabilen mantıksal bloklar tanımlamışlardır. İçerisinde 53136 adet mantıksal programlanabilen hücre bulunduran Xilinx Virtex-II Pro entegresi yardımıyla gerçekleştirilen bu çalışmada ağ cihazlarının en önemlilerinden biri olan yönlendirici uygulaması da bu çalışma kapsamında denenmiştir. TCAM (Ternary Content Addresable Memory) metoduyla hızlı bir şekilde IP yönlendirme tablosu üzerindeki rotalar taranarak en iyi rota seçimi yapılmıştır. Bu metod sonraki çalışmalarda çok faydalanılan bir yöntem olarak hala kullanılmaktadır. Kontrol kelimesinin genişliği 64-bit olan bu çalışmada PCI veriyolu üzerinden bağlı olan kontrol bilgisayarında çalıştırılan C tabanlı uygulamayla birlikte, OSPF (Open Shortest Path First) yönlendirme protokolü kullanılmıştır.

Lockwood, Watson, Hartke, ve McKeown başka bir çalışmada toplam 5 adet NetFPGA-1G kartıyla internet dünyasına bağlı yerel bir ağ kurup, verilog donanım programlama diliyle yönlendiriciler üzerinde bazı denemeler yapmışlardır. Bir web sunucusu, bir e-posta sunucusu 5 adet istemci bilgisayar ve 5 adet de yönlendiriciden oluşan bu ağ yapısında bir adet yönlendirici internet ağına çıkmak için kullanılmıştır. Çalışmalarının sonucunda 2007 yılında bu çalışmalarını Amerika'da yapılan ve dünya genelinden gelen akademisyenlere sunmuşlardır.

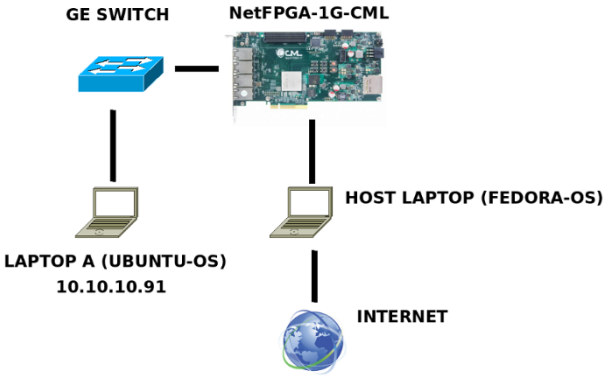
Bu çalışmamızdaki amacımız açık kaynak koduyla ağ cihazları üzerinde yapılabilecek uygulamaları gerçekleştirmek ve NetFPGA-1G-CML platformunun üzerindeki modülleri yazılımsal olarak Xilinx Kintex-7 FPGA entegresi ile kontrol etmektir. Çalışmada fiziksel FPGA entegresi üzerinde yazılımla bir işlemci oluşturulmuş, bu işlemci ile Microchip firmasının ürettiği PIC32MX795F512L entegresi haberleştirilmiştir. Şifreleme ve gerçek zaman entegreleri olan ATSHA204 ve M41T62 entegreleri fiziksel olarak PIC mikroişlemcisine bağlı olduğundan FPGA içerisinde oluşturulan Microblaze işlemcisinin PIC aracılığıyla şifreleme ve gerçek zaman entegrelerine ulaşması ve kontrol komutları göndermesi sağlanmıştır. Bizim çalışmamızda yerel bir ağ içerisindeki kablo ile bağlı olan kullanıcıların zaman ve kaynak IP adresi bazında listelenmesi, veritabanına kayıt edilerek ağ yöneticisine bir grafiksel arayüz olarak sunulacaktır.

2. MATERYAL

Çalışmamızda ana materyal olarak NetFPGA-1G-CML kartı kullanılmıştır. Bu kartın üzerinde çalışmamızla ilgili olan gerçek zaman entegresi, şifreleme entegresi ve ana işlemci görevi üstlenen, 326080 adet mantıksal hücreya sahip Xilinx Kintex-7 325T FPGA entegresi kullanılmıştır. Kintex-7 üzerinde reference_nic ismindeki rererans tasarım temel alınmıştır. Kontrol işlemleri Fedora işletim sistemi kurulu olan ve PCI veriyolu üzerinden NetFPGA kartına bağlanan bir bilgisayar ile yapılmıştır. Bu bilgisayarda yazılımsal mikroişlemci Microblaze için gömülü olarak çalışan C kodları derlenmiştir. Çalışmamızdaki yerel ağa bağlanan kullanıcıların zaman ve IP adresi bilgileri ise yine bu PC de çalışan MariaDB veritabanı ile toplanmıştır. Şekil 1'de toplu olarak çalışmamızda kullandığımız yerel ağ ve öğeleri görülmektedir.

Yerel ağdaki fiziksel kablo ile bağlı kullanıcı bilgisayarlarının internete çıkabilmesi için kontrol bilgisayarı ile kablosuz ağ arasında bağlantı kurulmuştur. NetFPGA-1G-CML arayüzünde yakalanan

paketler veritabanında işlenerek ağ yöneticisine sunulan görsel program sayesinde ekrana yansıtılmaktadır.



Şekil 1. Çalışmamızda kullandığımız yerel ağ ve tüm öğelerin bağlantı şeması

3. METOT

Yapmış olduğumuz bu çalışmada TCP/IP (Transmission Control Protocol/Internet Protocol) paketlerini yakalayıp işlemek için iki farklı metot kullanılmıştır. Her iki metotta da NetFPGA-1G-CML kartı üzerinde verilog donanım programlama dili ve kontrol bilgisayarında MariaDB veritabanı kullanılmıştır.

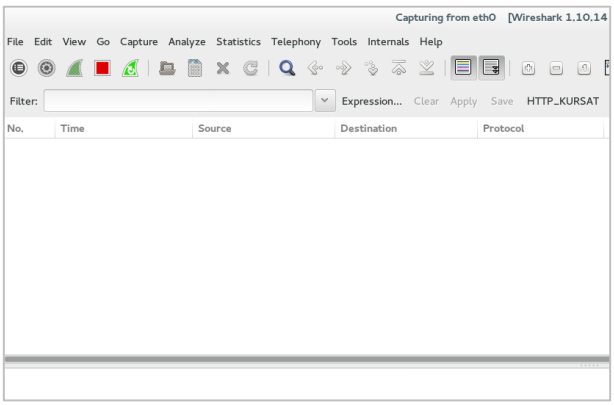
- NetFPGA-1G-CML kartı üzerinden akan TCP/IP paketleri PCAP (Packet Capturing) yöntemiyle yakalanmaya çalışıldı. Bu yöntemde HTTP-GET (Hyper Text Transfer Protocol - Get) paketlerinin yakalanamadığı görülmüştür.
- İkinci metotta paketler TCP Socket Programming (TCP Soket Programlama) yöntemiyle yakalandı ve ayrıntılı irdelenmek üzere karakter analizi yapılarak URL (User Resource Link) adresleri ayrıştırıldı.

3.1. Packet Capturing Yöntemiyle TCP/IP Paketlerinin Yakalanması

Bu metotla yaptığımız çalışmayı gerçekleştirirken aşağıdaki fonksiyonu kullanarak kontrol bilgisayarında C kodu derlenmiştir.

```
int pcap_compile(pcap_t *p, struct bpf_program *fp, char *str, int optimize, bpf_u_int32 netmask)
```

Fakat derlenen bu program çalıştırıldığında Şekil 2'de görüldüğü gibi NetFPGA-1G-CML arayüzünden TCP/IP paketi yakalanamadığı görülmüştür. Yakalanamamasının sebebinin, packet capturing yönetimiyle sadece gelen paketlerin yakalanabildiği, giden paketlerin yakalanamadığı anlaşılmaktadır. Serbest şekilde indirilip her bilgisayarda kullanılabilen Wireshark programında packet capturing yöntemiyle hiç veri alınamadığı açıkça görülmektedir.



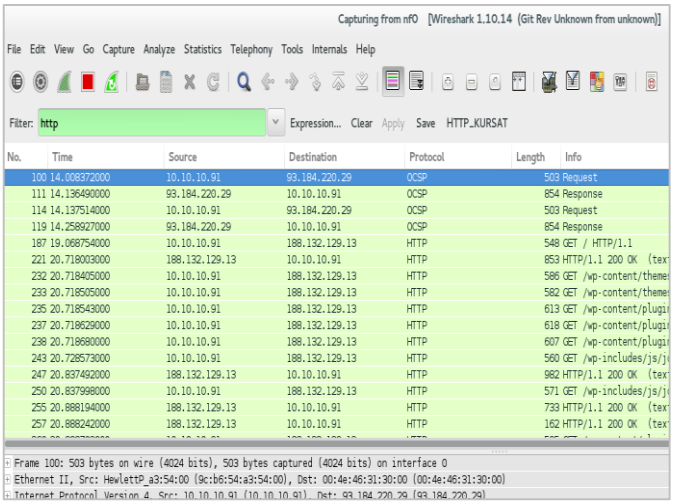
Şekil 2. PCAP yöntemiyle hiçbir paketin yakalanamadığı görülmektedir

3.2. **TCP Socket Programming Yöntemiyle TCP/IP Paketlerinin Yakalanması**

Çalışmamızda kullandığımız TCP socket programming metoduyla paketleri yakalamak için aşağıdaki fonksiyon kullanılmıştır.

int socket(int domain, int type, int protocol);

C programlama dilinde derlenen bu fonksiyonla HTTP-GET paketlerinin yakalandığı gözlemlenmiştir. Şekil 3'te yakalanan paketler yine Wireshark ekran görüntüsünde verilmiştir.



Şekil 3. TCP socket programming yöntemiyle HTTP-GET paketlerinin yakalanabildiği görülmektedir

NetFPGA-1G-CML arayüzüne kullanıcılardan gelen TCP/IP veri katarının haritalanmış hali Şekil 4'te oluşturulmuştur. Bu haritaya göre Linux tabanlı işletim sistemlerinde GET paketleri TCP_PAYLOAD katarının ilk oktetinde yani WORD_3'ün ikinci oktetinde TCP_PAYLOAD[14]'te yakalanmıştır.
Şekil 5'te görülen Windows işletim sistemi kullanan kullanıcıların GET paketlerinin ise yine TCP_PAYLOAD katarının ilk oktetinde yani WORD_2'nin onikinci oktetinde TCP_PAYLOAD[20]'de yakalandığı görülmüştür.

LINUX ETHERNET-IP-TCP PACKET HEADERS				
	1.	2.	3.	4.
	tdata(255:240)	tdata(239:224)	tdata(223:208)	tdata(207:192)
WORD_1	ETH DA[2]	ETH DA[1]	ETH DA[0]	ETH SA[2]
WORD_2	DST IP LOW	SRC PORT	DST PORT	SEQUENCE[1]
WORD_3	OPTIONS[0]	TCP PAYLOAD[14]	TCP PAYLOAD[13]	TCP PAYLOAD[12]
	5.	6.	7.	8.
	tdata(191:176)	tdata(175:160)	tdata(159:144)	tdata(143:128)
WORD_1	ETH SA[1]	ETH SA[0]	TYPE	VER, IHL, TOS
WORD_2	SEQUENCE[0]	ACKNOWLEDGE[1]	ACKNOWLEDGE[0]	DOFF, FLAGS
WORD_3	TCP PAYLOAD[11]	TCP PAYLOAD[10]	TCP PAYLOAD[9]	TCP PAYLOAD[8]
	9.	10.	11.	12.
	tdata(127:112)	tdata(111:96)	tdata(95:80)	tdata(79:64)
WORD_1	TOTAL LENGTH	IDENTIFICATION	FLAGS, FOFF	TTL, PROTO
WORD_2	WIN SIZE	CHECKSUM	URG POINTER	OPTIONS[5]
WORD_3	TCP PAYLOAD[7]	TCP PAYLOAD[6]	TCP PAYLOAD[5]	TCP PAYLOAD[4]
	13.	14.	15.	16.
	tdata(63:48)	tdata(47:32)	tdata(31:16)	tdata(15:0)
WORD_1	HEADER CSUM	SRC IP HIGH	SRC IP LOW	DST IP HIGH
WORD_2	OPTIONS[4]	OPTIONS[3]	OPTIONS[2]	OPTIONS[1]
WORD_3	TCP PAYLOAD[3]	TCP PAYLOAD[2]	TCP PAYLOAD[1]	TCP PAYLOAD[0]

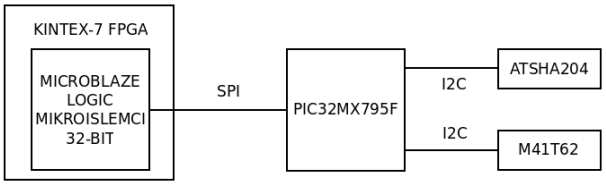
Şekil 4. Linux tabanlı işletim sistemlerinin TCP/IP paketi yapısı

MS WINDOWS ETHERNET-IP-TCP PACKET HEADERS				
	1.	2.	3.	4.
	tdata(255:240)	tdata(239:224)	tdata(223:208)	tdata(207:192)
WORD_1	ETH DA[2]	ETH DA[1]	ETH DA[0]	ETH SA[2]
WORD_2	DST IP LOW	SRC PORT	DST PORT	SEQUENCE[1]
WORD_3	TCP PAYLOAD[15]	TCP PAYLOAD[14]	TCP PAYLOAD[13]	TCP PAYLOAD[12]
	5.	6.	7.	8.
	tdata(191:176)	tdata(175:160)	tdata(159:144)	tdata(143:128)
WORD_1	ETH SA[1]	ETH SA[0]	TYPE	VER, IHL, TOS
WORD_2	SEQUENCE[0]	ACKNOWLEDGE[1]	ACKNOWLEDGE[0]	DOFF, FLAGS
WORD_3	TCP PAYLOAD[11]	TCP PAYLOAD[10]	TCP PAYLOAD[9]	TCP PAYLOAD[8]
	9.	10.	11.	12.
	tdata(127:112)	tdata(111:96)	tdata(95:80)	tdata(79:64)
WORD_1	TOTAL LENGTH	IDENTIFICATION	FLAGS, FOFF	TTL, PROTO
WORD_2	WIN SIZE	CHECKSUM	URG POINTER	TCP PAYLOAD[20]
WORD_3	TCP PAYLOAD[7]	TCP PAYLOAD[6]	TCP PAYLOAD[5]	TCP PAYLOAD[4]
	13.	14.	15.	16.
	tdata(63:48)	tdata(47:32)	tdata(31:16)	tdata(15:0)
WORD_1	HEADER CSUM	SRC IP HIGH	SRC IP LOW	DST IP HIGH
WORD_2	TCP PAYLOAD[19]	TCP PAYLOAD[18]	TCP PAYLOAD[17]	TCP PAYLOAD[16]
WORD_3	TCP PAYLOAD[3]	TCP PAYLOAD[2]	TCP PAYLOAD[1]	TCP PAYLOAD[0]

Şekil 5. Windows tabanlı işletim sistemlerinin TCP/IP paketi yapısı

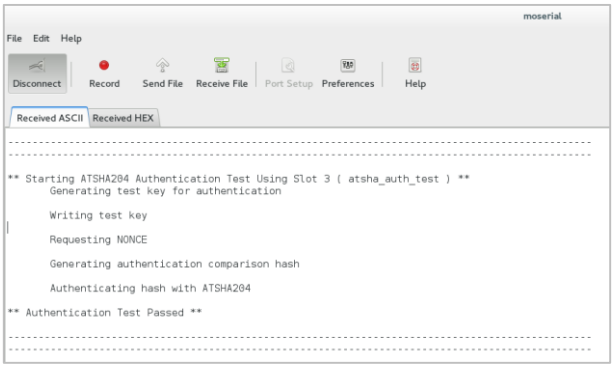
3.3. Şifreleme ve Gerçek Zaman Saati Entegreleri ile Haberleşilmesi Metodu

Çalışmamızda URL adreslerinin ve kullanıcıların kaynak IP adreslerinin yakalanmasının yanında FPGA içerisinde yazılımsal olarak tanımlanan 32-bitlik Microblaze mikroişlemcisinin şifreleme ve gerçek zaman saati entegreleri ile haberleşmesi metodu da gerçekleştirilmiştir. Bu haberleşme çalışılırken FPGA entegresi ile uçbirim olan ATSHA204 ve M41T62 entegreleri arasında aracı olarak PIC32MX795F512L mikrokontrolör entegresinden faydalanılmıştır. Şekil 6'da bu bağlantı mantığı detaylı olarak gösterilmiştir.



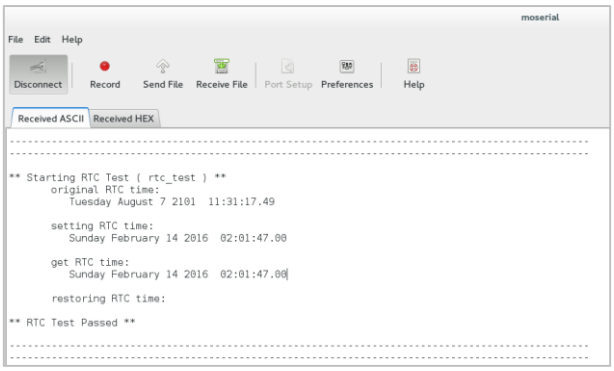
Şekil 6. FPGA entegresi ile ATSHA204 ve M41T62 entegrelerinin haberleştirilmesi

Bu haberleşme esnasında bizim yaptığımız çalışmada ATSHA204 entegresinin üzeinde bulunan 16 adet slottan 3. Slot içerisine doğrulama anahtar kodu yazılıraka sonra bu anahtar kodu tekrar okunup, doğrulama algoritması ile karşılaştırılarak doğrulama işlemi test edilmiştir. Şekil 7'de bu testin sonucu kendi konrolör bilgisayarımızdaki çıktısı ile görülmektedir.



Şekil 7. ATSHA204 entegresi doğrulama testi sonucu

M41T62 gerçek zaman saati entegresi için de benzer şekilde bir test prosedürü yazılmış olup, çalışmamızda gerçek zaman olarak testin gerçekleştirildiği 14 Şubat 2016 Pazar 02:01:47.00 saati yazılıp M41T62'ye gönderilmiş ve okunmuştur. Başarılı test sonucunun çıktısı Şekil 8'de verilmiştir.

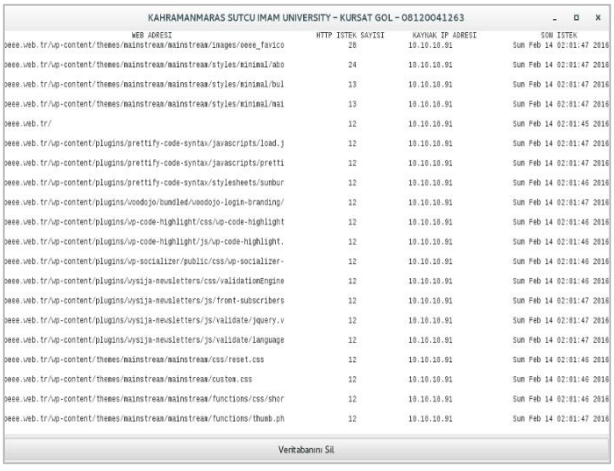


Şekil 8. M41T62 entegresi test sonucu

Çalışmamızdaki bu haberleşme testleri yapılırken Microblaze ile PIC32MX795 arasında çift yönlü SPI (Serial Peripheral Interface) protokolü, PIC32MX795 ile ATSHA204 ve M41T62 entegreleri arasında ise I2C (Inter Integrated Circuit) protokolü kullanılmıştır.
Testlerin sonuçlarını irdeleyebilmek ve ilgili entegrelerin arasındaki haberleşmeyi izleyebilmek için kullanıcı gözlem tarafında Microblaze mikrokontrolörünün RS232 (Recommended Standard 232) arayüzü kullanılmıştır. Fiziksel olarak Kintex-7 FPGA entegresinin dış dünyaya açılan pinleri kullanıldığı halde mantıksal olarak Microblaze mikrokontrolörünün UART (Universal Asynchronous Receiver Transmitter) modülünden faydalanılmıştır.

3.4. Elde Edilen Verilerin Ağ Yöneticisine GUI (Graphical User Interface) Aracılığıyla Sunulması

Testlerimizin ve paket yakalama-işleme süreçlerimizin sonucunda elde ettiğimiz verileri tek bir tabloda yansıtmak için Linux GTK kütüphanesi ile Şekil 9'da verilen tablo oluşturulup ağ yöneticisinin ekranına sunulmuştur. Bu çalışmada Fedora işletim sistemiyle çalışan kontrolör PC ağ yöneticisinin bilgisayarı konumundadır. Gerçek zaman, URL adresinin ilk 70 karakteri, istek adedi ve kullanıcının kaynak IP adresi sütunları en yüksek 20 istek adedine göre sıralanmıştır.



Şekil 9. Ağ Yöneticisine sunulan işlenmiş tablo

4. BULGULAR ve TARTIŞMA

Yaptığımız çalışma sırasında denediğimiz PCAP ve TCP socket programming yöntemlerinin sadece birinde HTTP-GET paketlerinin yakalanabildiğini gözlemlemiş olduk. PCAP metoduyla C dilinde derlediğimiz programla hiçbir TCP/IP paketi yakalayamadık. Buna karşılık olarak TCP socket programming temel fonksiyonu ile yine C dilinde derlediğimiz program yardımıyla ağ üzerindeki uç kullanıcıların hangi URL'e kaç adet erişim isteği yaptıklarını, istek gerçek zaman saatlerini ve isteği yaparken kullandıkları kaynak IP adresini tespit ederek kayıt altına alınması için veritabanı üzerinde tutarak ağ yöneticisine sunulmasını sağladık.
Testlerimiz için kullandığımız temel materyalimiz olan NetFPGA-1G-CML kartının üzerinde bulunan ATSHA204 şifreleme entegresi ve M41T62 gerçek zaman saati entegrelerini yazılımsal mikrokontrolör

Microblaze ile haberleştirdik. Bu iletişimi sağlayabilmek için SPI ve I2C protokollerini Microblaze üzerinde kullandık. Microblaze'in, donanım tanımlama dili verilog ile oluşturulan bir mikrokontrolör olmasına rağmen fiziksel mikrokontrolörlerin tüm yapısal özelliklerini gerçekleştirebildiğini deneyimlemiş olduk. UART arayüzü sayesinde Microblaze'in yan birimleriyle haberleşme aşamalarını gözledik.

5. SONUÇ

Bu çalışmamızda yerel-genel bilgisayar ağlarında uç kullanıcıların ağ tarayıcısı yoluyla erişmeye çalıştığı URL adreslerini, kullanıcının kaynak IP adresi, gerçek zaman saati ve erişim sayısına göre kayıt altına alınırken uygulanabilecek farklı metotları karşılaştırdık. NetFPGA-1G-CML ağ protokolleri geliştirme kartının şifreleme ve gerçek zaman saati özelliklerini test ettik ve bu özelliklerle ilgili uygulama gerçekleştirdik. Açık kaynak kodlu donanım ve yazılımların ağ yönetimi ve gözlemi konularında kolay bir şekilde kullanılabileceğini gördük. Ancak bu geliştirmeleri yaparken paket yakalama metodunun detaylı incelenerek, testlerle iyi araştırılması gereksinimiyle karşılaştık. Kullandığımız TCP socket programming metodu doğru bir şekilde ağ paketlerini yakalayıp istenilen şekilde kullanıcı bilgilerini bulabilmektedir.

Gelecek çalışmalarımızda yeni gelişmeye başlayan SDN (Software Defined Network) teknolojisi ve NetFPGA-1G-CML kartı ile OpenFlow 1.4 protokolünü çalıştırabilen bir ağ anahtarlama cihazı gerçekleştirmeyi hedefliyoruz.

KAYNAKLAR

Craig, H., 1997. TCP/IP Network Administration. O'Reilly., Pages 152-162.

Lockwood, J.W., McKeown, N. and Watson, G., 2007. NetFPGA-An Open Platform for Gigabit Rate Network Switching and Routing. IEEE., San Diego

Naous, J., McKeown, N., and Bolouki, S., 2008. NetFPGA:Reusable Router Architecture for Experimental Research. ACM, Seattle

Kreutz, D., Ramos, F.M.V., Verissimo, P., 2014. Software Defined Networking. IEEE., Page 6

Cankaya, H.C., 2014. Software Defined Networking and Security for Utilities., Phoenix., Page 4