PENGELOLAAN REMOTE KEAMANAN *EMBEDDED SYSTEM* PADA SISTEM OPERASI MIKROTIK ROUTER OS *MENGGUNAKAN PROPRIETARY PROTOCOLS*

[1]**Muhammad Imam Ghozali ,** [2]**Wibowo Harry Sugiharto**
[1 2] Program Studi Teknik Informatika, Fakultas Teknik, Universitas Muria Kudus
Gondangmanis, PO Box 53, Bae, Kudus 59352
[*]Email: [1] imam.ghozali@umk.ac.id, [2] wibowo.harrys@umk.ac.id

Abstrak

Keamanan data dalam komunikasi merupakan menjadi hal yang penting di era digital sekarang. Keamanan Protokol komunikasi yang ada pada sistem operasi bisa menjadi hal yang vital dari keamanan data, seperti pada system operasi MikroTik RouterOS. Protokol komunikasi MikroTik RouterOS dirancang khusus untuk melakukan manajemen jarak jauh. Pada umumnya protokol ini ditutup dan tidak dapat diakses oleh publik. Risiko keamanan yang terkait dengan penggunaan protokol komunikasi ini juga dianalisis dalam makalah ini. Serangan yang menggunakan bug konseptual dalam desain salah satu protokol komunikasi ini ditunjukkan pada contoh nyata. Akses penuh ke embedded system yang berjalan dengan sistem operasi ini bisa mengalami serangan ini.

***Kata kunci**: keamanan, ,mikrotik Router OS*

1. PENDAHULUAN

Sebagian besar embedded system saat ini, memungkinkan untuk melakukan manajemen jarak jauh. Desain dari embedded system menerapkan protokol komunikasi standar yang sudah diverifikasi ke dalam embedded system mereka atau merancang protokol komunikasi mereka sendiri untuk tujuan keamanan. Menggunakan protokol komunikasi mereka sendiri memiliki kelebihan tertentu. Perancang embedded system dapat membuat protokol komunikasi yang akan disesuaikan dengan aplikasi dan menawarkan pilihan yang tidak dapat kenali oleh protokol komunikasi standar. Protokol komunikasi MTP (MikroTik Transmission Protocol), yang digunakan untuk pengelolaan remote embedded system pada MikroTik RouterOS, dapat menjadi demonstrasi bagus dari fakta ini. Ini adalah protokol komunikasi yang dirancang lebih pada fitur utilitas tinggi daripada keamanan transmisi data. Pada artikel ini ditunjukkan bagaimana kombinasi beberapa fitur dasar dan kenyamanan dapat terjadi menjadi protokol komunikasi dengan bug yang sangat mendasar dalam hal keamanannya.

Sistem operasi MikroTik RouterOS adalah sistem operasi yang dikembangkan secara komersial yang dirancang khusus untuk perangkat embedded yang berfungsi sebagai komponen jaringan aktif seperti router, titik akses nirkabel, stasiun klien dan lain-lain. Sistem operasi ini dikembangkan oleh perusahaan MikroTik. Untuk menjalankan sistem operasi ini, perangkat keras khusus MikroTik RouterBoard telah dikembangkan. Sebagian besar dari perangkat keras ini berbasis pada arsitektur Mips dan desktop. Sistem operasi ini juga memungkinkan untuk dijalankan pada PC standar dengan arsitektur x86.

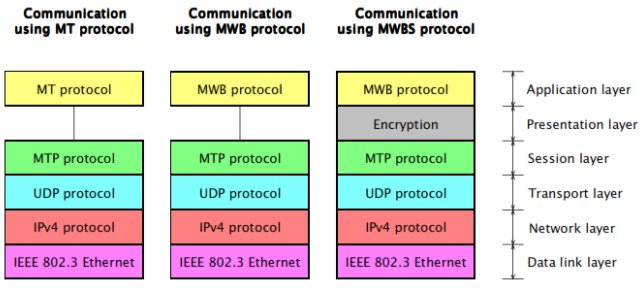
Pengelolaan embedded system pada MikroTik RouterOS dapat dilakukan baik secara lokal melalui serial RS-232 link atau jarak jauh melalui jaringan komunikasi dimana embedded system terhubung (Ethernet, WiFi, dan lain-lain). Manajemen jarak jauh embedded system dapat dilakukan dengan menggunakan protokol komunikasi standar (HTTP, HTTPS, SSH, Telnet, SNMP dan FTP) (MikroTik, 2011) dengan menggunakan standar aplikasi dan juga dengan menggunakan protokol berbayar (MT, MWB, MWBS) dengan menggunakan aplikasi khususnya. Protokol proprietary MT (MikroTik Terminal) memungkinkan koneksi terminal ke sebuah command line remote embedded system, serta memungkinkan protokol SSH atau Telnet. Proprietary protocol MWB (MikroTik WinBox) digunakan untuk melakukan pengelolaan embedded system melalui antarmuka pengguna grafis. Protokol MWBS (MikroTik WinBox Secure) adalah varian aman dari MWB. Hubungan antara protokol MWB dan MWBS memiliki kesamaan dengan protokol HTTP dan HTTPS. Pilihan terakhir adalah menggunakan protokol komunikasi MAPI (MikroTik API), yang mengakses antarmuka aplikasi sistem operasi MikroTik RouterOS. MAPI merupakan protokol komunikasi

berpemilik, namun dengan dokumentasi yang dapat diakses oleh publik. Protokol ini dirancang untuk manajemen jarak jauh embedded system.

2. METODOLOGI

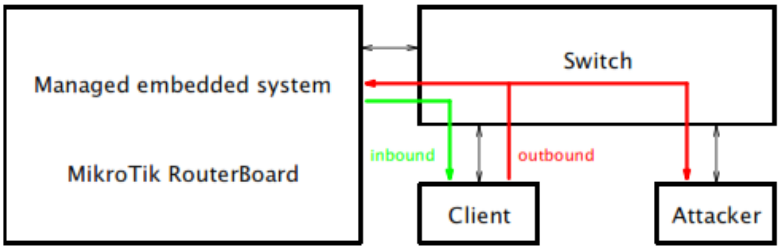
Pada penelitian ini dengan memanfaatkan bug konseptual yang ada pada protokol sistem operasi MikroTik RouterOS. Salah satu dari protokol untuk pengelolaan jarak jauh dari sistem ini untuk transfer data aplikasi adalah protokol komunikasi sesi Proprietary MTP. Di atas protokol ini dapat mengoperasikan protokol aplikasi MT, MWB, atau MWBS. Ketika aplikasi WinBox bekerja dengan pengetahuan tentang alamat IP dari embedded system yang dikelola, protokol aplikasi MWB atau MWBS bekerja secara langsung melalui protokol TCP. Jika tidak, protokol komunikasi MTP bekerja melalui protokol transport UDP, yang bekerja melalui protokol jaringan versi IPv4. Karena harus dimungkinkan untuk berkomunikasi dengan embedded system yang tidak memiliki alamat IP, protokol UDP digunakan dalam mode broadcast jaringan. Oleh karena itu, pengalamatan perangkat perlu dilakukan pada model referensi ISO / OSI yang lebih tinggi. Layanan yang perlu berada di jaringan dan lapisan transport dipecah, tidak dalam mode ini yang disediakan oleh protokol yang digunakan (Jones, 2002). Bisa juga dikatakan bahwa protokol MTP yang digunakan menggantikan protokol IPv4 dan UDP. Alasan penggunaan protokol MTP adalah protokol IPv4 dan UDP tidak digunakan untuk menjalankan semua fungsinya, yang diharapkan dari protokol pada lapisan tertentu. Oleh karena itu, fungsi jaringan dan lapisan transport dipecahkan pada lapisan sesi dalam protokol MTP (Pengalamatan perangkat dan transmisi data yang andal). Komunikasi yang diuraikan ditunjukkan dalam model referensi ISO / OSI (Braden, 1989) pada Gambar. 1

Keuntungan substansial dari solusi yang disarankan adalah bahwa *embedded system* yang dipilih dapat berkomunikasi walaupun tidak ada alamat IP yang diberikan dengan benar ke setiap perangkat atau alamat IP yang ditetapkan tidak diketahui. Identifikasi *embedded system* dan *client* PC direalisasikan pada layer referensi ISO / OSI yang lebih tinggi dan didasarkan pada alamat MAC dari interface jaringan dimana komunikasi dilakukan. Sedangkan data komunikasi dikirim sebagai siaran UDP, hanya dapat dikomunikasikan dengan embedded system yang berada pada segmen jaringan yang sama. Paket IP yang ada di sisi klien (PC dengan aplikasi klien) ditujukan ke lapisan jaringan ke alamat IP broadcast 255.255.255.255 dan frame pada lapisan data link ditujukan ke alamat link broadcast ff: ff: ff : Ff: ff (jílek, 2011). Semua frame pada sisi client dikirim ke semua komponen jaringan di segmen jaringan tertentu. Akibatnya semua komunikasi ke arah komunikasi dari client ke perangkat yang dikelola mudah diakses untuk semua perangkat lain yang ada di segmen jaringan yang sama. Paket IP pada sisi *managed embedded system* juga ditujukan untuk menyiarkan alamat IP 255.255.255.25, namun pada lapisan data link tidak digunakan alamat link broadcast ff: ff: ff: ff: ff, namun secara langsung menggunakan alamat MAC dari antarmuka jaringan penerima yang diinginkan. Data yang dikirim oleh embedded system hanya dikirim ke klien, yang ditunjukkan pada Gambar 2.

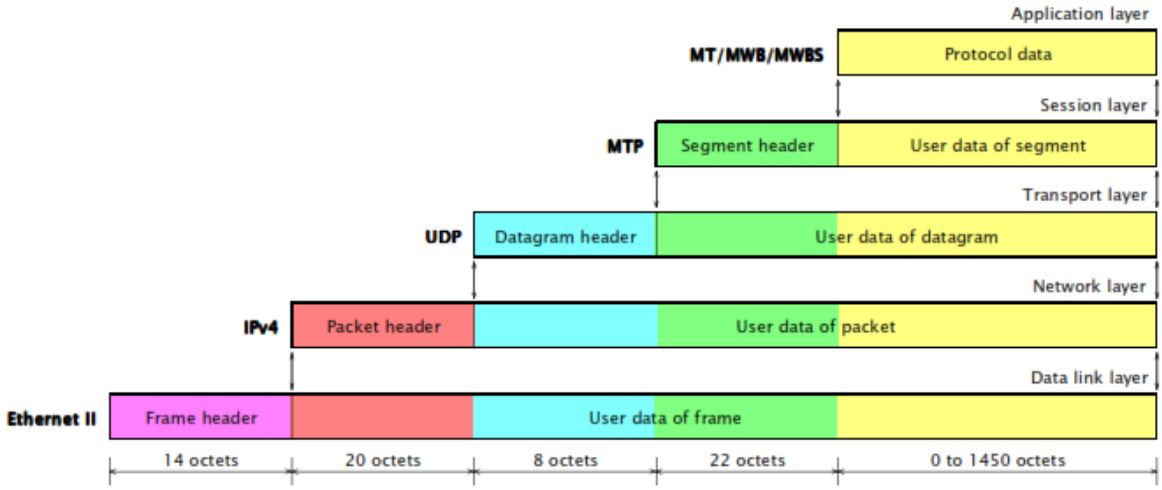


Gambar 1. Protokol komunkasi ISO/OSI

Metode pengiriman data yang sama juga bisa digunakan pada klien sisi. Dalam kasus ini tidak dapat digunakan soket standar untuk protokol UDP, namun harus menggunakan tipe soket mentah. Desainer aplikasi klien pada sisi PC jelas ingin menghindari penggunaan jenis soket ini, dan karena itu mereka menggunakan siaran standar melalui protokol UDP. Struktur dari protocol UDP ditunjukkan pada Gambar 3.



Gambar 2. Pengiriman data oleh embedded system



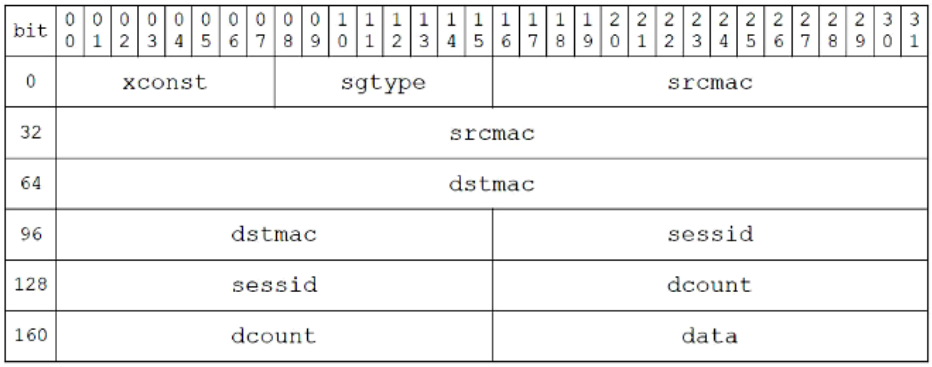
Gambar 3. Struktur frame protocol UDP

2.1 Protokol komunikasi MTP

Ini adalah protokol komunikasi tanpa dokumentasi yang tersedia untuk umum, yang dirancang oleh MikroTik. Protokol komunikasi ini beroperasi melalui protokol UDP dan menyediakan dua tugas utama yang tidak disediakan oleh protokol yang digunakan pada jaringan dan lapisan transport. Tugas pertama adalah pengalamatan perangkat yang tidak bekerja pada lapisan jaringan karena penggunaan siaran UDP. Tugas kedua adalah memastikan transfer data yang andal (pengiriman data dengan urutan yang benar, transmisi ulang jika terjadi kehilangan datagram UDP). Berdasarkan analisis komunikasi antara aplikasi client dan *managed embedded system* dibuat deskripsi masing-masing oktet dan dibuat deskripsi diagram keadaan perilaku protokol ini, dan ditunjukkan pada Gambar 4.

Dengan analisis komunikasi, diketahui bahwa semua segmen MTP mengandung oktet pertama (xconst) konstan 0x01 yang tidak diketahui artinya. Pada oktet kedua (sgtype) adalah tipe segmen MTP dan pada ketiga sampai kedelapan oktet (srcmac) adalah alamat MAC dari pengirim segmen MTP dan pada oktet kesembilan sampai keempat belas (dstmac) adalah alamat MAC dari sebuah Penerima segmen MTP Pada 15 sampai 18 oktet (sessid) adalah connection identifier, yang mencakup kode protokol yang digunakan pada lapisan komunikasi yang lebih tinggi. Pada 19 sampai 22 oktet (dcount) adalah sejumlah oktet pengguna yang sudah terkirim / diterima. Komunikasi segmen MTP tipe OPEN (0x00), embedded system yang dikelola harus mengetahuinya dengan mengirimkan segmen MTP tipe ACK (0x02). Setelah diterima, boleh dikirim segmen MTP tipe DATA (0x01), yang harus selalu dikenali segmen MTP tipe ACK (0x02) kepada pengirimnya. Jika tipe segmen MTP DATA tidak diakui dalam waktu tertentu, maka akan dikirim ulang oleh pengirim. Sambungan tersebut menutup segmen MTP CLOSE (0xFF) dan harus dikenali sama.

Segmen MTP Memastikan urutan data pengiriman yang benar dilakukan hanya pada penghitung data yang dikirim atau diterima. Setiap tipe segmen MTP DATA berisi hitungan (dcount) oktet dimana oktet pengguna di segmen MTP mengikuti. Oktet ini diakui oleh segmen MTP tipe ACK, dengan sejumlah (dcount) dari tipe segmen MTP DATA dan hitungan oktet pengguna yang ditransfer.



Gambar 4. Komunikasi antar client dan managed embedded system

2.2 Protokol Komunikasi MT

Protokol komunikasi memungkinkan otentikasi klien ke *embedded system* dan transmisi string teks untuk memperbarui jendela terminal. Seorang klien pertama kali mengirimkan informasi tentang inisiasi proses otentikasi dengan permintaan untuk mengirim string acak. Embedded system yang diakses berdasarkan panggilan ini mengirimkan string acak sepanjang 16 oktet. String ini dikaitkan dengan karakter nol dan dengan password login. Dari string ini kemudian dihitung hash dengan menggunakan algoritma MD5. Hash ini bersama dengan data lain dikirim kembali ke embedded system yang diakses. Setelah menerima data, sistem yang diakses, mengirimkan informasi tentang akhir proses otentikasi. Dalam kasus otorisasi yang berhasil kemudian ditransmisikan data antara jendela terminal dan embedded system yang diakses secara remote. Jika tidak, hanya pesan tentang otorisasi yang tidak berhasil dikirim dan pada tingkat protokol MTP, dan komunikasi dihentikan.

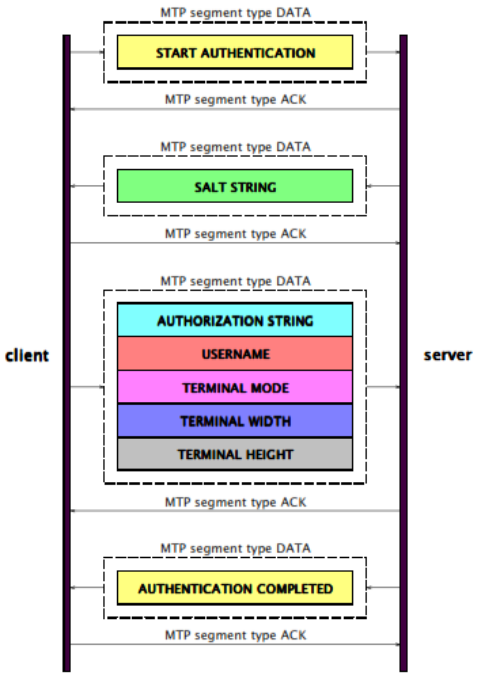
3. HASIL DAN PEMBAHASAN

Hasil yang didapatkan penulis adalah menyampaikan beberapa bug dalam desain potokol MTP dan jenis serangan yang dapat digunakan, antara lain :

3.1 BUG DALAM DESAIN PROTOKOL MTP

Kelemahan yang vital adalah bahwa isi komunikasi klien ke embedded system yang dikelola tersedia untuk semua perangkat jaringan lain yang terhubung ke segmen jaringan yang sama dan protokol komunikasi tidak dienkripsi. Hanya dari komunikasi ke arah ini ada perangkat jaringan lain pada segmen jaringan yang tersedia semua informasi untuk rekonstruksi semua parameter status protokol komunikasi MTP. Setiap perangkat dalam segmen jaringan dapat menggunakan informasi ini untuk menghasilkan segmen MTP yang akan diterima oleh embedded system atau aplikasi klien, seperti segmen MTP ini dikirim oleh perangkat dengan komunikasi telah dibuat. Masalahnya di sini adalah bahwa setelah otorisasi klien yang sukses ke embedded system yang dikelola, tidak digunakan fitur keamanan untuk memastikan keaslian data yang dipancarkan.

Masalah lain adalah bahwa layanan dalam embedded system yang dikelola tidak memeriksa sumber alamat MAC dalam frame yang diterima pada lapisan data link. Layanan ini bahkan menerima frame dimana alamat MAC sumber pengirim berubah selama komunikasi berlangsung. Demikian pula, layanan tidak memeriksa nomor port sumber dalam datagram UDP yang diterima. Oleh karena itu mereka bahkan menerima datagram UDP, di mana nomor port UDP sumber berubah selama komunikasi.



Gambar 5. Proses Autentikasi

Ethernet frame *client → managed embedded system*
(The last Ethernet frame that was sent by client to managed embedded system before an attack.)

Link layer: **Ethernet II**, Src: 90:e6:ba:46:d3:cd, Dst: ff:ff:ff:ff:ff:ff
Network layer: **Internet Protocol Version 4**, Src: 10.0.200.245, Dst: 255.255.255.255
Transport layer: **User Datagram Protocol**, Src Port: 20561, Dst Port: 20561
Session layer: **MikroTik Transmission Protocol**: Data, Src: 90:e6:ba:46:d3:cd, Dst: 00:0c:42:5c:61:32
Application layer: **MikroTik Terminal**: User: admin, ...

```
0000  ff ff ff ff ff ff 90 e6 ba 46 d3 cd 08 00 45 00   .........F....E.
0010  00 7e 25 a8 00 00 80 11 41 d2 0a 00 c8 f5 ff ff   .~%.....A.......
0020  ff ff 50 51 50 51 00 6a 07 87 01 01 90 e6 ba 46   ..PQPQ.j.......F
0030  d3 cd 00 0c 42 5c 61 32 0a 9c 00 15 00 00 00 09   ....B\a2........
0040  56 34 12 ff 02 00 00 00 11 00 0f 68 82 27 4f 21   V4.........h.'O!
0050  cb cf 1f aa 4d ba dc 16 f6 8f 56 34 12 ff 03 00   ....M.....V4....
0060  00 00 05 61 64 6d 69 6e 56 34 12 ff 04 00 00 00   ...adminV4......
0070  05 6c 69 6e 75 78 56 34 12 ff 05 00 00 00 02 50   .linuxV4.......P
0080  00 56 34 12 ff 06 00 00 00 02 18 00               .V4.........
```

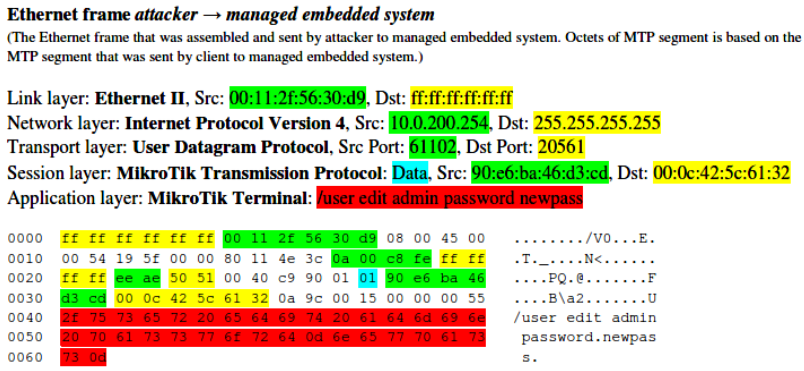
Gambar 6. protokol system embedded sebelum serangan

3.2 DESKRIPSI SERANGAN PROTOKOL MTP

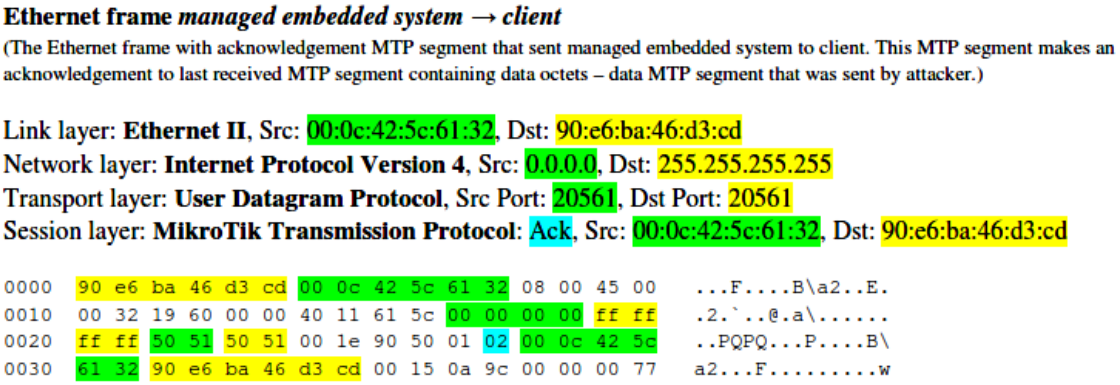
Serangan terhadap bug dalam protokol MTP ditunjukkan dalam situasi di mana ia menggunakan aplikasi terminal MikroTik untuk komunikasi dengan embedded system yang dikelola. Penyerang mendengarkan pada port UDP 20561 yaitu port yang mendengarkan layanan jaringan yang sesuai pada embedded system. Jika penyerang menyadap di port ini pada antarmuka jaringannya, data UDP broadcast, pertama-tama periksa apakah menggunakan protokol MTP pada lapisan aplikasi dengan memeriksa bidang sesspr (oktet 3 dan 4 bidang sessid) untuk nilai 0x0015. Jika positif kemudian mengevaluasi otorisasi klien ke embedded system yang dikelola. Jika klien sudah berwenang (misalnya, dapat diverifikasi bahwa klien mengirim segmen ACK MPS yang tetap), penyerang dapat melanjutkan perakitan dan mengirim tipe segmen MTP palsu. Jika tidak, harus menunggu penyelesaian otorisasi klien sampai pada embedded system menerimanya. Pada titik ini, penyerang memiliki semua data yang diperlukan untuk mengumpulkan segmen segmen MTP DATA, yang akan dikelola oleh embedded system yang diterima.

Penyerang dalam proses segmentasi tipe MTP DATA mengumpulkan oktet 1 sampai 18 dengan nilai sama seperti pada oktet ini pada tipe segmen MTP yang terakhir ditangkap, yang dirakit dan dikirim oleh aplikasi klien. Nilai bidang dcount tipe segmen MTP DATA meningkat dengan hitungan oktabel pengguna yang berada pada tipe segmen MTP yang terakhir ditangkap.

Pada oktet pengguna (oktet ke-23 dan ke atas) bisa dimasukkan perintah apapun yang ingin di eksekusi pada embedded system yang tersimpan. Dengan demikian yang system embedded yang diakses dapat mengubah kata sandi pengguna mana pun, atau buat pengguna baru dengan hak penuh. Akun pengguna ini bisa kemudian digunakan untuk komunikasi lebih lanjut dengan embedded system.



Gambar 7. Penyadapan data pada protocol MTP



Gambar 8. Perubahan data informasi dari system embedded

Diberikan bug yang terdaftar, segmen segmen MTP ini DATA dari penyerang diterima oleh sistem tertanam yang dikelola dan string teks yang terkandung dalam oktet pengguna dilewatkan ke baris perintah. Jika klien mengirim segmen MTP tipe DATA, maka tidak akan dikenali oleh embedded system. Oleh karena itu client mentransmisikan ulang segmen MTP berkali-kali untuk melelahkan jumlah usaha pengiriman jenis segmen MTP DATA. Setelah melebihi batas waktu terakhir, aplikasi klien dihentikan. Satu-satunya kemungkinan untuk menghindari penghentian aplikasi klien cukup cepat menghasilkan sekitar satu oktet lebih banyak data daripada penyerang.

4. KESIMPULAN

Penulis menjelaskan risiko keamanan yang ada saat protokol proprietary untuk pengelolaan remote embedded system yang menjalankan sistem operasi MikroTik RouterOS digunakan. Serangan terhadap keaslian data yang ditransfer ditunjukkan pada jaringan nyata. Dari perangkat mana pun yang berada dalam segmen jaringan yang sama seperti data embedded system yang dikelola dapat dikirim ke embedded system tanpa kemungkinan embedded system dapat mengenali bahwa data ini tidak berasal dari klien dengan siapa komunikasi tersebut terbentuk.

Karena penyebaran embedded system secara massal dengan sistem operasi ini, terutama oleh penyedia internet, ini merupakan masalah yang cukup besar. Administrator embedded system ini harus menghindari penggunaan protokol komunikasi ini di segmen jaringan karena terhubung dan berpotensi menyerang. Untuk mengakses command line dari embedded system hanya protokol SSH yang harus digunakan (Held, 2000).

DAFTAR PUSTAKA

Braden, R. (1989). *RFC1122 – Requirements for Internet Hosts – Communication Layers*. IETF, Marina del Rey.

Held, G. (2000). *Managing TCP/IP networks: Techniques,tools and security*. Wiley, 1st edition, 352 pages, ISBN: 0471800031

Jílek, T. (2011). *Remote control of remote communication board of mobile robot*. Brno University of Technology, Faculty of Electrical Engineering and Communication, Brno. 77 pages.

Jones, M. T. (2002). *TCP/IP Application layer protocols for embedded systems*. Charles River Media, 1st edition, 460 pages, ISBN: 1584502479.

MikroTik (2011). *MikroTik Wiki*. MikroTik, Latvia.