Lampiran

Instalasi Wireshark pada Server

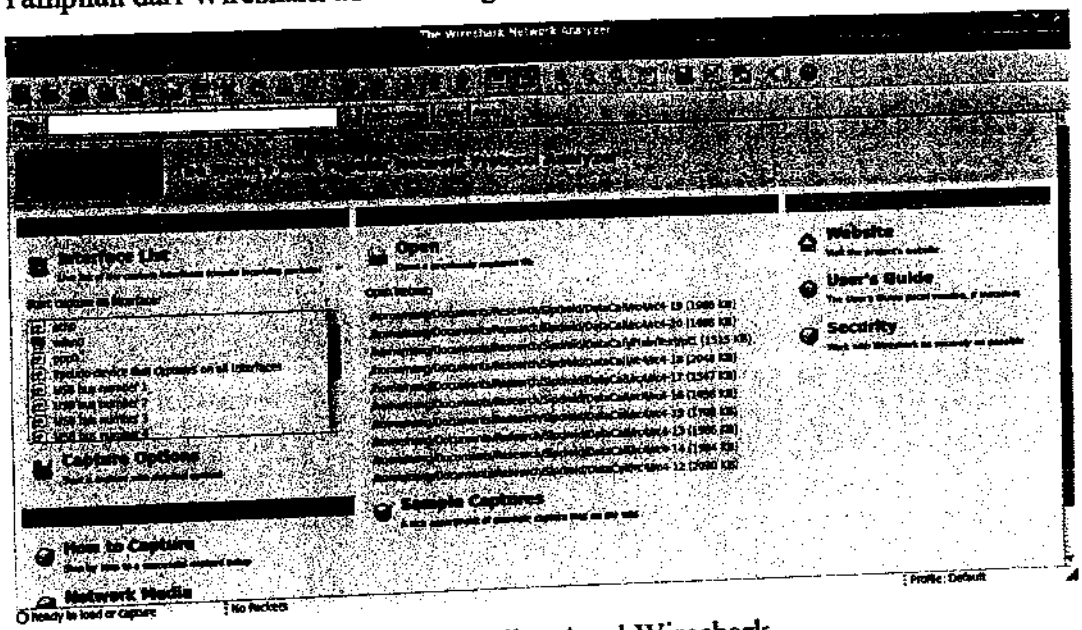
Instalasi Wireshark pada masing-masing distro linux terdapat beberapa perbedaan. Biasanya tiap-tiap distro sudah menyiapkan paket instalasi Wireshark yang biasanya dijadikan satu pada repositori linux masing-masing distro. Penelitian ini menggunakan Ubuntu linux 10.04 sebagai inang dari server VoIP, sehingga hanya perlu instalasi Wireshark pada sistem operasi inang. Untuk melakukan instalasi Wireshark pada Ubuntu hanya perlu menjalankan script berikut:

```
$ sudo apt-get install wireshark
```

Jika sudah selesai melakukan instalasi, maka untuk menjalankan Wireshark adalah sebagai berikut:

```
$ sudo wireshark
```

Tampilan dari Wireshark adalah sebagai berikut:



Gambar Tampilan Awal Wireshark

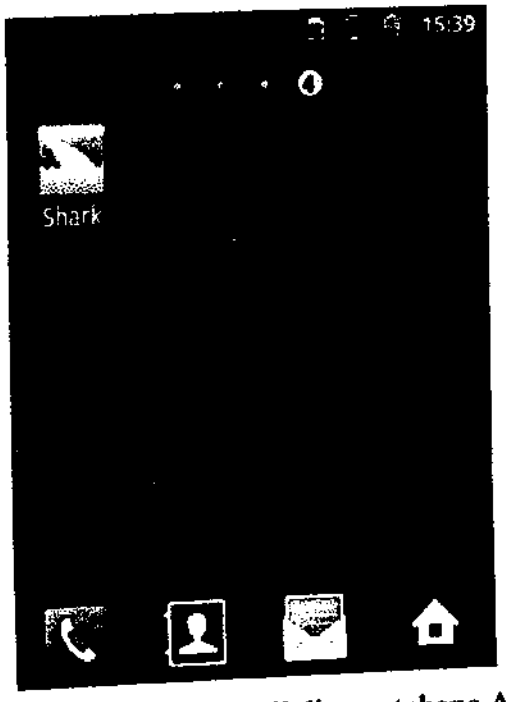
Untuk menangkap data paket yang akan dianalisa, hanya tinggal memilih *interface* pada menu *Capture* yang ada di tampilan awal atau bisa juga dari menu Capture kemudian pilih Interfaces, kotak dialog baru akan muncul, kemudian tekan tombol start pada interface yang diinginkan.

Instalasi Shark pada Smartphone Android

Untuk melakukan instalasi shark pada smartphone Android Tahapan-tahapannya adalah sebagai berikut:

1. Membuka market yang terdapat pada smartphone Android. Minimanl harus terhubung dengan internet melalui jaringan WLAN ataupun packet data.
2. Mengetikkan shark pada kolom pencarian.
3. Jika sudah muncul shark pada tampilan dari Android market, selanjutnya masuk pada tahapan instalasi, mengklik ikon shark pada daftar pencarian di market Android.
4. Mengkonfirmasi jalannya instalasi.
5. Proses pengunduhan paket Shark akan dilakukan setelah konfirmasi.
6. Setelah proses unduh selesai, selanjutnya proses instalasi paket Shark ke smartphone Android.
7. Selesai, dan Shark bisa digunakan untuk menangkap packet data dari jaringan wireless dan juga jaringan gsm.

Hasil Capture menggunakan Shark yang berjalan di Android



Gambar Shark yang terinstall di smartphone Android

Perangkat Keras dan Perangkat Lunak yang diperlukan Server VoIP

Perangkat keras yang diperlukan untuk membangun sebuah server VoIP adalah sebagai berikut:

- Komputer dengan spesifikasi sebagai berikut:
  - Prosesor Standar 900 MHz.
  - RAM 1 GB.
  - Hard disk 20 GB.
- *Access Point.*
- *Smartphone* Android dengan sistem operasi Android 2.3.

Sedangkan perangkat lunak yang diperlukan untuk membangun server VoIP pada penelitian ini, digunakan beberapa perangkat lunak berikut:

- Asterisk.
- FreePBX.
- MySQL.
- Dahdi.
- OpenSSL.

Perangkat-perangkat lunak tersebut telah dibungkus menjadi satu dalam sebuah sistem operasi Centos dan dibuat iso dengan nama AsteriskNow32.iso.

Instalasi Sipdroid pada Smartphone Android

Setelah server VoIP telah siap, selanjutnya adalah menyiapkan VoIP client Sipdroid yang sudah diintegrasikan dengan modul enkripsi pada smartphone Android. Untuk tahap ini, berkas instalasi yang berekstensi apk harus disalin ke dalam smartphone Android.

Untuk menyalin berkas apk yang diperlukan bisa mengkoneksikan smartphone Android dengan laptop yang menjadi lingkungan pengembangan Android.menggunakan media kabel USB atau menggunakan media bluetooth. Proses penyalinan berkas apk tersebut sama seperti menyalin berkas dari laptop ke dalam usb flashdisk, secara otomatis berkas akan disalin ke dalam SD card dari smartphone Android.

Tahap instalasi berkas apk yang telah disalin ke dalam smartphone Android sangatlah mudah. Tahap-tahapnya adalah sebagai berikut:

1. Buka aplikasi jelajah berkas yang ada di Android, biasanya bawaan dari Android adalah aplikasi bernama "MyFiles".
2. Cari path di mana berkas apk tersebut disalin dalam direktori-direktori yang ada di SD Card.
3. Sentuh ikon dari berkas apk yang ingin diinstal pada layar, maka proses instalasi akan dimulai.
4. Ikuti proses instalasi sampai dengan selesai.
5. Jika terdapat pesan yang menandakan bahwa instalasi berhasil, berarti aplikasi sudah terinstal dengan sukses, sebaliknya jika muncul pesan yang menandakan bahwa aplikasi gagal, telusuri kembali apa yang menyebabkan aplikasi gagal diinstal.

Tahap-tahap di atas adalah cara instalasi dari berkas apk secara umum, untuk instalasi Sipdroid.apk hanya perlu untuk menekan ikon dari Sipdroid yang telah disalinkan ke dalam SD Card smartphone Android.

Selain dengan cara menyalin berkas apk ke dalam smartphone Android, ada sebuah cara lagi untuk menginstal berkas apk ke dalam smartphone Android. Cara tersebut menggunakan perangkat ADB (*Android Debug Bridge*) yang berada di dalam Android SDK.

Adapun tahapan-tahapan untuk menyalin menggunakan berkas apk ke dalam smartphone Android menggunakan perangkat ADB adalah sebagai berikut:

1. Pindah ke dalam direktori Android SDK yang telah diinstall.
2. Masuk ke dalam direktori tools
3. Merestart adb dengan perintah
    - $ adb kill server
    - $ adb start-server
4. Cek koneksi dengan perangkat smartphone menggunakan perintah
    - $ adb devices

    Jika sudah muncul tipe dari perangkat smartphone pada hasil eksekusi perintah di atas, berarti perangkat sudah terhubung.

5. Jika sudah terkoneksi, selanjutnya instal berkas apk dengan perintah sebagai berikut:
    - $ adb install [path_dari_berkas_apk]

    Sebagai contoh:
    - $ adb instal Sipdroid.apk
6. Jika instalasi berhasil, akan muncul ikon dari aplikasi yang diinstal pada bagian menu di smartphone Android.

Membangun Server VoIP

Server VoIP yang dibangun pada penelitian ini akan dijalankan pada mesin virtual (*virtual machine*) pada sebuah laptop yang sudah terinstal sistem operasi Ubuntu Linux 10.04. Untuk membangun server VoIP ini, perangkat keras yang digunakan adalah sebagai berikut:

- Laptop dengan spesifikasi sebagai berikut:
    - Prosesor Intel Dual Core, 1.3 GHz.
    - RAM 2 GB.
    - HardDisk 320 GB.
- Wireless Router sebagai Access Point
    - Mendukung DHCP
    - Standar Protokol IEEE 802.11 b/g/n.
    - Mendukung WLAN dan LAN
    - Data rate sampai dengan 150 Mbps.
- Smartphone Android
    - Sistem operasi Android 2.3
    - Prosesor 800 MHz.
    - Penyimpanan (storage) internal 158 MB, eksternal 2 GB.
    - RAM 278 MB.
    - Wireless 802.11 b/g/n

Sedagnkan untuk perangkat lunak yang diperlukan untuk membangun server VoIP pada penelitian ini adalah sebagai berikut:

- Virtual Box OSE.

- AsteriskNow32.iso.
    - Sistem operasi Centos
    - Asterisk
    - Freepbx
    - MySQL
    - Dahdi
    - OpenSSL

Untuk melakukan instalasi AsteriskNow32.iso pada mesin virtual Virtual Box OSE, langkah-langkahnya adalah sebagai berikut:

1. Jalankan Virtual Box OSE pada laptop, jika sudah terinstal Virtual Box OSE. Jika belum terinstal, instal terlebih dahulu Virtual Box OSE pada sistem operasi host.
2. Unduh AsteriskNow.iso pada alamat berikut: http://dl.digium.com/load_balance.php?q=AsteriskNOW-1.7.1-i386.iso
3. Buat sebuah mesin baru pada Virtual Box sesuai dengan kebutuhan dari AsteriskNow32.iso. Untuk AsteriskNow32.iso hanya perlu sekitar 4 GB untuk instalasi di mesin virtual.
4. Tambahkan AsteriskNow32.iso pada mesin yang baru dibuat sebagai sumber untuk instalasi.
5. Lakukan instalasi seperti instalasi sistem operasi linux biasa.
6. Jika sudah selesai, dan Virtual Box meminta untuk merestart mesin virtualnya, hapus AsteriskNow32.iso dari mesin virtual.
7. Selanjutnya jalankan mesin virtual yang sudah terinstal AsteriskNow32.
8. Untuk konfigurasi, FreePBX dapat diakses dari alamat IP mesin virtual yang sudah berjalan.

Source Hasil Integrasi

Enkripsi dan Dekripsi RC4

```
public byte[] encryptRC4(byte[] s, byte[] byteKey) {
                Cipher cipher;
                byte[] bytePack = new byte[s.length];
                // create key
                SecretKeySpec sKey = new SecretKeySpec(byteKey, "RC4");
```

```
                // Activating chiper
                try {
                        cipher = Cipher.getInstance("RC4");
                        cipher.init(Cipher.ENCRYPT_MODE, sKey);
                        // encryp data
                        bytePack = cipher.doFinal(s);
                } catch (Exception e) {
                        //
                }
                return bytePack;
        }
public byte[] decryptRC4(byte[] s, byte[] byteKey) {
                Cipher cipher;
                byte[] bytePack = new byte[s.length];
                // create key
                SecretKeySpec sKey = new SecretKeySpec(byteKey, "RC4");
                // Activating chiper
                try {
                        cipher = Cipher.getInstance("RC4");
                        cipher.init(Cipher.DECRYPT_MODE, sKey);
                        // decryp data
                        bytePack = cipher.doFinal(s);
                } catch (Exception e) {

                }
                return bytePack;
        }
```

Enkripsi dan Dekripsi AES

```
public byte[] encryptAES(String key, byte[] atad){
                byte[] byteEnc = new byte[atad.length];
                try {
                        byte[] keyByte = key.getBytes();
                        KeyGenerator kGen = KeyGenerator.getInstance("AES");
                        SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");
                        sr.setSeed(keyByte);
                        kGen.init(128, sr);
                        SecretKey sKey = kGen.generateKey();
                        byte[] sRaw = sKey.getEncoded();
                        SecretKeySpec sKS = new SecretKeySpec(sRaw, "AES");
                        Cipher cipher = Cipher.getInstance("AES/OFB/NoPadding");
                        cipher.init(Cipher.ENCRYPT_MODE, sKS,
ivParameterSpecAES);
                        byteEnc = cipher.doFinal(atad);
                } catch (Exception e) {

                }
                return byteEnc;
        }

        public byte[] decryptAES(String key, byte[] atad){
                byte[] byteEnc = new byte[atad.length];
                try {
```

```
                byte[] keyByte = key.getBytes();
                KeyGenerator kGen = KeyGenerator.getInstance("AES");
                SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");
                sr.setSeed(keyByte);
                kGen.init(128, sr);
                SecretKey sKey = kGen.generateKey();
                byte[] sRaw = sKey.getEncoded();
                SecretKeySpec sKS = new SecretKeySpec(sRaw, "AES");
                Cipher cipher = Cipher.getInstance("AES/OFB/NoPadding");
                cipher.init(Cipher.DECRYPT_MODE, sKS,
ivParameterSpecAES);
                byteEnc = cipher.doFinal(atad);
            } catch (Exception e) {

            }
            return byteEnc;
        }
```

Enkripsi dan Dekripsi DES

```
public byte[] encryptDES(String key, byte[] atad){
            byte[] byteEnc = new byte[atad.length];
            try {
                byte[] keyByte = key.getBytes();
                KeyGenerator kGen = KeyGenerator.getInstance("DES");
                SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");
                sr.setSeed(keyByte);
                kGen.init(64, sr);
                SecretKey sKey = kGen.generateKey();
                byte[] sRaw = sKey.getEncoded();
                SecretKeySpec sKS = new SecretKeySpec(sRaw, "DES");
                Cipher cipher = Cipher.getInstance("DES/OFB/NoPadding");
                cipher.init(Cipher.ENCRYPT_MODE, sKS,
ivParameterSpecDES);
                byteEnc = cipher.doFinal(atad);
            } catch (Exception e) {

            }
            return byteEnc;
        }

        public byte[] decryptDES(String key, byte[] atad){
            byte[] byteEnc = new byte[atad.length];
            try {
                byte[] keyByte = key.getBytes();
                KeyGenerator kGen = KeyGenerator.getInstance("DES");
                SecureRandom sr = SecureRandom.getInstance("SHA1PRNG");
                sr.setSeed(keyByte);
                kGen.init(64, sr);
                SecretKey sKey = kGen.generateKey();
                byte[] sRaw = sKey.getEncoded();
                SecretKeySpec sKS = new SecretKeySpec(sRaw, "DES");
                Cipher cipher = Cipher.getInstance("DES/OFB/NoPadding");
```

```
                    cipher.init(Cipher.DECRYPT_MODE, sKS,
ivParameterSpecDES);
                    byteEnc = cipher.doFinal(atad);
                } catch (Exception e) {

                }
                return byteEnc;
        }
```

Perubahan pada RTPSocket.java

```
/** Receives a RTP packet from this socket */
        public void receive(RtpPacket rtpp) throws IOException {
                // DIRUBAH
                String kunciTemp = null;
                byte[] keyEnc = null;
                kunciTemp = Receiver.getKunciEnkripsi();
                keyEnc = kunciTemp.getBytes();
                String algoEnc = Receiver.getAlgoEnkripsi();

                datagram.setData(rtpp.packet);
                datagram.setLength(rtpp.packet.length);
                socket.receive(datagram);
                // DIRUBAH
                byte[] isiPayload = rtpp.getPayload();
                if(algoEnc.equals("RC4")) {
                        isiPayload = sec.decryptRC4(isiPayload, keyEnc);
                        rtpp.setPayload(isiPayload, isiPayload.length);
                } else if (algoEnc.equals("AES")){
                        isiPayload = sec.decryptAES(kunciTemp, isiPayload);
                        rtpp.setPayload(isiPayload, isiPayload.length);
                } else if(algoEnc.equals("DES")) {
                        isiPayload = sec.decryptDES(kunciTemp, isiPayload);
                        rtpp.setPayload(isiPayload, isiPayload.length);
                } else if(algoEnc.equals("PT")){
                        // plaintext ( do nothing )
                }

                if (!socket.isConnected())
                        socket.connect(datagram.getAddress(),datagram.getPort());
                rtpp.packet_len = datagram.getLength();
        }

        /** Sends a RTP packet from this socket */
        public void send(RtpPacket rtpp) throws IOException {
                // DIRUBAH
                String kunciTemp = null;
                byte[] keyEnc = null;
                kunciTemp = Receiver.getKunciEnkripsi();
                keyEnc = kunciTemp.getBytes();
                String algoEnc = Receiver.getAlgoEnkripsi();

                // DIRUBAH
                byte[] isiPayload = rtpp.getPayload();
```

```
                //isiPayload = sec.encrypDataAES(isiPayload, keyEnc);
                if(algoEnc.equals("RC4")) {
                        isiPayload = sec.encryptRC4(isiPayload, keyEnc);
                        rtpp.setPayload(isiPayload, isiPayload.length);
                } else if (algoEnc.equals("AES")){
                        isiPayload = sec.encryptAES(kunciTemp, isiPayload);
                        rtpp.setPayload(isiPayload, isiPayload.length);
                } else if(algoEnc.equals("DES")) {
                        isiPayload = sec.encryptDES(kunciTemp, isiPayload);
                        rtpp.setPayload(isiPayload, isiPayload.length);
                } else if(algoEnc.equals("PT")){
                        // plaintext ( do nothing )
                }

                datagram.setData(rtpp.packet);
                datagram.setLength(rtpp.packet_len);
                datagram.setAddress(r_addr);
                datagram.setPort(r_port);
                socket.send(datagram);
        }
```

Penambahan crypthopreferences.xml

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen
 xmlns:android="http://schemas.android.com/apk/res/android">
 <PreferenceCategory
            android:title="Algoritma Enkripsi"
            android:key="algoritma_option">

                <ListPreference
                        android:key="list_algoritma"
                        android:title="Pilihan Algoritma"
                        android:summary="Algoritma untuk melakukan enkripsi
data"
                        android:defaultValue="RC4"
                        android:entries="@array/option_algoritma"
                        android:entryValues="@array/value_algoritma"
                />
 </PreferenceCategory>


 <PreferenceCategory
            android:title="Kunci Enkripsi"
            android:key="key_input">
            <EditTextPreference
        android:summary="Kunci untuk melakukan enkripsi"
        android:defaultValue="11111111"
        android:title="Kunci Enkripsi"
        android:key="editKeyInput"
        />
 </PreferenceCategory>
</PreferenceScreen>
```

Perubahan pada Sipdroid.java

```
        // UBAH
        private static final int KEY_MANAGEMENT = FIRST_MENU_ID + 4;

@Override
        public boolean onOptionsItemSelected(MenuItem item) {
                boolean result = super.onOptionsItemSelected(item);
                Intent intent = null;

                switch (item.getItemId()) {
                case ABOUT_MENU_ITEM:
                        if (m_AlertDlg != null)
                        {
                                m_AlertDlg.cancel();
                        }
                        m_AlertDlg = new AlertDialog.Builder(this)

        .setMessage(getString(R.string.about).replace("\\n","\n").replace("${VERSION}",
getVersion(this)))
                        .setTitle(getString(R.string.menu_about))
                        .setIcon(R.drawable.icon22)
                        .setCancelable(true)
                        .show();
                        break;

                case EXIT_MENU_ITEM:
                        on(this,false);
                        Receiver.pos(true);
                        Receiver.engine(this).halt();
                        Receiver.mSipdroidEngine = null;
                        Receiver.reRegister(0);
                        stopService(new Intent(this,RegisterService.class));
                        finish();
                        break;

                case CONFIGURE_MENU_ITEM: {
                        try {
                                intent = new Intent(this,
 org.sipdroid.sipua.ui.Settings.class);
                                startActivity(intent);
                        } catch (ActivityNotFoundException e) {
                        }
                }
                        break;
                case KEY_MANAGEMENT:{
                        try {
                                intent = new Intent(this,
 org.sipdroid.sipua.ui.KeyManagement.class);
                                startActivity(intent);
                        } catch (ActivityNotFoundException e) {
                        }

                }
                /*case KONTAK_MENU_ITEM: {
                        try {
                                intent = new Intent(this,
 org.sipdroid.sipua.ui.Kontak.class);
```

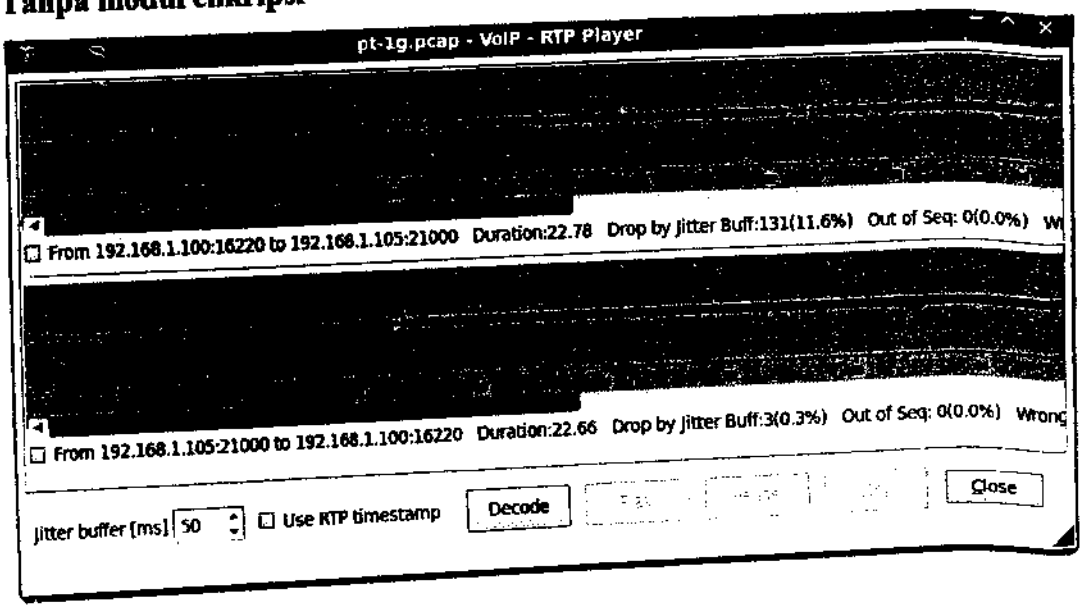
```
                        startActivity(intent);
                    } catch (ActivityNotFoundException e) {

                    }
                }*/
                    break;
                }

                return result;
        }
```

Perubahan pada Receiver.java

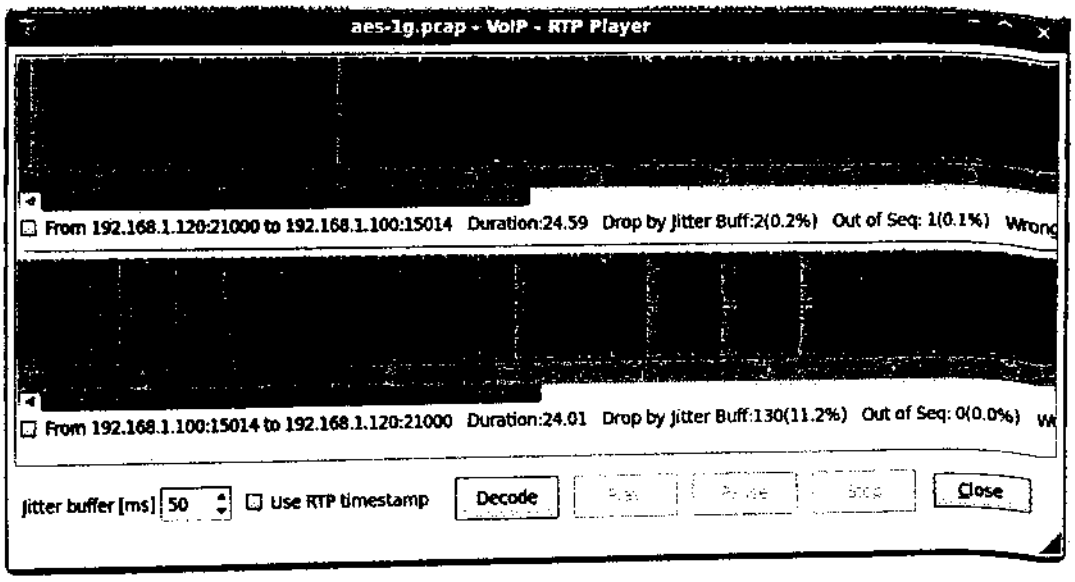
```
// START UBAH
                public static String kunciShared = null;
                // END UBAH

                // START UBAH
                public static String getKunciEnkripsi(){
                        String kunci = null;
                        SharedPreferences encKeyPrefs =
mContext.getSharedPreferences("CryptoPreferences", MODE_PRIVATE);
                        kunci = (encKeyPrefs.getString("editKeyInput",
"1111111111111111"));
                        return kunci;
                }
```

Hasil Decode RTP

Tanpa modul enkripsi

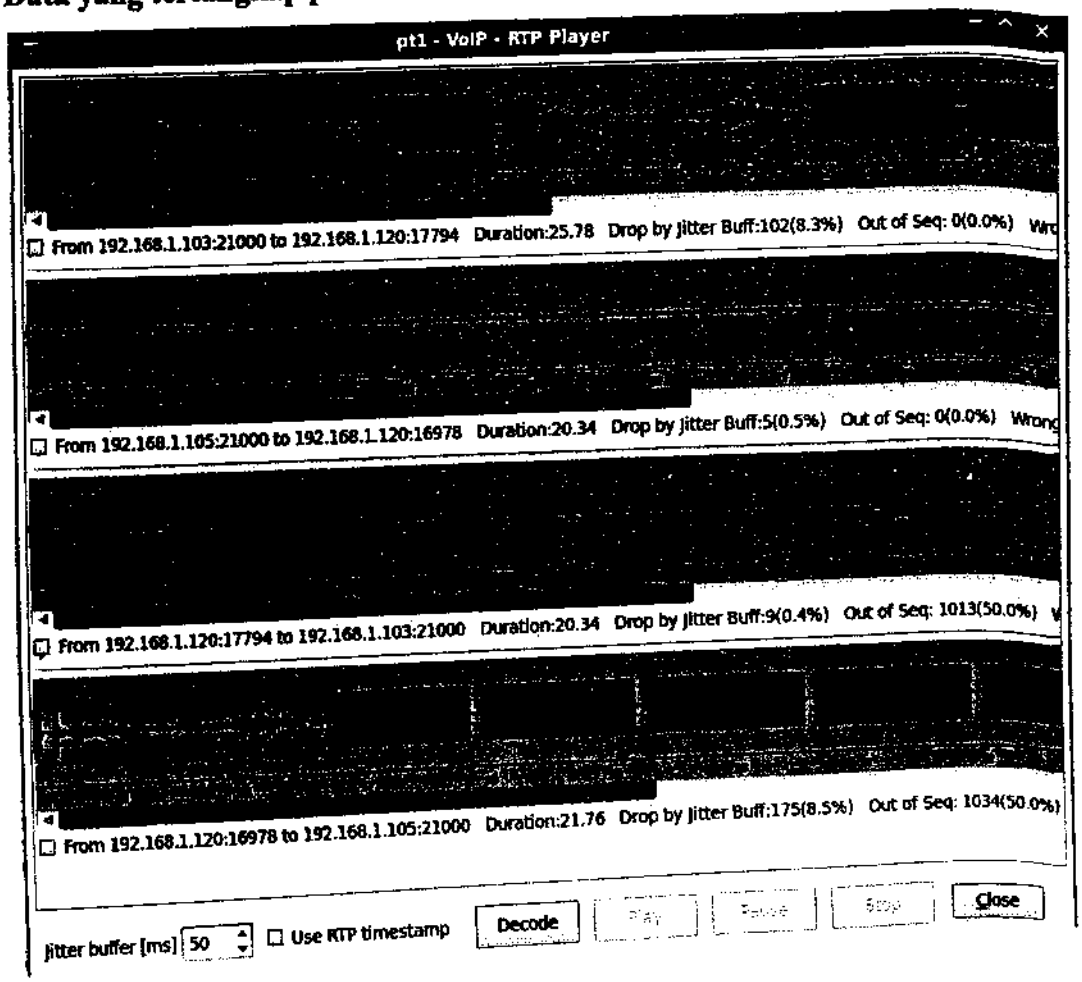


Menggunakan Modul Enkripsi

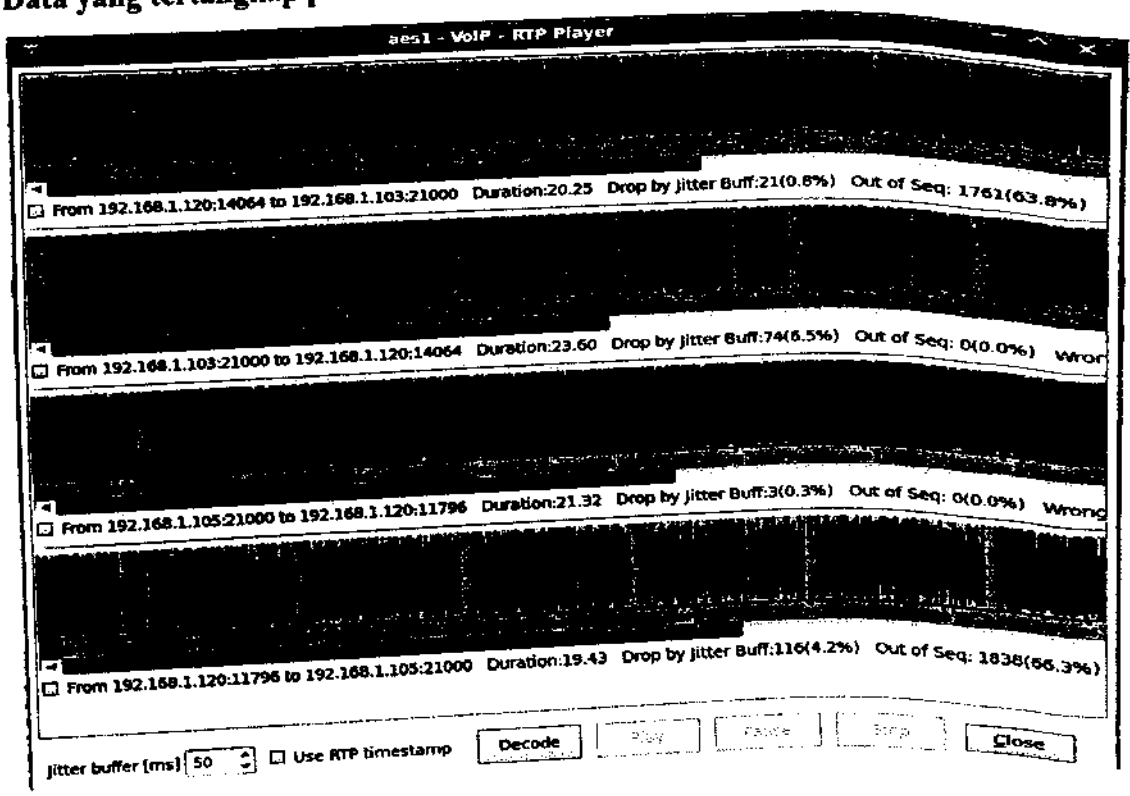


Gambar Penangkapan Data pada Simulasi Penyerangan

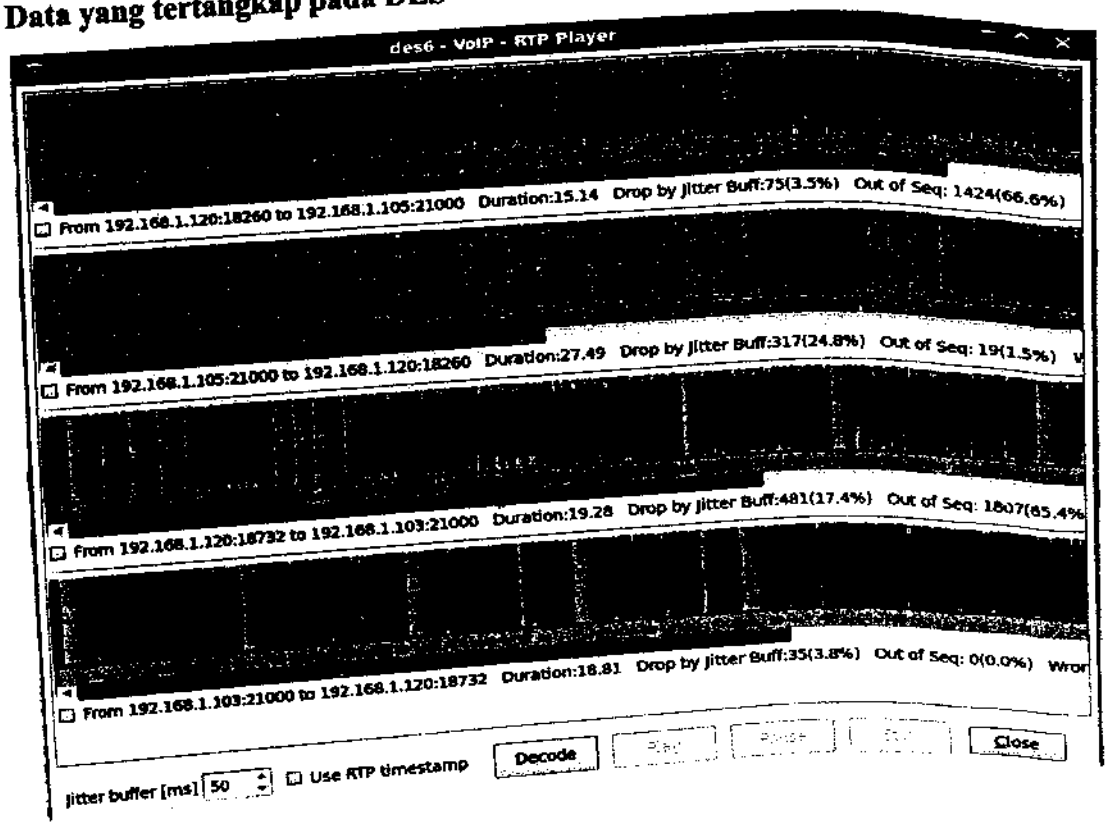
Data yang tertangkap pada Plain text.



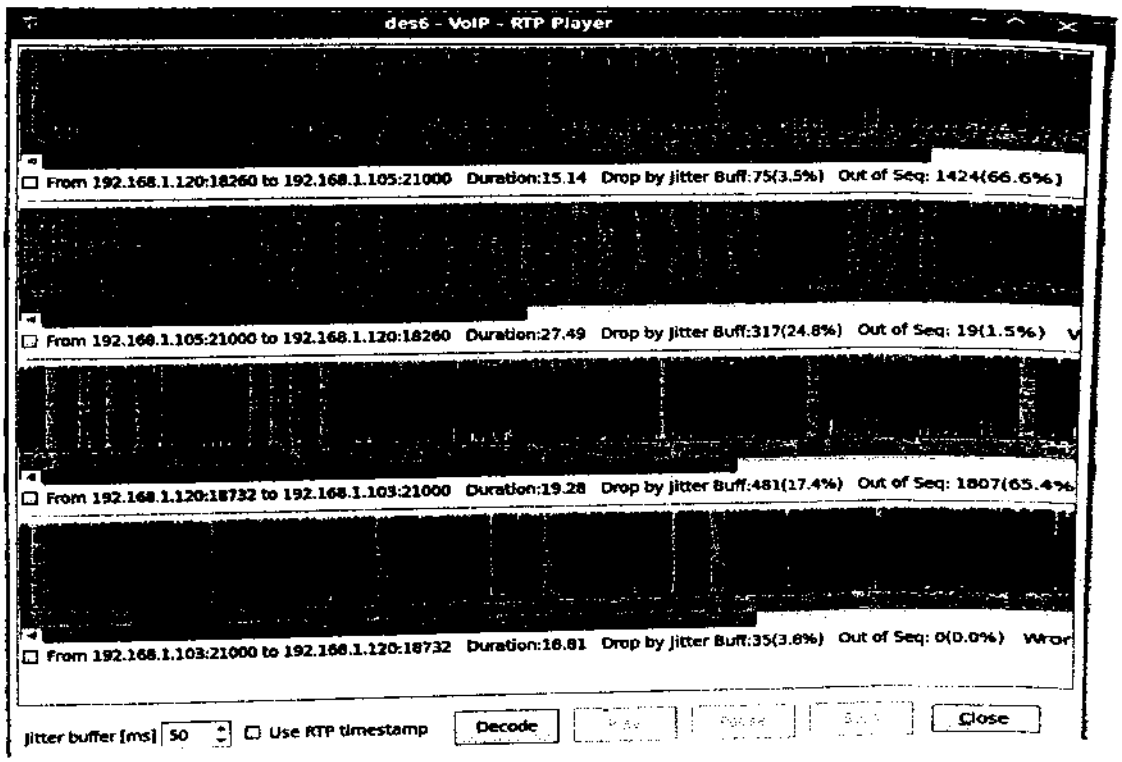
Data yang tertangkap pada AES



Data yang tertangkap pada DES



Data yang tertangkap pada DES



Data Hasil Pengukuran

Delay

Call	PT	AES	DES	RC4
1	20.07765	20.1062	20.17485	20.6969
2	19.9317	20.2209	20.2994	19.9563
3	20.25115	20.0289	20.3001	20.09365
4	19.68765	20.24035	19.97815	20.2497
5	20.1969	20.13905	20.27035	19.85285
6	20.3132	20.14095	20.19205	20.19315
7	19.9164	20.21785	20.25745	20.06415
8	20.0318	20.2997	20.0558	20.0876
9	20.14745	20.20875	20.22715	19.95505
10	20.175	20.22745	20.0422	20.03335
11	20.28495	19.99455	20.22145	20.08215
12	19.96685	19.9808	20.0921	20.3892
13	20.26045	19.9961	20.11735	20.30925
14	20.1974	20.0451	19.96365	19.97385
15	19.96395	20.109	20.16325	20.14275
16	20.05545	20.14525	19.9185	19.95175
17	20.21245	19.92005	20.07175	19.91625
18	20.3047	20.10905	20.01145	20.10475
19	20.02495	20.1216	19.9998	19.92255
20	20.13905	19.98175	19.93745	20.23965

Throughput

Call	PT	AES	DES	RC4
1	78.80055	79.6316	79.1115	78.93585
2	78.99205	78.7754	79.42175	78.8797
3	79.43115	79.2776	78.8535	78.73115
4	78.779	79.03945	79.01045	78.5158
5	78.62065	78.87365	78.73235	78.829
6	78.6712	78.94885	78.9045	78.7291
7	78.37575	78.78305	78.75885	78.8263
8	79.3668	78.41215	79.15365	78.89885
9	78.6695	78.68875	78.71595	78.65435
10	78.0557	78.8611	78.95195	78.7561
11	78.65445	78.67275	78.2989	78.8523
12	78.7394	78.6861	78.75255	78.8063
13	78.6643	79.00085	78.5651	78.7358
14	78.78745	79.21335	78.755	78.9576
15	79.1351	78.89055	78.29235	78.7207
16	79.43045	78.55925	78.8554	78.94005
17	78.5964	79.11415	78.9701	78.6444
18	78.67185	78.78915	78.95715	79.00765
19	79.01585	78.4119	78.7351	78.92275
20	78.79135	78.37215	78.7643	78.8511

Packet Loss

Call	PT	AES	DES	RC4
1	1.92%	0.65%	0.25%	2.01%
2	0.66%	0.47%	2.33%	1.06%
3	0.77%	1.27%	2.05%	1.00%
4	0.59%	1.46%	0.64%	0.18%
5	0.20%	1.37%	0.58%	0.28%
6	1.26%	0.27%	0.18%	0.85%
7	1.00%	1.20%	0.37%	0.58%
8	0.77%	0.66%	1.45%	0.48%
9	0.71%	0.58%	0.29%	0.55%
10	1.48%	0.35%	1.09%	0.28%
11	0.70%	0.66%	0.47%	0.80%
12	0.52%	0.95%	0.99%	1.60%
13	0.59%	0.38%	0.87%	0.63%
14	0.57%	0.75%	1.35%	1.30%
15	0.49%	0.48%	0.18%	0.65%
16	0.18%	1.60%	1.45%	1.41%
17	0.48%	1.58%	0.82%	0.89%
18	0.47%	0.17%	1.28%	0.53%
19	0.83%	0.78%	1.49%	1.27%
20	0.36%	0.64%	0.76%	0.51%

MOS

Pengujian skala lab

Pengguna	Sipdroid Normal	Sipdroid AES	Sipdroid DES	Sipdroid RC4
1	5	3	3	3
2	4	3	3	3
3	5	4	3	4
4	5	3	3	4
5	5	4	4	3
6	5	4	3	3
7	5	3	3	3
8	4	3	3	3
9	4	3	2	3
10	5	3	3	3
11	5	3	3	3
12	4	3	3	3
13	5	3	4	3
14	5	3	3	3
15	5	4	3	4
16	5	3	4	3
17	4	4	3	3
18	5	4	3	4
19	5	3	3	3
20	5	4	4	3
21	4	3	4	4
22	5	3	3	4
23	5	3	3	3
24	5	4	3	3
25	4	3	3	3
26	5	3	3	3
27	5	4	3	3
28	5	3	4	3
29	4	4	3	4
30	5	3	3	3
	4.733333333	3.33333333	3.16666667	3.23333333

Pengujian MOS pada alpha testing

Pengguna	Sipdroid Normal	Sipdroid UMB AES
1	5	3
2	4	3
3	5	4
4	5	3
5	5	3
6	5	4
7	5	3

8	5	3
9	4	3
10	5	3
11	5	3
12	4	3
13	4	3
14	5	4
15	5	4
16	5	3
17	4	4
18	5	4
19	5	4
20	5	4
21	4	3
22	5	3
23	5	3
24	4	4
25	4	3
26	5	3
27	5	4
28	5	3
29	4	3
30	4	3
	4.666666667	3.33333333