الجامعة الافتراضية السورية

SYRIAN VIRTUAL UNIVERSITY

# خدمات الشبكة

الدكتور غسان سابا

ISSN: 2617-989X

Books

# خدمات الشبكة

الدكتور غسان سابا

من منشورات الجامعة الافتراضية السورية

الجمهورية العربية السورية 2018



# Network Services

Ghassan Saba

Publications of the Syrian Virtual University (SVU)

Syrian Arab Republic, 2018



ISSN: 2617-989X

# الفهرس



ISSN: 2617-989X

ISSN: 2617-989X

# التطبيقات الشبكية
# Network Applications

ISSN: 2617-989X

لا توجد فعلياً حاجة للشبكات أو لبروتوكولات الشبكات بدون التطبيقات الشبكية. فمنذ بدايات الإنترنت، انتشرت مجموعة من التطبيقات الشبكية التي شكلت المحرك الأساسي لنجاح الإنترنت وحفزت الناس سواء في البيوت أو في المدارس أو في المؤسسات الحكومية والخاصة على جعل الإنترنت جزءاً لا يتجزأ من مهامهم اليومية.

بدأت الإنترنت بتطبيقات شبكية نصية لنقل البريد الإلكتروني والنفاذ البعيد إلى الحواسيب ونقل الملفات. بعد ذلك انتشر بروتوكول الوب في منتصف التسعينيات والذي اعتبر نقطة انطلاق الإنترنت لما يملكه من مقومات الملاحة والبحث والتجارة الإلكترونية.

كما انتشر أيضاً، مع نهاية القرن الماضي، تطبيقان معروفان وهما الدردشة الفوريةInstant Messaging ومشاركة الملفات من نوع P2P. شهدنا، مع بداية عام 2000، انتشار تطبيقات نقل الصوت عبر الإنترنت VoIP وتطبيقات المؤتمرات المرئية عبر الإنترنت مثل سكايب وتطبيقات توزيع الفيديو الذي يولده المستثمر مثل Youtube والفيديو حسب الطلب مثل Netflix. انتشرت أيضاً في هذه الفترة الألعاب على الخط متعددة الأطراف مثل Second Life. أما حديثاً، فقد شهدنا انتشار جيل جديد من تطبيقات الشبكات الاجتماعية مثل الفيسبوك والتويتر واللذين أوجدا شبكة من الأشخاص فوق شبكة الإنترنت المكونة من المسيرات Routers والوصلات.

## 1. مبادئ التطبيقات الشبكية

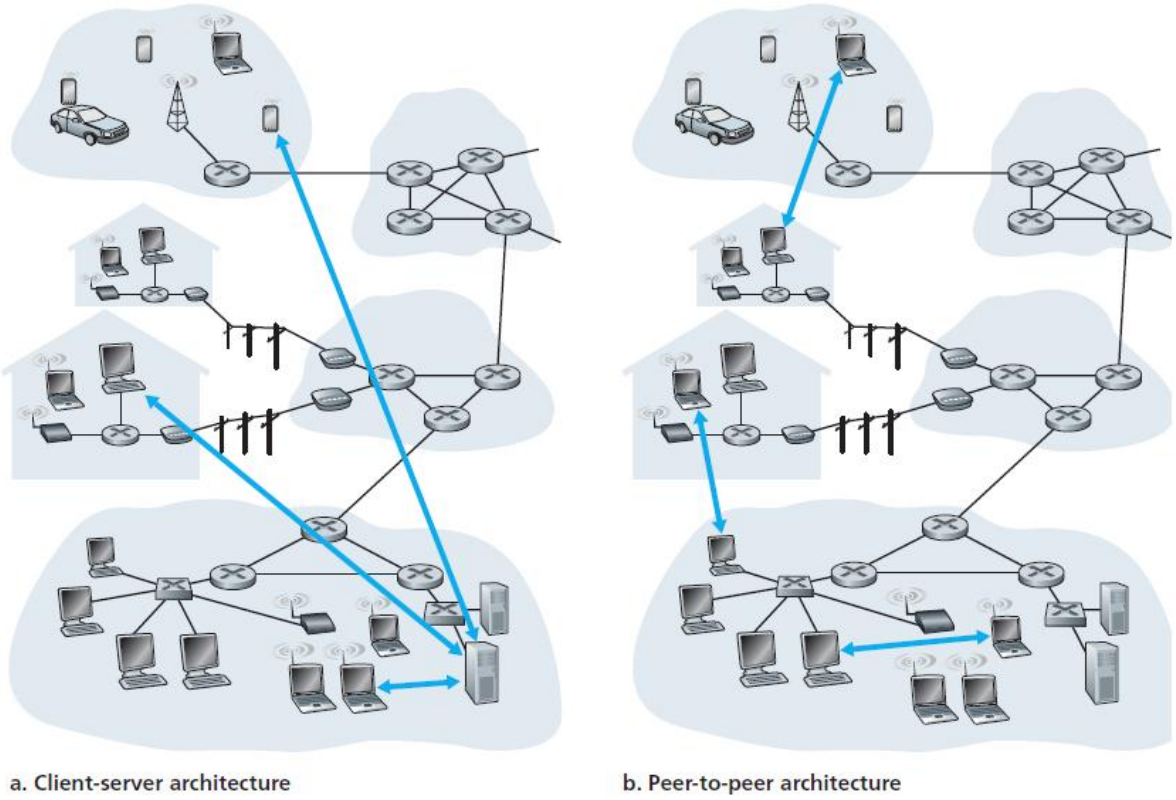
| | |
|---|---|
| تعتبر كتابة برامج لتطوير تطبيقات شبكية تعمل على أنظمة نهائية End Systems (ES) مختلفة وتتخاطب مع بعضها البعض عبر الشبكة هي أساس التطبيقات الشبكية. فضمن تطبيق الوب، على سبيل المثال، يوجد برنامجان مختلفان يتخاطبان مع بعضهما: متصفح الإنترنت الذي يعمل على حاسب المستثمر (سنطلق عليه اسم المضيف Host لأنه يمكن أن يكون حاسب شخصي أو محمول أو هاتف ذكي أو هاتف لوحي) وبرنامج مخدم الوب الذي يعمل على مضيف مخدم الوب. بينما يوجد، في شبكات مشاركة الملفات من نوع P2P، برنامج ضمن كل مضيف يشارك في عملية نقل الملفات أي تكون البرامج ضمن المضيفين هنا متماثلة. | **سنستخدم من الآن فصاعداً مصطلح نظام نهائي ES أو مضيف للدلالة على أي حساب أو مخدم أو هاتف ذكي قادر على تشغيل أي تطبيق شبكي** |

## 2. بنيان التطبيقات الشبكية Network Applications Architecture

يختلف بنيان التطبيق الشبكي عن بنيان الشبكة نفسها الذي يتكون من مجموعة من الطبقات تزود كل واحدة مجموعة من الخدمات للطبقة الأعلى منها مباشرةً. بينما يقوم مصمم التطبيق بتصميم بنيان التطبيق الذي يعمل بين مجموعة من الأنظمة النهائية. هنا يحتاج المصمم إلى الاختيار بين أحد البنيانين المعروفين ضمن التطبيقات الشبكية وهما الزبون مخدم Client-server أو الند للند P2P.

يوجد، في بنيان زبون–مخدم، مخدم يعمل طوال الوقت يتميز بكونه قادراً على تخديم طلبات قادمة من عدة حواسب مضيفة، تدعى زبائن، في الوقت نفسه. لاحظ أن التخاطب يكون بين زبون ومخدم ولا تستطيع الزبائن التخاطب مع بعضها. يجب أن يملك المخدم هنا عنوان IP ثابت ومعروف حتى تستطيع الزبائن التخاطب معه. يبين الجزء (a) من الشكل التالي التطبيقات من نوع زبون–مخدم. عادةً، لا يستطيع مخدم واحد متابعة العدد المتزايد من الطلبات التي يولدها الزبائن وخاصة في التطبيقات المشهورة مثل تطبيقات الفيسبوك والتويتر، لذلك يجري هنا بناء مركز معطيات Data Center أو عدة مراكز لاستضافة مجموعة كبيرة من المخدمات تعمل ضمن بيئة مخدمات افتراضية. فشركة Google مثلاً تملك حوالي 50 مركز معطيات لاستضافة أكثر من مليون مخدم موزعين حول العالم تعالج طلبات البحث واليوتيوب وبريد Gmail وخرائط غوغل وغيرها.

في شبكات من نوع P2P، لا يوجد اعتماد على مخدمات مخصصة موجودة ضمن مراكز معطيات. تستخدم هنا التطبيقات الاتصال المباشر بين العقد، التي تدعى Peers. يمكن لهذه العقد أن تكون حواسب شخصية أو محمولة موجودة في أي مكان. تجدر الإشارة هنا إلى وجود العديد من التطبيقات الشبكية من نوع P2P والتي تولد حجماً هائلاً من حركة المرور على الإنترنت. تشمل هذه التطبيقات مشاركة الملفات (مثل بت تورنت) ومسرعات التحميل (مثل Xunlei) والهاتف عن طريق الإنترنت (مثل سكايب) وغيرها. يبين الجزء (b) من الشكل التالي بنيان P2P.




الشكل 1– أنواع بنيان التطبيقات الشبكية

## 3. الاتصال بين الإجراءات Process communications

يتألف البرنامج عادةً من مجموعة من الإجراءات التي تتخاطب مع بعضها. تتخاطب الإجراءات ضمن المضيف نفسه باستخدام الاتصال بين الإجراءات الذي يتيحه نظام التشغيل. ما يهمنا هنا، هو طريقة التخاطب بين إجراءات تابعة لأجهزة مختلفة.

يجري التخاطب بين إجراءات تابعة لأنظمة مختلفة عن طريق تبادل الرسائل عبر الشبكات الحاسوبية.

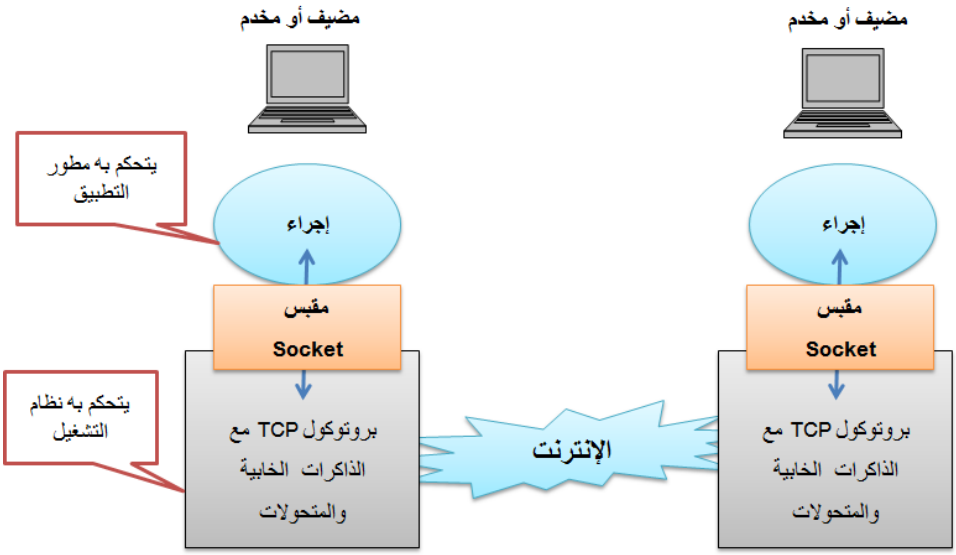
### 1.3. إجراءات الزبون والمخدم

يتألف أي تطبيق شبكي هنا من زوج من الإجراءات تتبادلان الرسائل مع بعضهما عبر الشبكة. فمثلاً، في تطبيق الوب، يتبادل إجراء ضمن متصفح الإنترنت الرسائل مع إجراء ضمن مخدم الوب. يمكن تعريف إجراء الزبون وإجراء المخدم على الشكل التالي:

في سياق جلسة اتصال بين إجراءين، يدعى الإجراء الذي يبادر إلى الاتصال بالإجراء الزبون والإجراء الذي ينتظر الاتصال لبدء الجلسة بالإجراء المخدم.

### 2.3.واجهة التخاطب بين الإجراء والشبكة

يجري تبادل الرسائل بين إجراءي المخدم والزبون عبر الشبكة باستخدام واجهة تخاطب برمجية تدعى المقبس Socket. يبين الشكل التالي آلية التخاطب بين الإجراءات باستخدام بروتوكول TCP.




الشكل 2– التخاطب بين الإجراءات عن طريق الإنترنت باستخدام المقابس

كما هو موضح بالشكل السابق، المقبس هو صلة الاتصال بين التطبيق وبين طبقة النقل ضمن المضيف. يطلق عليه أيضاً اسم واجهة برمجة التطبيقات Application Programming Interface (API). يملك مطور التطبيقات تحكماً كاملاً من طرف طبقة التطبيقات للمقبس بينما يملك تحكماً ضئيلاً من طرف طبقة النقل للمقبس. التحكم الوحيد الذي يستطيع المطور فرضه على طرف طبقة النقل للمقبس هو (1) اختيار طبقة النقل و(2) تثبيت بعض وسطاء النقل مثل حجم الخابية Buffer size وطول مقطع الشبكة الأعظمي Max. MTU. بعد اختيار طبقة النقل، يستطيع المطور بناء التطبيق باستخدام خدمات طبقة النقل التي جرى اختيارها.

### 3.3. عنونة الإجراءات Addressing Processes

حتى يستطيع إجراء يعمل ضمن مضيف ما التخاطب مع إجراء آخر يعمل ضمن مضيف آخر، يجب على الإجراء الوجهة أن يملك عنواناً. يتألف العنوان من جزأين: (1) عنوان للمضيف و(2) معرف يحدد الإجراء الوجهة ضمن المضيف الوجهة.

يجري تعريف المضيف، ضمن الإنترنت، باستخدام عنوان IPv4 المكون من 4 بايتات أو IPv6 المكون من 16 بايتاً بينما يجري تعريف الإجراء (أو المقبس الوجهة) ضمن المضيف باستخدام رقم البوابة Port Number الممتد على بايتين. نحتاج إلى تعريف رقم البوابة للمقبس لأنه يمكن لمضيف ما أن يحوي عدة تطبيقات شبكية في الوقت نفسه. جرى تخصيص التطبيقات المعروفة برقم بوابات معروفة أيضاً، فخصص مخدم الوب بالبوابة 80 وخصص مخدم البريد الالكتروني بالبوابة 25.

## 4. خدمات النقل المتوافرة للتطبيقات

نذكّر هنا أن التطبيق الذي يعمل عند المرسل يدفع الرسائل عبر المقبس حيث تقوم طبقة النقل عنده بنقل الرسائل لمقبس الإجراء الوجهة.

توفر الإنترنت خيارين للخدمات التي يمكن أن تقدمها طبقة النقل، وهما: TCP أو UDP. يقدم كل منهما مجموعة من الخدمات إلى التطبيق.

خدمات TCP

عند اختيار طبقة النقل TCP ليعمل فوقها التطبيق فإننا نتوقع الحصول على الخدمات التالية:

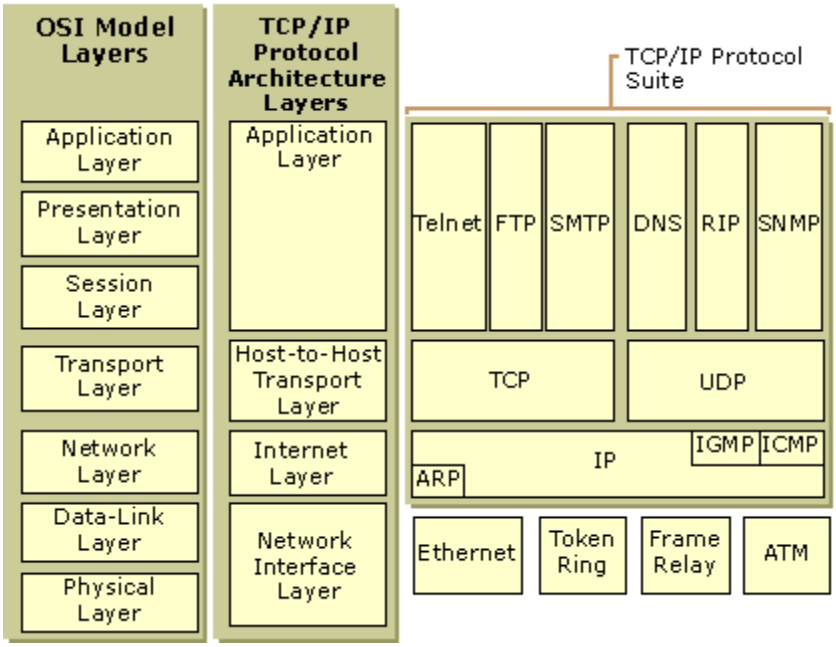
- خدمه ذات ارتباط Connection-oriented service.
- خدمة نقل معطيات موثوقة بدون أخطاء.
- خدمة التحكم بالتدفق وبالاختناقات.

خدمات UDP

تقدم طبقة UDP مجموعة ضئيلة من الخدمات فهي عديمة الارتباط Connectionless وغير موثوقة ولا تضمن ترتيب الطرود المستقبلة. لا تعالج UDP الاختناقات مع الشبكة ولا تتحكم بتدفق الطرود مع الوجهة.

## 5. لمحة عن بروتوكولات TCP/IP

جرى تطوير بروتوكولات TCP/IP قبل النموذج Open Systems Interconnection (OSI). أدى ذلك إلى كون طبقات TCP/IP غير مطابقة تماماً لطبقات OSI. تتألف بروتوكولات TCP/IP من خمس طبقات، وهم: فيزيائية ووصلة معطيات وشبكة ونقل وتطبيقات.




الشكل 3– طبقات TCP/IP مقارنة مع نموذج OSI

### 1.5. طبقات TCP/IP

#### 1.1.5. الطبقة الفيزيائية وطبقة وصلة المعطيات (النفاذ إلى الشبكة)

مع أن TCP/IP لا تعرِّف أي بروتوكول ينتمي إلى هاتين الطبقتين إلى أنها تدعم جميع البروتوكولات المعيارية والخاصة. كما يمكن أن تكون الشبكة حسب TCP/IP محلية LAN أو واسعة WAN.

#### 2.1.5. طبقة الشبكة Internet Layer

تطلق TCP/IP على هذه الطبقة اسم طبقة ترابط الشبكات Internet Layer (IP). يستخدم بروتوكول IP ثلاثة بروتوكولات: ARP, ICMP, and IGMP.

بروتوكول IP

يشكل بروتوكول IP وسيلة النقل التي تستخدمها بروتوكولات TCP/IP. يؤمن IP خدمة نقل غير موثوقة وعديمة الارتباط لا تضمن اكتشاف الأخطاء أو ترتيب الطرود.

تدعى الطرود التي يستخدمها بروتوكول IP ببرقيات المعطيات Datagrams. يجري نقل كل برقية معطيات نقلاً مستقلاً عن بعضها. يمكن أن تسلك برقيات المعطيات طرقاً مختلفة الأمر الذي يؤدي إلى وصولها إلى الوجهة بغير ترتيب إرسالها أو تكرار وصول برقية معطيات نفسها عدة مرات.

بروتوكول حل العناوين Address Resolution Protocol (ARP)

يستخدم بروتوكول ARP لمقابلة عنوان IP مع عنوان فيزيائي. يجري عادةً استخدام العناوين الفيزيائية مع المحطات المربوطة إلى شبكة محلية. يُطبع عنوان بطاقة الشبكة Network Interface Card (NIC) عليها ويكون مميزاً لها. يفيد بروتوكول ARP في معرفة العنوان الفيزيائي للوجهة من خلال عنوان IP.

بروتوكول رسائل التحكم بالإنترنت Internet Control Message Protocol (ICMP)

يسمح هذا البروتوكول للمحطات وللمسيرات بإعلام المحطة المصدر عن بعض المشاكل التي تتعرض لها برقيات المعطيات.

بروتوكول رسائل مجموعات الإنترنت Internet Group Management Protocol (IGMP)

يسهل هذا البروتوكول إرسال برقيات المعطيات إلى مجموعة من المحطات في نفس الوقت.

## 2.5. طبقة النقل Host-to-host layer

تحوي طبقة النقل البروتوكولين: TCP وUDP. لاحظ أن بروتوكول IP هو من مضيف إلى مضيف Host-to-host protocol، أي أنه يستطيع توصيل المعطيات من جهاز فيزيائي إلى جهاز فيزيائي آخر. أما بروتوكولي طبقة النقل فهما مسؤولون عن توصيل الرسائل من إجراء إلى آخر Process-to-process protocols.

- بروتوكول User Datagram Protocol (UDP)
  - يعتبر بروتوكول UDP بسيطاً جداً لأنه يقوم بالعمليات التالية فقط:
  - يضيف عناوين البوابات Port addressing
  - يفحص بعض أنواع الأخطاء وبعض المعلومات المتعلقة بطول المعطيات القادمة أو المتجهة إلى الطبقات العليا.
- بروتوكول Transmission Control protocol (TCP)

يزود بروتوكول TCP خدمة نقل كاملة إلى طبقة التطبيقات. أي أنه يزود خدمة موثوقة ذات ارتباط الأمر الذي يقضي بإنشاء ارتباط (أو اتصال) بين الطرفين قبل البدء بإرسال الرسائل.

يقوم الطرف المرسل بتجزئة سيلان المعطيات Data Stream إلى وحدات صغيرة تدعى مقاطع Segments. يحمل كل مقطع رقماً تسلسلياً، يسمح بإعادة التجميع عند المستقبل، ورقم إقرار للمقطع المُستَقبل. يجري تغليف مقاطع TCP ضمن برقيات معطيات IP حتى يمكن إرسالها عبر الإنترنت. يقوم الطرف المستقبل بتجميع كل برقية معطيات مستقبلة ومن ثم إعادة ترتيبها حسب الأرقام التسلسلية.

### 3.5. طبقة التطبيقات Application layer

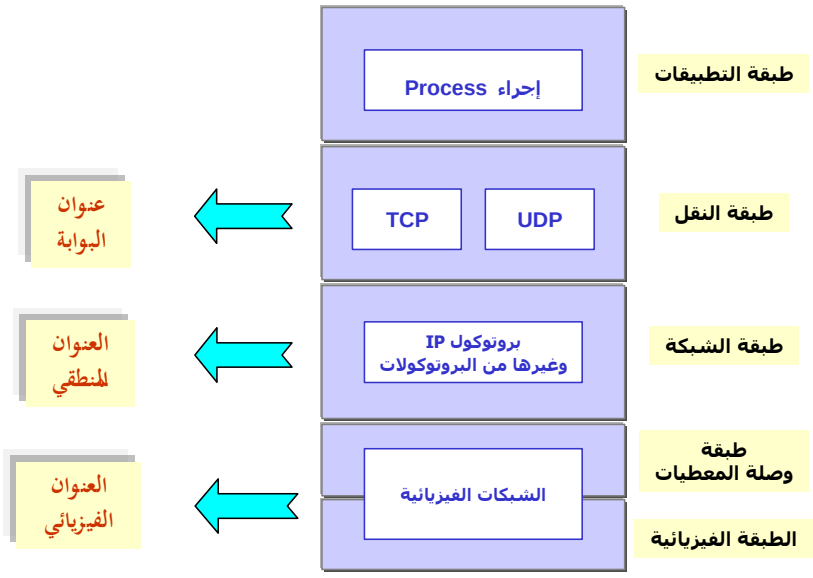
تكافئ طبقة التطبيقات ضمن TCP/IP كلاً من طبقة الجلسات والتقديم والتطبيقات ضمن النموذج OSI. لقد جرى تعريف مجموعة كبيرة من البروتوكولات التي تعمل ضمن هذه الطبقة سندرس معظمها لاحقاً.

## 6. العنونة Addressing

يجري عادةً استخدام ثلاثة مستويات من العنونة في أي ترابط شبكات يستخدم TCP/IP:

1. العنونة الفيزيائية Physical (link) address
2. العنونة المنطقية Logical (IP) address
3. عنوان البوابة Port address

ينتمي كل عنوان إلى طبقة محددة من طبقات TCP/IP كما هو موضح في الشكل التالي:




الشكل 4– العلاقة بين طبقات TCP/IP والعنونة

### العنوان الفيزيائي

العنوان الفيزيائي هو عنوان العقدة كما تعرفه الشبكة المحلية أو الواسعة. يجري استخدام العنوان الفيزيائي ضمن إطار طبقة وصلة المعطيات كما يعتبر أدنى مستوى عنونة يمكن استخدامه.

يتعلق طول وصيغة العنوان الفيزيائي بالشبكة المستخدمة، فتستخدم شبكة إيثرنت عنواناً فيزيائياً مطبوعاً على بطاقة الشبكة ومكوناً من 48 بتاً (أو ستة بايتات) بينما تستخدم شبكة LocalTalk لشركة Apple عنواناً ديناميكياً مكوناً من بايتاً واحداً يتغير كل مرة يعاد إقلاع الجهاز.

يمكن أن تكون العناوين الفيزيائية إما وحيدة الوجهة unicast أو متعددة الوجهات multicast أو معممة على الجميع Broadcast. تدعم بعض الشبكات جميع الأنواع السابقة من العناوين مثل شبكة إيثرنت بينما يمكن أن لا تدعم شبكة أخرى العنونة متعددة الوجهات مثلاً. يجري هنا محاكاة الإرسال متعدد الوجهات عن طريق إرسال نسخة من الإطار نفسه إلى كل وجهة على حدا.

مثال 1

يبين الشكل التالي مثالاً عن شبكة مؤلفة من مجموعة عقد، تريد العقدة ذات العنوان الفيزيائي 10 أن ترسل إطاراً إلى العقدة ذات العنوان الفيزيائي 87. يحوي هذا الإطار على مستوى طبقة وصلة المعطيات العناوين الفيزيائية ضمن الترويسة المستخدمة.


الشكل 5– العناوين الفيزيائية

Animation: تضيف العقدة رقم 10 عنوان الوجهة 87 وعنوانها واللاحقة T2 إلى المعطيات لتشكيل الإطار Frame وترسلها على وسيط النقل. يصل الإطار إلى جميع العقد. العقد رقم 87 تستقبله بينما ترفضه بقية العقد.

مثال 2

تستخدم غالبية الشبكات المحلية عناوين فيزيائية مكونة من 48 بتاً تكتب على شكل 12 رقم ست عشري حيث يجري فصل كل رقمين بنقطتين كما يلي:

08:02:04:3C:5D:77

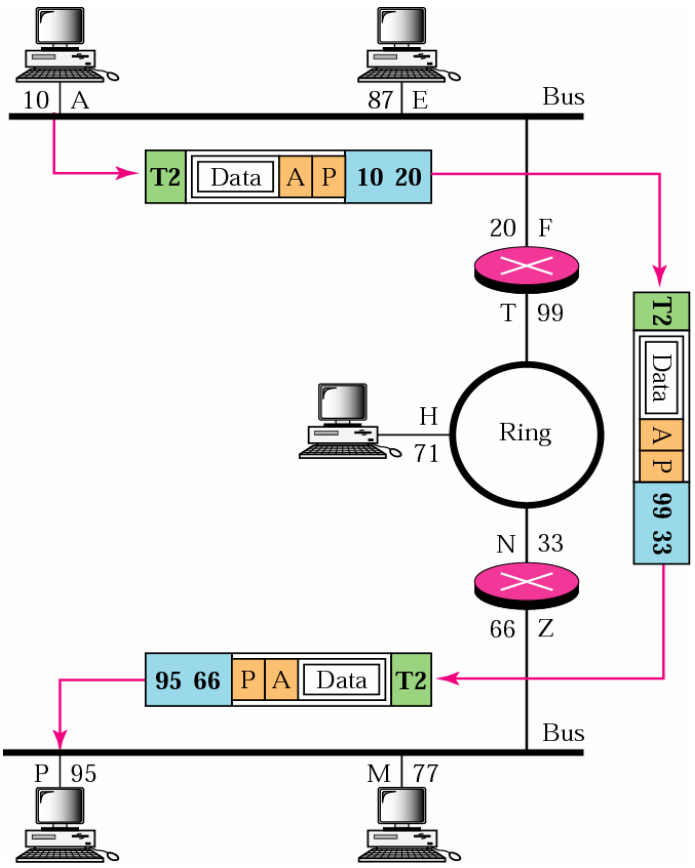
## العنوان المنطقي

نحتاج العناوين المنطقية لتحقيق خدمات الاتصال الشامل تحقيقاً مستقلاً عن الشبكات الفيزيائية المستخدمة، وذلك لأن العناوين الفيزيائية تصبح غير مناسبة في حال وجود شبكات فيزيائية مختلفة تستخدم عناوين فيزيائية مختلفة.

تستخدم الإنترنت 32 بتاً كعنوان منطقي أو عنوان IP يستطيع تعريف اتصال مضيف ما Host إلى الإنترنت تعريفاً وحيداً. لا يوجد مضيفَين موجودين على الإنترنت (أو قابلين للوصول إليهم) لهما نفس عنوان IP.
يمكن أن يكون العنوان المنطقي وحيد الوجهة أو متعدد الوجهات أو معمم على الجميع.

### مثال 3

يبين الشكل التالي كيف يمكن للعقدة ذات العنوان المنطقي A (أو عنوان الشبكة) والعنوان الفيزيائي 10 والموجودة على شبكة محلية، إرسال معطيات إلى العقدة ذات العنوان المنطقي P والعنوان الفيزيائي 95 والموجودة على شبكة محلية أخرى.




الشكل 6– عناوين IP

لاحظ أن العناوين الفيزيائية تتغير عند المرور من شبكة إلى أخرى بينما تحافظ عناوين الشبكة على قيمها من المرسل حتى الوصول إلى الوجهة النهائية.
Animation: يتحرك الطرد من A إلى المسير الأول ومنه إلى المسير الثاني ومنه إلى الوجهة النهائية P.
يضيف A الحقول A,P ثم 10,20 ثم T2 ويرسل الإطار. يحذف المسير الأول الحقول T2 ,P ,A ويصبح الطرد مكوناً من Data, A, P الذي يرسله إلى البوابة T حيث يضيف الحقول T2, 33, 99.

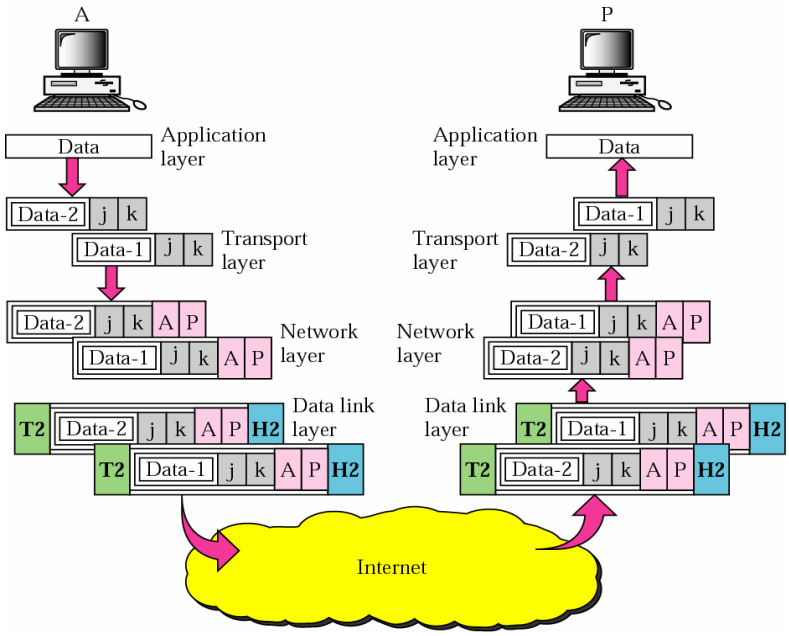
يقوم المسير الثاني بالعملية نفسها بعد أن يغير الحقول T2 ,33 ,99 بالحقول T2 ,66,95. ثم يصل الطرد إلى P.

## عنوان البوابة

ليست الغاية من تراسل المعطيات عبر الإنترنت هي إيصال المعطيات إلى الوجهة النهائية فقط وإنما يجب تأمين وصول المعطيات إلى الإجراء النهائي Final process وذلك لأنه يمكن لأي حاسوب تنفيذ عدة إجراءات في نفس الوقت. فيستطيع الحاسوب A الاتصال مع الحاسوب B باستخدام بروتوكول Telnet ومع الحاسوب C باستخدام بروتوكول نقل الملفات FTP. حتى يجري تحقيق الاتصال بين الإجراءات في نفس الوقت تحقيقاً جيداً فإننا نحتاج إلى طريقة تسمية أو عنونة الإجراءات المختلفة. يدعى هذا العنوان ضمن TCP/IP بعنوان البوابة Port address وهو مكوناً من 16 بتاً.

مثال 4

يبين الشكل التالي مثالاً عن اتصال على مستوى طبقة النقل. تحمل المعطيات القادمة من الطبقات العليا عنوان البوابات J و K (J هو عنوان الإجراء المرسل و K هو عنوان الإجراء المستقبل). بما أن طول المعطيات هو أكبر من الطول المسموح به ضمن طبقة الشبكة فيجري هنا تجزئة المعطيات إلى طردين (أو برقيتي معطيات) يحمل كل منهما عناوين البوابات J و K. يجري بعد ذلك إضافة عناوين IP على مستوى طبقة الشبكة وهي A و P. يمكن للطرود أن تسلك طرقاً متعددة للوصول إلى وجهتها النهائية الأمر الذي من شأنه وصول الطرود بغير الترتيب التي أرسلت به. عندما يصل الطردان إلى طبقة النقل لدى الوجهة فإنها تقوم بنزع ترويسة طبقة IP وبتجميع قطعتي المعطيات بغية توجيه الناتج إلى الطبقات العليا.




الشكل 7– عناوين البوابات

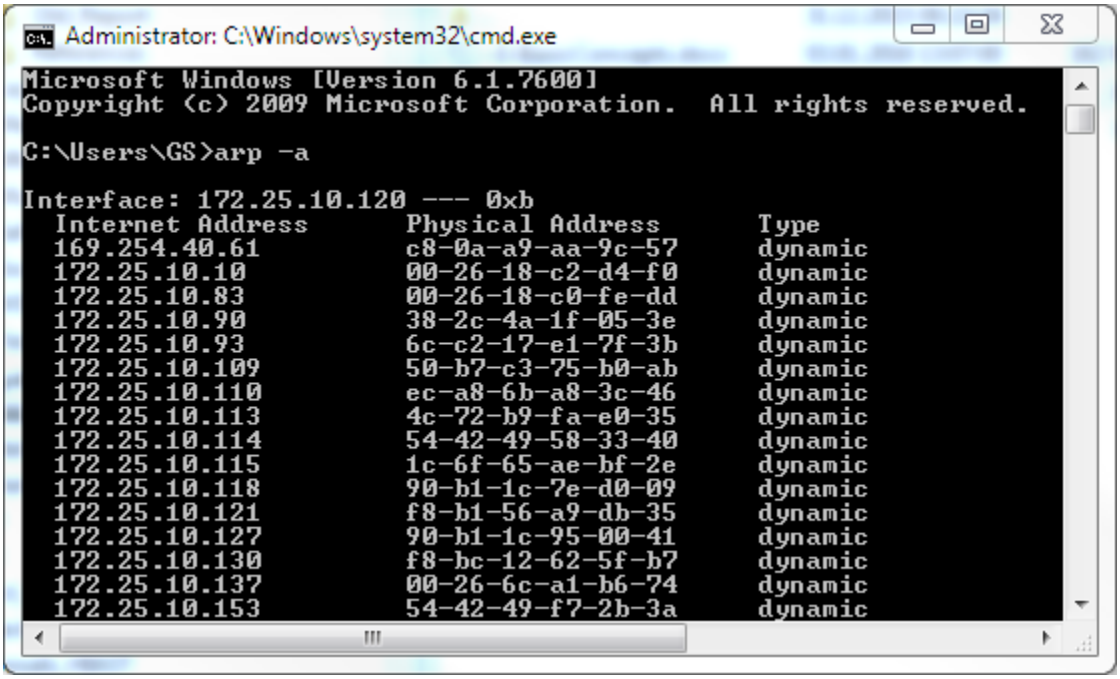
## 7. تمارين عملية

تمرين 1: التعرف على الأمرين arp –a و ipconfig من سطر الأوامر cmd

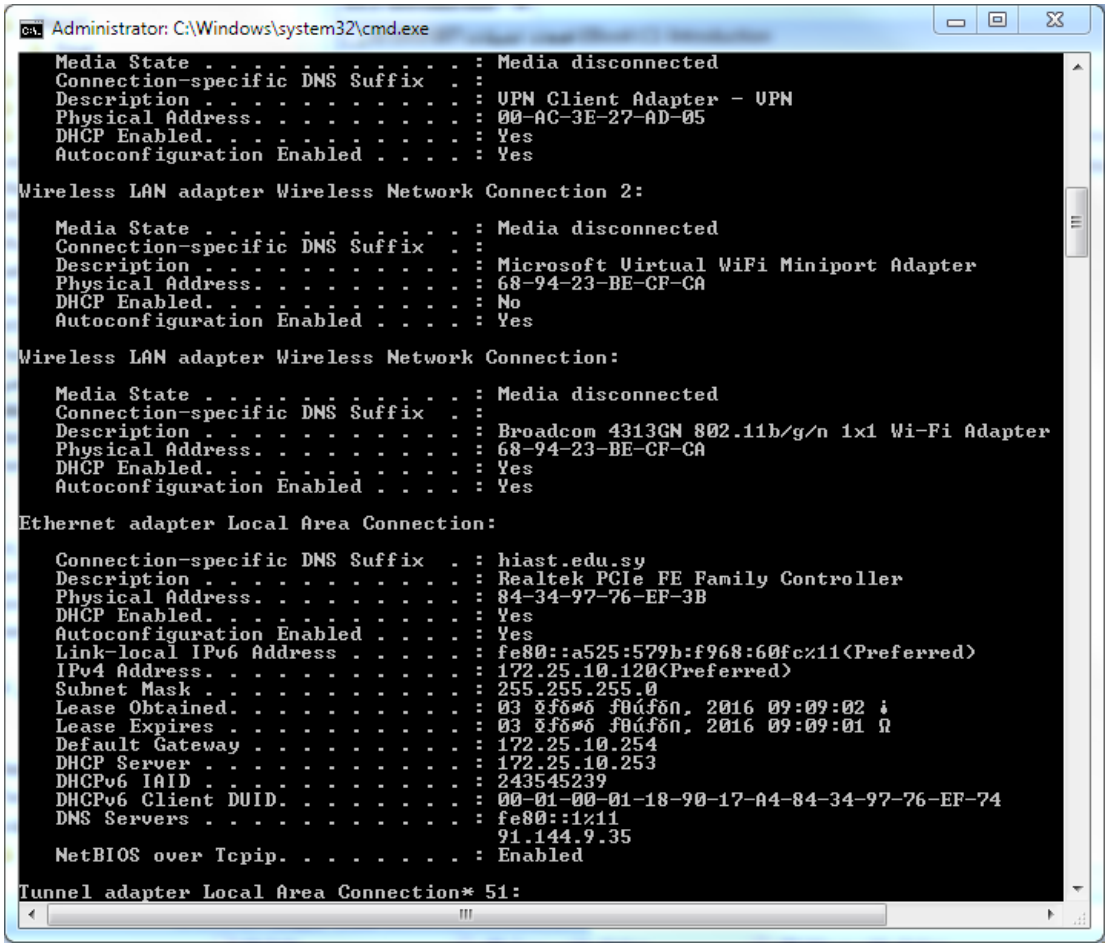
يجب أن يكون الطالب قادراً على الدخول إلى سطر الأوامر عن طريق تنفيذ cmd run ومن ثم طلب الأمر arp –a لمعرفة جدول التقابلات بين العناوين الفيزيائية والعناوين المنطقية المخزنة في الذاكرة الخابية للمضيف. يقوم الطالب بتنفيذ الخطوات التالية:

```
run cmd
arp –a
ipconfig /all |more
```

ستحصل على نتائج قريبة لتلك المعروضة في الشكلين التاليين:



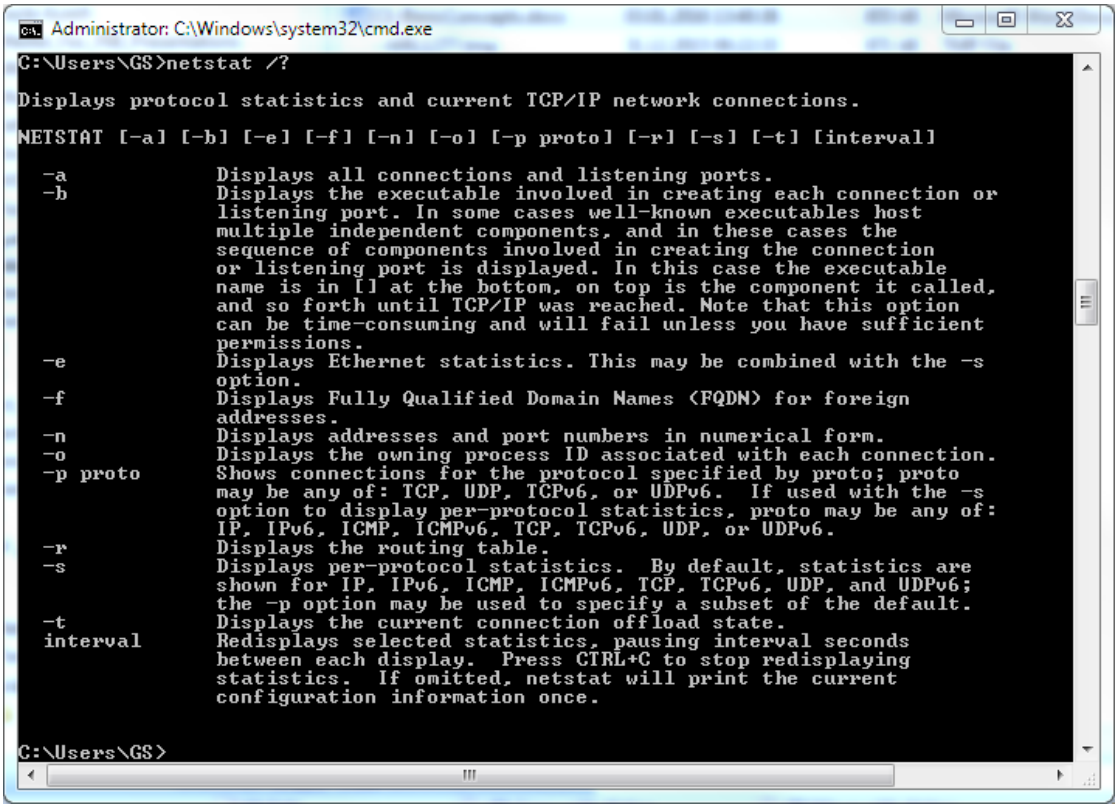

الشكل 8– تنفيذ أمر arp –a




الشكل 9– تنفيذ أمر `ipconfig /all | more`

لاحظ أن الأمر الثاني يظهر إعدادات IP وبطاقات الشبكة المركبة ضمن الحاسب وقناع الشبكة لكل بطاقة وغيرها من الإعدادات المفيدة.

من المفيد أيضاً هنا التأقلم مع الأمر `netstat` الذي يظهر معلومات عن اتصالات TCP واتصالات UDP والبوابات قيد التنصت إضافةً إلى إظهار إحصائيات عن عدد الأطر وعدد الطرود على كل بطاقة شبكة.




الشكل 10– تنفيذ netstat /?

يتوجب على الطالب تجريب الأوامر التالية وفهم النتائج المعروضة:

```
netstat
netstat -e
netstat -n
netstat -s
netstat -r
```

## تمرين 2: التأقلم مع الأمر ping

يستخدم الأمر ping خدمات بروتوكول ICMP echo/reply لاختبار الوصلة بين مضيفين على مستوى IP فقط.

يجرب الطالب الأوامر التالية:

```
Ping /?
Ping -l 2000 some_local_IP_Address
Ping -f -l 2000 some_local_IP_Address
```

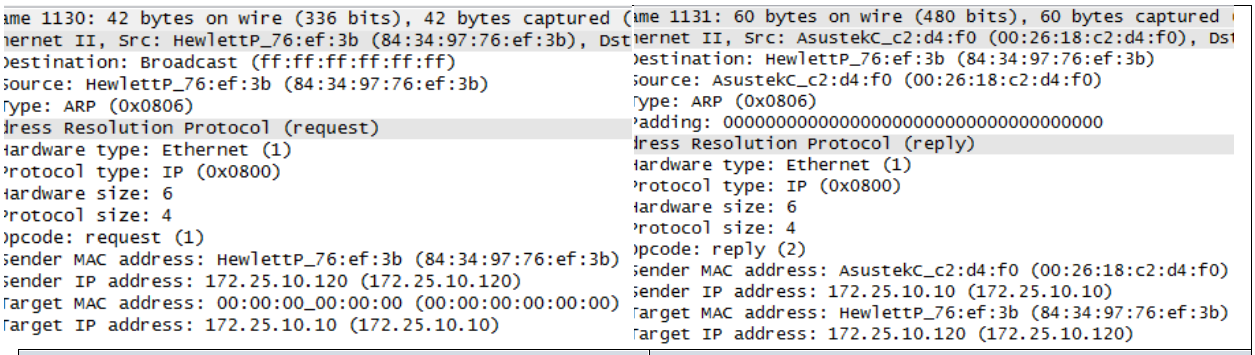
## تمرين 3

سنحاول في هذا التمرين تحليل أطر البروتوكول arp باستخدام wireshark.

| | |
|---|---|
| 1. Type: arp -a | عرض جدول المقابلة بين العناوين الحالي |
| 2. Type: arp -d * | حذف جميع المقابلات الموجودة |
| 3. Type: arp -a | التأكد من نجاح عملية الحذف |
| 4. Start wireshark; choose Interface; then start packet capturing | البدء بالتقاط الطرود عبر البطاقة المستخدمة |
| 5. Ping any local or external IP address | تنفيذ أمر يتطلب حل العناوين |
| 6. Stop wireshark capturing | توقيف التقاط الطرود |
| 7. Type: arp -a | ظهور عنوان MAC للحاسب الوجهة لأمر ping |

يبين الفيديو المرفق تسلسل تنفيذ هذه الأوامر arp_example.avi

بالعودة إلى wireshark سنلاحظ التقاط طردين حسب ما يلي:




| arp request | arp reply |
|---|---|

## تمرين 4: التعرف على عناوين IP وعلى أرقام البوابات

سنقوم الآن بالخطوات التالية:

```
1. Start wireshark capturing
2. Open www.google.com from your browser
3. Stop packet capturing
```

أوجد القيم التالية:

```
4. Source MAC address?
5. Destination MAC address?
6. Source IP address?
7. Destination IP address?
8. Transport layer protocol: UDP or TCP?
9. Source Port address?
10.Destination Port address?
```

# نظام اسم النطاق
# DNS

تستخدم بروتوكولات TCP/IP عناوين IP، التي تعرِّف اتصال مضيف ما إلى الإنترنت تعريفاً شاملاً ووحيداً. لكن الأشخاص يفضلون استخدام الأسماء مكان العناوين (الأرقام). لذلك نحتاج إلى نظام قادر على مقابلة اسم مع عنوان IP أو العكس في بعض الأحيان.

تجدر الإشارة هنا أنه يجب ألا يكون هناك جهازان يحملان الاسم نفسه، لأن ذلك يمكن أن يؤدي إلى التباس في تفسير العنوان خلال عملية التراسل، الأمر الذي من شأنه توجيه المعلومات إلى الجهاز الخاطئ.

بما أن تفويض جهة واحدة لإدارة جداول التقابل بين الأسماء والعناوين غير ممكن عملياً بسبب العدد الكبير للأسماء المختلفة على الإنترنت، حيث وصل عدد أسماء النطاقات المختلفة في نهاية شهر آذار عام 2015 إلى 294 مليون اسم نطاق بينما وصل عدد المضيفين (حواسب) الموجودين ضمن هذه النطاقات إلى مليار في بداية عام 2014. طبعاً عدد مستثمري الإنترنت تجاوز 3.245 مليار عند نهاية العام 2015.

كما أن الحل المتمثل في استخدام حاسوب مركزي وحيد لتخزين جداول المقابلة ووضعه على الإنترنت أيضاً غير ممكن حالياً وذلك نظراً للحجم الكبير للمعلومات (عدد عناوين IP الموجودة على الإنترنت)، إضافةً إلى حركة المرور الكبيرة، من وإلى الحاسوب، التي يولدها مثل هذا الحل. لذلك تعتمد الحلول الحقيقية والتي سندرسها ضمن هذا الفصل على تقسيم المعلومات إلى أجزاء ووضع كل جزء على حاسوب.

## 1. فراغ الأسماء Name space

بما أن عناوين IP وحيدة فيجب أن تكون الأسماء وحيدة أيضاً. يمكننا تنظيم فراغ الأسماء الذي يقابل كل عنوان IP مع أسم وحيد بطريقتين: مسطحة flat وتراتبية Hierarchical.

### فراغ الأسماء المسطح

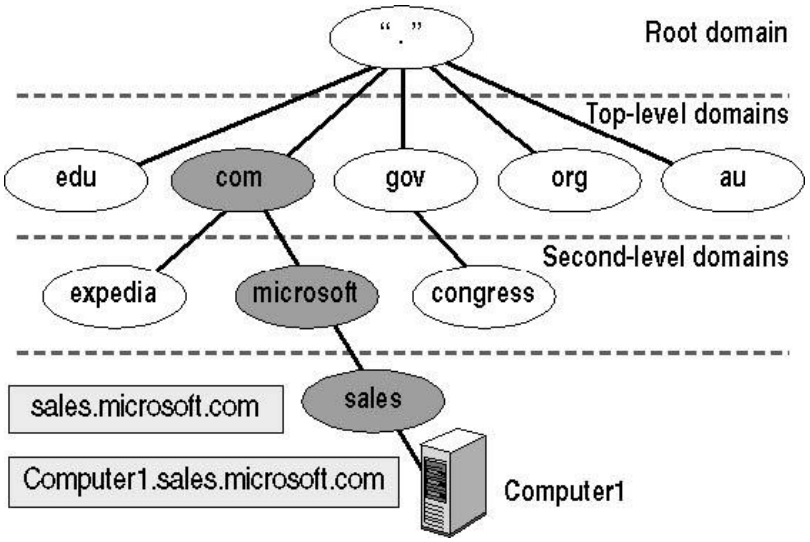
يتكون الاسم ضمن هذا النوع من الفراغات من سلسلة محارف بدون أي بنية. تعود السيئة الأساسية لهذا النوع من الفراغات إلى عدم إمكانية استخدامه ضمن الأنظمة الكبيرة مثل الإنترنت بسبب حاجته إلى التحكم المركزي لتجنب التكرار والالتباس.

### فراغ الأسماء التراتبي

يتكون الاسم هنا من عدة أجزاء. يمكن أن يعرف الجزء الأول طبيعة عمل المؤسسة بينما يعرف الجزء الثاني اسم المؤسسة ويعرف الجزء الثالث أقساماً داخل المؤسسة وهكذا. يمكن هنا جعل سلطة توزيع الأسماء والتحكم بها غير مركزية ويمكننا أيضاً إعطاء المؤسسة إمكانية التحكم بجزء من الأسماء.

### 1.1. فراغ اسم النطاق Domain Name space

جرى تصميم فراغ اسم النطاق بغية تحقيق فراغ أسماء تراتبي. تعرف الأسماء، ضمن هذا التصميم، كبنية شجرة مقلوبة حيث يكون الجذر في الأعلى. يمكن أن تملك الشجرة 128 مستوى على الأكثر. من المستوى "0" المخصص للجذر حتى المستوى "127". (انظر الشكل التالي).




الشكل 1– فراغ اسم النطاق

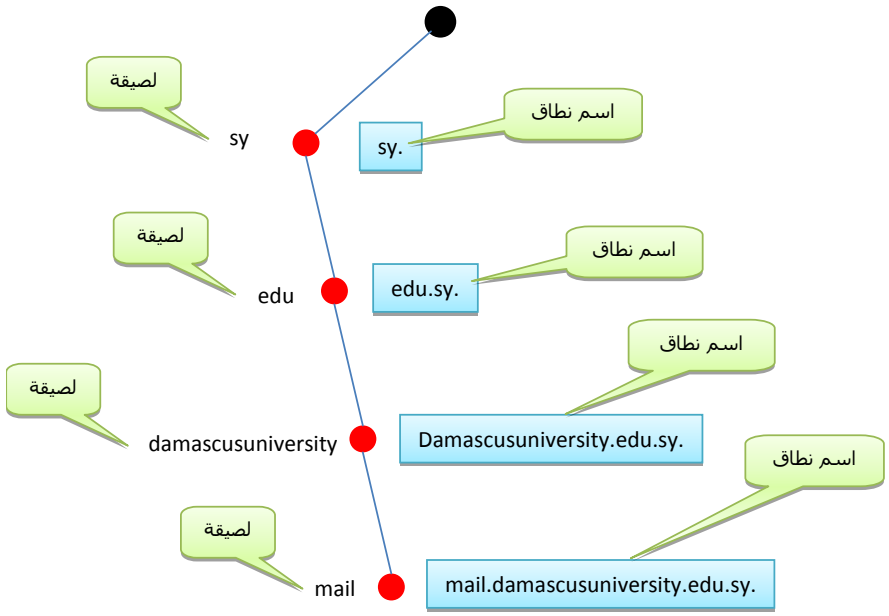
يأتي بعد الجذر في التراتبية مجموعة من النطاقات الأساسية والتي تسمى نطاقات المستوى العلوي Top-Level Domains).

#### اللصيقة Label

تمتلك كل عقدة ضمن الشجرة لصيقة، وهي سلسلة مؤلفة من 63 محرف على الأكثر. تكون لصيقة الجذر هي السلسلة الفارغة null string. يتطلب نظام DNS أن تملك أولاد عقدة ما لصيقات مختلفة لضمان وحدانية أسماء النطاق.

#### اسم النطاق Domain Name

تمتلك كل عقدة من عقد الشجرة اسم نطاق. يتكون اسم النطاق الكامل Full domain name من سلسلة لصيقات مفصولة عن بعضها بنقطة (.). تُقرأ أسماء النطاقات دائماً من العقدة حتى الجذر.
اللصيقة الأخيرة هي لصيقة الجذر الفارغة null. يبين الشكل التالي بعض أسماء النطاقات.




الشكل 2– أسماء النطاقات واللصيقات

### اسم النطاق كامل التأهيل Fully Qualified Domain Name (FQDN)

عندما تنتهي اللصيقة بالسلسة الفارغة، فإنها تدعى FQDN. يحوي FQDN الاسم الكامل للمضيف من الجزء الخاص إلى الجزء العام. فاسم النطاق

`Mailproxy.itech.edu.sy.`

هو FQDN لحاسب اسمه mailproxy موجود لدى شركة itech. يستطيع مخدم DNS مقابلة FQDN إلى عنوان IP. لاحظ أنه يجب على الاسم أن ينته بالسلسلة الفارغة، وبما أنها فارغة فإن الاسم ينته بالنقطة.

### اسم النطاق جزئي التأهيل Partially Qualified Domain Name (PQDN)

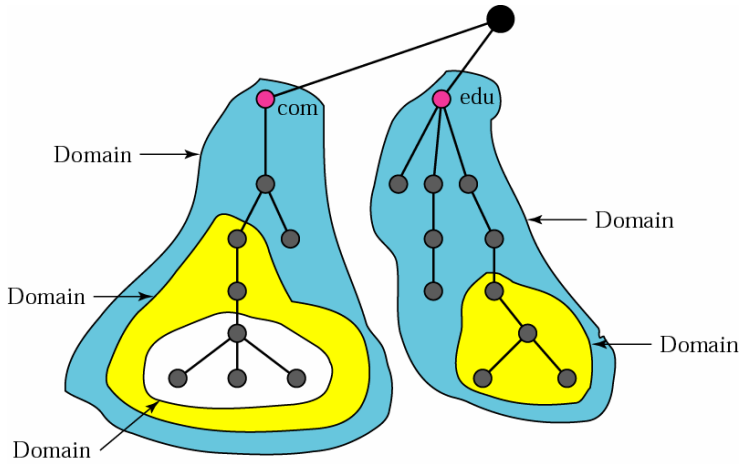
عندما لا ينتهي اسم النطاق بالسلسلة الفارغة فإنه يدعى PQDN. يبدأ PQDN من عقدة ما دون أن يصل إلى الجذر. يستخدم هذا النوع من الأسماء عندما يقع الاسم المطلوب حله ضمن موقع الزبون نفسه. يستطيع هنا الحلال إضافة الجزء الناقص، الذي يدعى اللاحقة suffix، لتوليد FQDN. أي يمكن لمستثمر موجود على موقع `itech.edu.sy` أن يطلب حل الاسم `mailproxy` فقط إلى عنوان IP.

يضيف زبون DNS اللاحقة `itech.edu.sy` إلى الاسم قبل تمرير الطلب إلى مخدم DNS.

يحوي زبون DNS عادةً لائحة من اللاحقات.

### النطاق Domain

النطاق هو شجرة جزئية من فراغ أسماء النطاقات. اسم النطاق هو اسم العقدة الكائنة في أعلى الشجرة الجزئية. يبين الشكل التالي بعض النطاقات. لاحظ أنه يمكن تقسيم النطاق إلى نطاق آخر أو إلى نطاق جزئي.



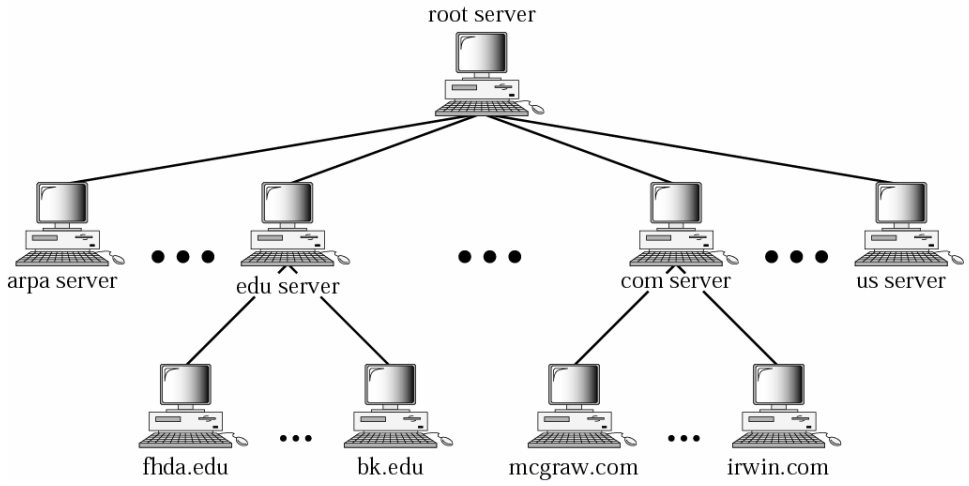

الشكل 3 – النطاقات

## 2.1. توزيع فراغات الأسماء

يجب تخزين المعلومات الموجودة ضمن فراغ أسماء النطاقات في مكان ما أو في عدة أمكنة. طبعاً، لا يمكن تخزين كل هذه المعلومات في حاسوب وحيد لأن ذلك ينعكس على الفعالية والوثوقية. يعتبر هذا الحل غير فعال لأن وجود حاسوب وحيد يجيب على جميع طلبات الإنترنت لحل الأسماء يضيف عبء كبير على النظام ويعتبر غير موثوق لأن تعطل الحاسوب يجعل عملية الحل غير ممكنة.

### تراتبية مخدمات الأسماء

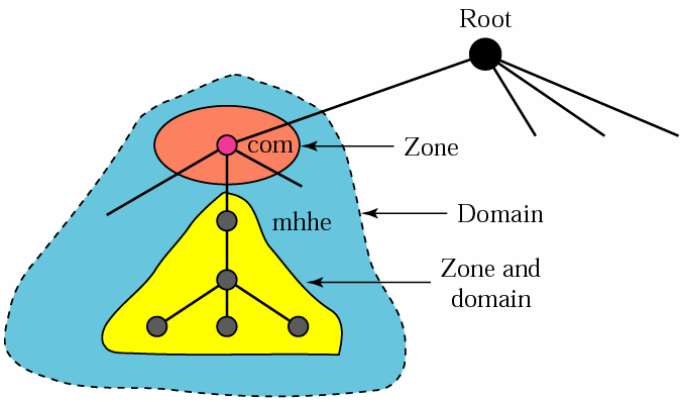
يتمثل الحل لهذه المشكلة في توزيع المعلومات على عدة مخدمات تدعى مخدمات DNS. نقوم، لتحقيق ذلك، بتقسيم الفراغ الكلي إلى عدة نطاقات انطلاقاً من المستوى الأول. أي أننا نترك الجذر كما هو ونقسم المستوى الأول إلى عدة نطاقات (شجرات جزئية). يمكن أن يكون كل مخدم مسؤولاً (ذو سلطة) عن نطاق كبير أو صغير. أي أننا نمتلك الآن تراتبية مخدمات مشابهة لتراتبية أسماء النطاقات كما هو موضح في الشكل التالي.




الشكل 4 – تراتبية مخدمات الأسماء

## المنطقة Zone

بما أنه تم تقسيم فراغ نطاقات الأسماء إلى عدة مخدمات فالجزء الذي يقع تحت مسؤولية (أو تحت سلطة) مخدم ما يدعى بالمنطقة zone. يمكننا تعريف المنطقة على أنها جزء مستمر contiguous part من الشجرة الكلية. إذا كان المخدم صاحب سلطة على نطاق ولم يجر تقسيم النطاق إلى مناطق فيصبح كل من النطاق والمنطقة يؤشران إلى الشيء نفسه. ينشئ المخدم قاعدة معطيات، تدعى ملف المنطقة Zone file، ويحفظ فيه جميع المعلومات المتعلقة بكل عقدة تنتمي لهذا النطاق. لكن إذا قسم المخدم النطاق إلى عدة نطاقات جزئية وفوض جزءاً من سلطته إلى مخدمات أخرى، يصبح مفهومي النطاق والمنطقة مختلفين. يجري هنا تخزين المعلومات المتعلقة بعقد النطاقات الجزئية ضمن مخدمات المستويات الأدنى، بينما يحفظ المخدم الأصلي بعض المراجع عن مخدمات المستويات الأدنى. لا يحرر المخدم الأصلي نفسه من كامل المسؤولية: فهو ما يزال مسؤولاً عن المنطقة، لكن المعلومات التفصيلية عنها موجودة ضمن مخدمات ذات مستوٍ أدنى كما هو موضح في الشكل التالي.




الشكل 5 – المناطق والنطاقات

## المخدم الجذر Root Server

المخدم الجذر هو المخدم الذي تشمل منطقته على الشجرة الكلية. لا يخزن عادةً المخدم الجذر معلومات عن النطاقات وإنما يفوض سلطته إلى مخدمات أخرى، على أن يحافظ على مراجع عن هذه المخدمات.

## المخدمات الأساسية والثانوية Primary and Secondary servers

يعرف نظام DNS نوعين من المخدمات: أساسي وثانوي. المخدم الأساسي هو المخدم الذي يخزن ملف يحوي معلومات عن المنطقة التي يملك سلطة عليها. يكون المخدم مسؤولاً عن إنشاء ومتابعة وتحديث ملف المنطقة الموجود على قرص محلي.

المخدم الثانوي هو المخدم الذي ينقل المعلومات الكاملة المتعلقة بمنطقة من مخدم آخر (أساسي أو ثانوي) ويخزن الملف على قرص محلي. لا ينشئ ولا يحدث المخدم الثانوي ملفات المنطقة. يقوم دائماً المخدم الأساسي بعملية التحديث ويرسل نسخة عن ملف المنطقة المحدَّث إلى المخدم الثانوي.

يمتلك المخدمان الأساسي والثانوي السلطة على المنطقة المسؤولين عنها.

تتمثل الفكرة الرئيسة من استخدام نوعين من المخدمات في تحقيق نوعاً من التكرارية في المعلومات بحيث إذا تعطل أحد المخدمات فإن المخدم الآخر يستمر في تخديم الزبائن.

لاحظ أنه يمكن أن يكون مخدم أساسياً لمنطقة ما وثانوياً لمنطقة أخرى.

## 2. مخدمات DNS على الإنترنت

تاريخياً كانت أسماء نطاقات المستوى العلوي مقسمة إلى 3 أصناف لكن منذ عام 2015، بدأت سلطة Internet Corporation for Assigned Names and Numbers (ICANN) (وهي السلطة المخولة توزيع عناوين الإنترنت والأسماء) بإضافة مجموعات أخرى أو تعديل التسميات الموجودة.

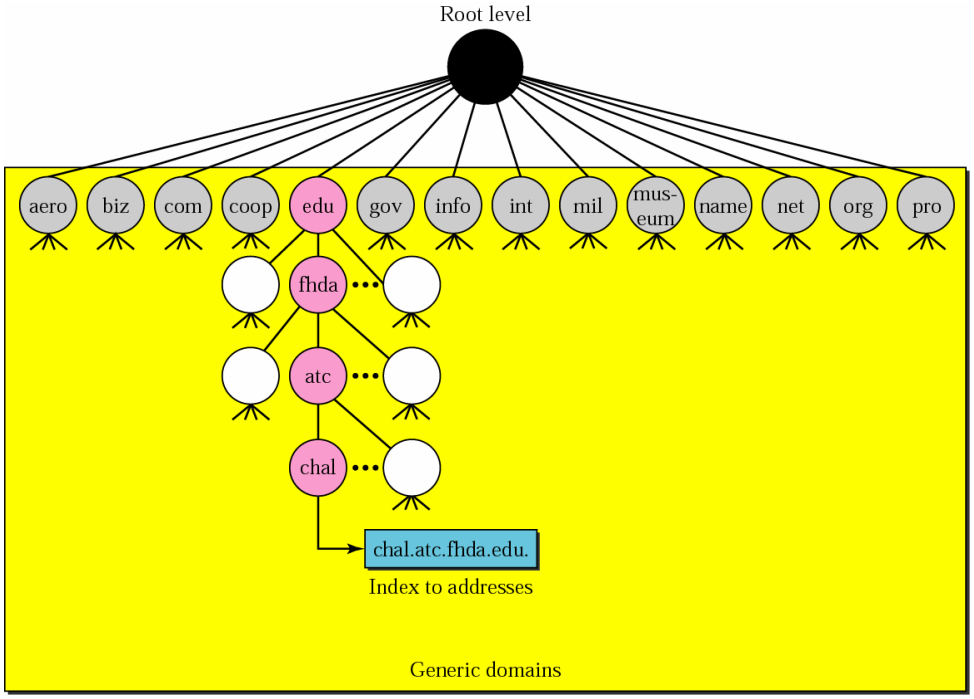
التسميات الحالية لهذه المجموعات هي نطاقات أسماء:

- البنية التحتية Infrastructure TLD والمعروف تحت اسم ARPA
- العامة Generic TLD (gTLD)
- العامة المحصورة Restricted Generic TLD (grTLD): تتم إدارته من قبل مسجلين معترف بهم من قبل IANA
- راعو المستوى الأول Sponsored TLD (sTLD): تتم إدارتها من قبل شركات خاصة
- البلدان Country code TLD (ccTLD)
- البلدان العالمية Internationalized country code TLD (IDN ccTLD): تستخدم نطاقات الأسماء غير اللاتينية مثل اسم النطاق العلوي الذي ينتهي باللاحقة ".سورية" أو اللاحقة ".شبكة" أو اللاحقة ".ভারত" للهند.
- الاختبارية test TLD (tTLD): أسماء نطاقات لأغراض الاختبارات فقط.

### النطاقات العامة Generic domains

تاريخياً وحتى نهاية الثمانينيات كان معرف 7 مجموعات من المستوى العلوي العام وهي (.com, .edu, .gov, .int, .mil, .net, and .org) خلال عامي 2001 و2002 ، تمت إضافة 7 مجموعات إضافية وهي (.biz, .info, .name, .pro, .aero, .coop, and .museum)

تعرف النطاقات العامة المضيفين المسجلين حسب سلوكهم العام. تعرف كل عقدة من عقد الشجرة نطاقاً الذي هو مؤشر لقاعدة معطيات فراغ أسماء النطاقات (انظر الشكل التالي).




الشكل 6 – النطاقات العامة

بالنظر إلى الشجرة، نلاحظ أن المستوى العلوي من قسم النطاقات العامة يعرف 14 لصيقة ممكنة. تصف هذه اللصيقات نوع المؤسسة كما هو مبين في الجدول التالي:

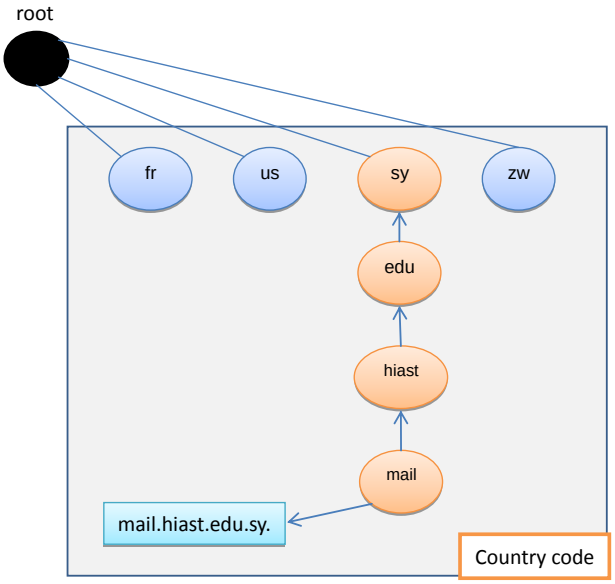
| Label | الوصف |
|---|---|
| aero | شركات الخطوط الجوية والفضاء الخارجي |
| biz | الشركات والأعمال Businesses |
| com | المؤسسات التجارية |
| coop | منظمات الأعمال التعاونية |
| edu | المؤسسات التعليمية |
| gov | المؤسسات الحكومية |
| info | مزودو خدمات المعلومات |
| int | المنظمات العالمية |
| mil | المجموعات العسكرية |
| museum | المتاحف وغيرها من المنظمات غير الربحية |
| name | الأسماء الشخصية (الأفراد) |
| net | مراكز دعم الشبكات |
| org | المنظمات غير الربحية |
| pro | المنظمات الاحترافية |

الشكل 7 – النطاقات العامة

### نطاقات أسماء البلدان

يستخدم قسم نطاقات أسماء البلدان محرفين للدلالة على اختصار لاسم البلد (مثلاً sy لسورية). يمكن أن تكون اللصيقة الثانية (المستوى الثاني من اللصيقات) معرفة للمنظمات أو المؤسسات أو تكون أكثر تحديداً كما هو عليه الحال في الولايات المتحدة حيث يدل المستوى الثاني على اختصار المقاطعة مثل ca.us.

يوضح الشكل التالي قسم أسماء البلدان.




الشكل 8 – نطاقات أسماء البلدان

### النطاقات المعكوسة Inverse Domain

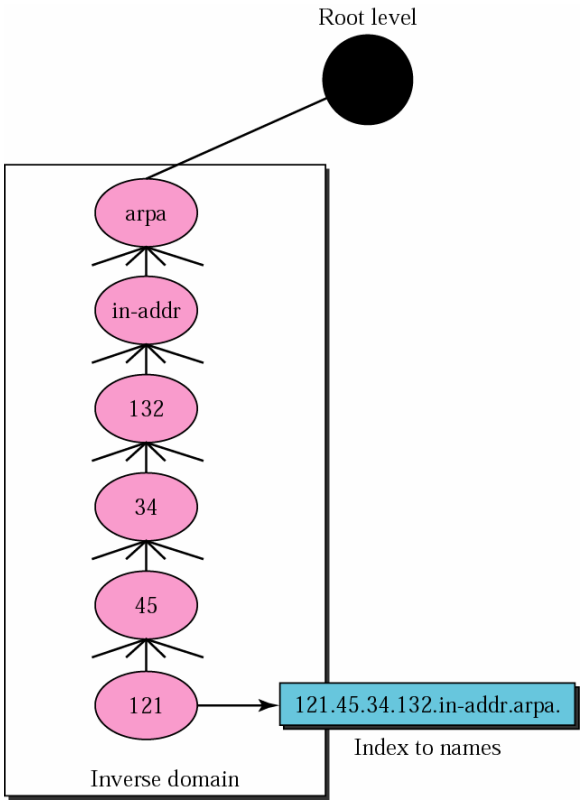
تستخدم النطاقات المعكوسة لحل عنوان إلى اسم. يمكن حدوث ذلك عندما يتلقى مخدم ما لطلب من زبون بغية تحقيق مهمة ما. برغم أن المخدم يملك لائحة بأسماء المستخدمين المخولين للنفاذ إليه، فهو لا يستطيع معرفة سوى عنوان IP للزبون (من طرد IP المستقبل). لذلك يطلب المخدم من الحلال resolver إرسال طلب استفسار إلى مخدم DNS لحل عنوان IP هذا إلى اسم للتحقق من وجود اسم الزبون ضمن لائحة المستخدمين المخولين.

يدعى هذا النوع من الاستفسارات بالاستفسار المعكوس أو الاستفسار عن المؤشر Pointer (PTR) query. لمعالجة الاستفسار عن المؤشر، يجري إضافة النطاق العكسي إلى فراغ أسماء النطاقات حيث يجري تسمية عقدة المستوى العلوي arpa. يطلق على عقدة المستوى الثاني الوحيدة أيضاً الاسم (inverse address) in-addr. تعرف بقية النطاق عناوين IP.

تكون المخدمات التي تعالج النطاقات المعكوسة أيضاً تراتبية. هذا يعني أن الجزء المخصص لمعرف الشبكة يجب أن يكون على مستوٍ أعلى من الجزء المخصص لمعرف الشبكة الجزئية وكذلك الأمر بالنسبة للجزء

المخصص لمعرف الشبكة الجزئية الذي يجب أن يكون على مستوٍ أعلى من جزء معرف المضيف. هذا من شأنه جعل النطاق يبدو معكوساً مقارنةً بالنطاقات العامة أو نطاقات أسماء البلدان.

لمتابعة اصطلاح قراءة لصيقات النطاق من الأسفل إلى الأعلى، فإنه تجري قراءة عنوان IP التالي 132.34.45.121 (وهو عنوان من الصف B ذو معرف شبكة 132.34) على الشكل 121.45.34.132.in-addr.arpa. انظر الشكل التالي لتفسير النطاق المعكوس.




الشكل 9 – النطاق المعكوس

## المسجل Registrar

يقوم المسجل بإضافة النطاقات الجديدة إلى DNS. المسجل هو كيان تجاري معتمد من قبل ICANN. يبدأ المسجل بالتحقق من كون اسم النطاق المطلوب وحيد (غير مخصص مسبقاً) ومن ثم يضيفه إلى قاعدة معطيات DNS.

## 1.2. الحل Resolution

تدعى عملية مطابقة عنوان باسم أو اسم بعنوان بالحل name-address resolution.

### الحلال Resolver

يعمل نظام DNS على أساس تطبيق مخدم–زبون. عندما يريد حاسوب ما مطابقة عنوان باسم أو العكس فإنه يستدعي زبون DNS الذي يدعى بالحلال. يتصل الحلال بأقرب مخدم DNS ويطلب من المطابقة. إذا كان المخدم يملك الجواب فإنه يجاوب الحلال؛ وإلا فإنه إما يوجه الحلال إلى مخدمات DNS أخرى أو يسأل مخدمات أخرى لتزويد المعلومة المطلوبة.

بعد استلام الحلال للمطابقة المطلوبة، فإنه يقوم بتفسير الجواب للتأكد من كونه حل صحيح أو خاطئ، ومن ثم يسلم النتائج إلى الإجرائية الطالبة.

### مطابقة الأسماء بعناوين

يقوم الحلال في معظم الأحيان بإرسال اسم نطاق لمخدم DNS وبطلب عنوان IP المقابل. في هذه الحالة، يبحث المخدم ضمن النطاقات العامة أو نطاقات أسماء البلدان لإيجاد المطابقة.

### مطابقة العناوين بأسماء

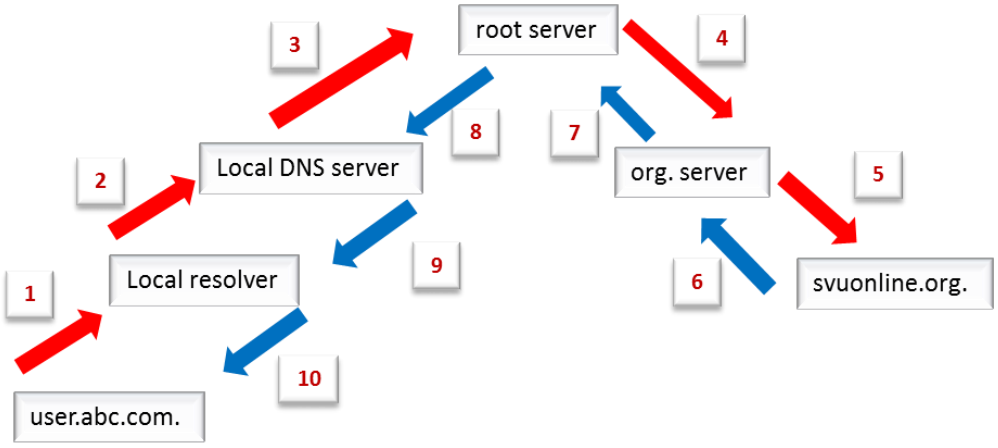
يمكن أن يرسل الزبون عنوان IP إلى المخدم ويطلب مطابقته مع اسم نطاق. يستخدم هنا المخدم النطاق المعكوس. على جميع الأحوال، فإن المخدم يستقبل عنوان IP بالصيغة المعكوسة بعد إضافة لصيقتي in-addr و arpa. أي بفرض أن الحلال استقبل العنوان 132.34.45.121 فإنه يقوم أولاً بعكس العنوان ومن ثم يضيف اللصيقتين ويرسل القيمة

"121.45.34.132.in-addr.arpa."

التي يجري استقبالها ومطابقتها من قبل مخدم DNS المحلي.

### الحل العودي Recursive Resolution

يمكن أن يطلب الزبون (الحلال) إجابة عودية من المخدم. هذا يعني أن الحلال يتوقع من المخدم تزويد الإجابة النهائية. فإذا كان المخدم هو صاحب السلطة على نطاق الأسماء المطلوب فإنه يبحث ضمن قاعدة المعطيات ويرسل جواب المطابقة مباشرةً. أما إذا لم يكن المخدم هو صاحب السلطة على نطاق الأسماء المطلوب فإنه يرسل الطلب إلى المخدم الجذر root وينتظر الجواب. يرسل المخدم الجذر الطلب إلى مخدم المستوى الأعلى TLD. عندما تجري مطابقة الطلب في مكان ما فإنه يجري إرسال جواب المطابقة عودياً حتى تصل إلى الزبون الطالب كما هو موضح في الشكل التالي.



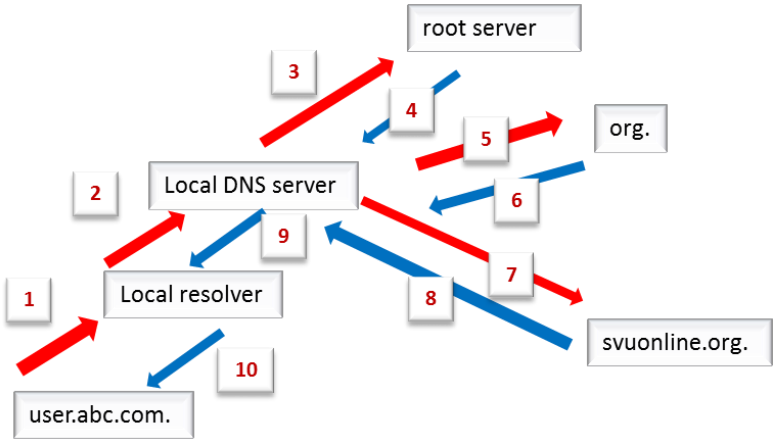

الشكل 10– الحل العودي

فرضنا في الشكل السابق أن المستثمر user.abc.com يريد التواصل مع أحد مخدمات الجامعة الافتراضية وليكن mail.svuonline.org. لذلك يقوم بإرسال الطلب إلى الحلال الموجود عادةً على الجهاز نفسه والذي بدوره يرسل الطلب إلى مخدم نطاقات الأسماء المحلي الذي يحوله إلى المخدم الجذر. يرسل المخدم الجذر الطلب إلى مخدم المستوى العلوي org الذي يرسله إلى مخدم أسماء الجامعة الافتراضية وليكن ns.svuonline.org (فعلياً مخدم أسماء النطاقات في الجامعة الافتراضية هو ns2.scs-net.org). يرسل مخدم أسماء الجامعة الافتراضية الجواب إلى المخدم الأعلى ومنه إلى الجذر أي بالطريق العكسي حتى يصل الجواب إلى المستثمر. تفيد هذه الطريقة في تخزين التقابل بين الاسم والعنوان في جميع الذاكرات الخابية على هذه المخدمات حيث يمكن الإجابة في المستقبل على أي طلب محلياً دون الرجوع إلى المخدم النهائي الذي يعرف باسم المخدم صاحب السلطة Authoritative Server.

Animation: يجري عرض تسلسل خطوات طلب الحل وفق الترتيب المبين في الشكل السابق

## الحل التكراري Iterative Resolution

في حال لم يطلب الزبون حلاً عودياً فإن الحل سيكون تكرارياً. عندما يصل الطلب إلى المخدم الجذر فإنه يرسل عنوان IP للمخدم org. الذي يرسل الجواب إذا كان متوافراً في الذاكرة الخابية لديه وإلا فإنه يرسل عنوان IP للمخدم ns.svuonline.org. الذي باعتباره ذو السلطة يرسل الجواب إلى مخدم الأسماء المحلي وهكذا حتى يصل الجواب إلى المستثمر.




الشكل 11– الحل التكراري

Animation: يجري عرض تسلسل خطوات طلب الحل وفق الترتيب المبين في الشكل السابق

## التخبئة Caching

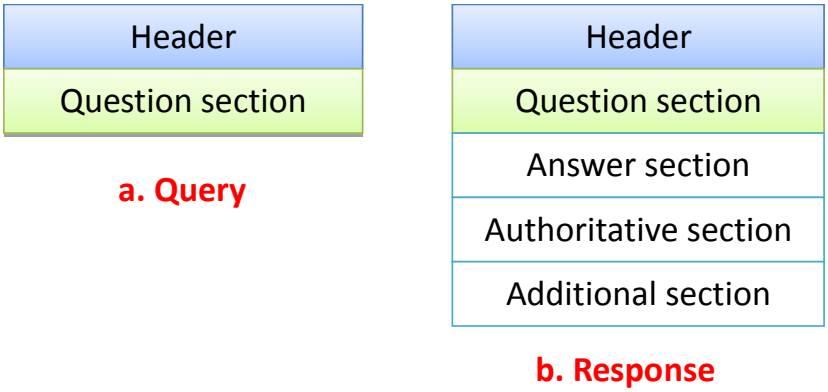
عندما يطلب مخدم ما مطابقة من مخدم آخر ويستقبل الجواب فإنه يخزن هذه المعلومات ضمن ذاكرة خابية قبل أن يرسل الجواب إلى الزبون. فإذا طلب زبون آخر نفس المطابقة، فيقوم المخدم بالبحث ضمن الذاكرة الخابية ويحل المشكلة. لكن حتى يعلم الزبون أن المطابقة قادمة من الذاكرة الخابية وليس من مخدم صاحب سلطة فإن المخدم يؤشر الجواب بالعلامة unauthoritative.

تسرع التخبئة عملية المطابقة لكن يمكن أن تسبب بعض المشاكل. إذا خبأ المخدم معلومات المطابقة لفترة طويلة فإنه يمكن أن يرسل للزبون معلومات مطابقة قديمة. لحل هذه المشكلة، يمكن استخدام التقنيتين التاليتين: أولاً، يضيف المخدم صاحب السلطة معلومات متعلقة بمدة الحياة Time-to-live (TTL) لسجل المطابقة. يعرِّف الزمن (بالثانية) الممكن من خلاله تخبئة المعلومات. بعد انقضاء هذا الزمن، تصبح المطابقة غير صالحة ويجب إعادة إرسال أي استفسار إلى المخدم صاحب السلطة. ثانياً، يطلب DNS أن يحتفظ كل مخدم بعداد TTL لكل مطابقة يقوم بوضعها ضمن الذاكرة الخابية. يجب على المخدم البحث ضمن الذاكرة الخابية دورياً وحذف المطابقات التي انقضت صلاحيتها.

تجدر الإشارة هنا إلى أن الاتصال من المخدم المحلي مع المخدم الجذر لا يجري بشكل دائم وذلك لكون عناوين IP لمخدمات النطاق العلوي تكون عادةً موجودة ضمن الذاكرة الخابية للمخدم المحلي الذي يستطيع تجاوز المخدم الجذر والاتصال مباشرةً مع مخدم النطاق العلوي الأمر الذي يخفف من عبء المخدم الجذر.

## 3. رسائل DNS

يستخدم DNS نوعين من الرسائل: استفسار وجواب query and response لهما الصيغة نفسها. تحوي رسالة الاستفسار على ترويسة وعلى سجلات سؤال؛ بينما تحوي رسالة الجواب على ترويسة وعلى سجلات سؤال وسجلات جواب وسجلات سلطة وسجلات إضافية كما هو مبين في الشكل التالي:




الشكل 12– رسائل الاستفسار والجواب

### الترويسة Header

تملك رسائل الاستفسار والجواب صيغة الترويسة نفسها حيث تكون بعض الحقول معدومة القيمة من أجل رسائل الاستفسار. تتألف الترويسة من 12 بايت موضحة في الشكل التالي.

| Identification | Flags |
|---|---|
| Number of question records | Number of answer records (All 0s in query message) |
| Number of Authoritative records (All 0s in query message) | Number of additional records (All 0s in query message) |

الشكل 13– صيغة الترويسة

- التعريف Identification. حقل مؤلف من 16 بت يستخدمه الزبون لمطابقة الجواب مع السؤال. يستخدم الزبون رقم تعريف مختلف لكل استفسار جديد. يكرر المخدم قيمة التعريف عند إرساله للجواب.
- الراية Flag. حقل مؤلف من 16 بت تشمل الحقول الجزئية التالية.


الشكل 14– حقل الراية

أما بالنسبة لمعنى الحقول الجزئية فهو:

- QR (Query/Response). بت واحد يعرف نوع الرسالة: استفسار (0) أو جواب (1).
- OpCode. حقل جزئي مكون من 4 بتات يعرف نوع الاستفسار أو نوع الجواب ("0" معياري، "1" معكوس، "2" طلب حالة المخدم).
- AA (Authoritative Answer). حقل مكون من بت واحد. يأخذ القيمة "1" عندما يكون المخدم صاحب السلطة هو الذي حل المطابقة وإلا فإنه يأخذ القيمة "0".
- TC (Truncated). حقل مكون من بت واحد. عندما يأخذ القيمة "1" فإنه يعني أن الإجابة تتألف من أكثر من 512 بايت وتم تقليصها إلى 512 بايت. تستخدم عندما يستخدم DNS خدمات UDP.
- RD (Recursion Desired). حقل مكون من بت واحد. عندما يأخذ القيمة "1" يعني أن الزبون يرغب بإجابة عودية. تكرر هذه القيمة في رسائل الجواب.
- RA (Recursion Available). حقل مكون من بت واحد. عندما يأخذ القيمة "1" ضمن الجواب فهذا يعني أن الإجابة العودية متاحة.
- Reserved. حقل مكون من 3 بتات تأخذ القيمة "000".
- rCode. حقل مكون من 4 بتات يستخدم لإظهار حالة الخطأ في الجواب. طبعاً، يستطيع فقط المخدم صاحب السلطة أن يصدر هذا الحكم. يبين الجدول التالي القيم المحتملة لهذا الحقل.

| Value | Meaning |
|---|---|
| 0 | No error |
| 1 | Format error |
| 2 | Problem at name server |
| 3 | Domain reference problem |
| 4 | Query type not supported |
| 5 | Administratively prohibited |
| 6-15 | Reserved |

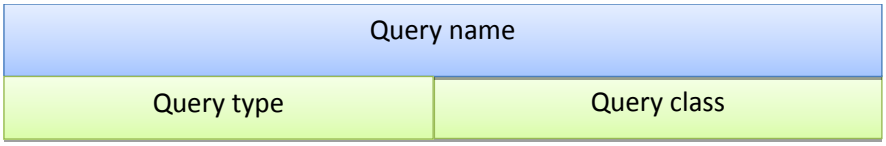
الشكل 15– قيم rCode

- عدد سجلات الأسئلة. حقل مكون من 16 بت يحوي عدد الاستفسارات ضمن قسم السؤال من الرسالة.
- عدد سجلات الإجابة. حقل مكون من 16 بت يحوي عدد سجلات الإجابة ضمن قسم الجواب من رسالة الجواب. يأخذ القيمة "0" في رسالة الاستفسار.
- عدد سجلات السلطة. حقل مكون من 16 بت يحوي عدد سجلات السلطة ضمن قسم السلطة من رسالة الجواب. يأخذ القيمة "0" في رسالة الاستفسار.
- عدد السجلات الإضافية. حقل مكون من 16 بت يحوي عدد السجلات الإضافية ضمن القسم الإضافي من رسالة الجواب. يأخذ القيمة "0" في رسالة الاستفسار.

## 4. أنواع السجلات

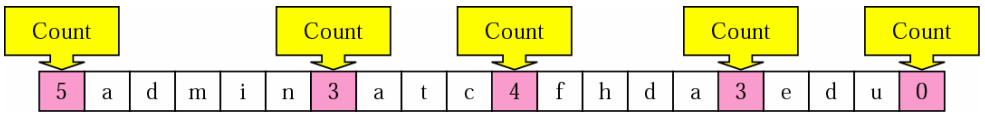
### سجل السؤال

يستخدم الزبون سجل السؤال للحصول على معلومات من المخدم. يبين الشكل التالي صيغة سجل السؤال.




الشكل 16– صيغة سجل السؤال

- اسم الاستفسار Query name. حقل متغير الطول يحوي اسم النطاق كما هو مبين في الشكل التالي:




الشكل 17– صيغة اسم الاستفسار

- نوع الاستفسار Query type. حقل مكون من 16 بت يعرف نوع الاستفسار. يبين الشكل التالي بعض الأنواع الشائعة.

| النوع | اختصار | المعنى | الوصف |
|---|---|---|---|
| 1 | A | Address | عنوان IPv4. يستخدم لحل اسم نطاق إلى عنوان IPv4 |
| 2 | NS | Name Server | يعرف المخدم صاحب السلطة للمنطقة |
| 5 | CNAME | Canonical Name | يعرف اسم مستعار alias للاسم الرسمي للمضيف |

| | | | |
|---|---|---|---|
| 6 | SOA | Start Of Authority | يعلم بداية المنطقة. يكون عادةً أول سجل ضمن ملف المنطقة |
| 11 | WKS | Well-known services | يعرف خدمات الشبكة التي يزودها المضيف |
| 12 | PTR | Pointer | يستخدم لحل عنوان IP إلى اسم نطاق |
| 13 | HINFO | Host Information | يوصف البنية الصلبة ونظام تشغيل المضيف |
| 15 | MX | Mail Exchange | لإعادة توجيه البريد الإلكتروني إلى مخدم بريد |
| 28 | AAAA | Address | عنوان IPv6 |
| 252 | AXFR | | طلب نقل ملف المنطقة الكلي |
| 255 | ANY | | طلب جميع السجلات |

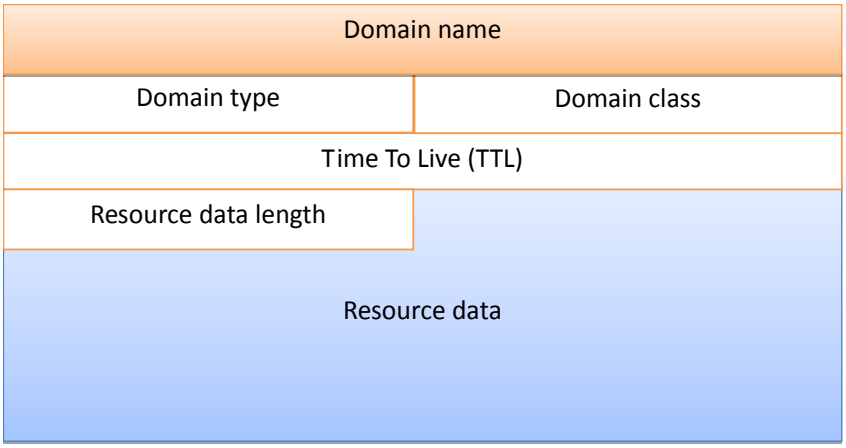
الشكل 18– أنواع الاستفسار

- صنف الاستفسار Query class. حقل مكون من 16 بت يعرف البروتوكول المحدَّد الذي يستخدم DNS.

ملاحظة: يجري عادةً استخدام Query Type=12 و (standard) OpCode=0 عند التحويل من عنوان إلى اسم أي OpCode=1 غير مستخدمة فعلياً هنا.

### سجل الموارد Resource record

يُقرن كل اسم نطاق بسجل يدعى سجل الموارد. تشمل قاعدة معطيات المخدم على سجلات الموارد. سجلات الموارد هي في الحقيقة ما يرسله المخدم إلى الزبون. يبين الشكل التالي صيغة سجل الموارد.




الشكل 19– صيغة سجل الموارد

- اسم النطاق Domain name. حقل متغير الطول يحوي اسم النطاق. هو تكرار لاسم النطاق المطلوب ضمن سجل السؤال.
- نوع النطاق Domain type. هذا الحقل هو نفس الحقل المستخدم ضمن نوع الاستفسار في سجل السؤال باستثناء أن آخر نوعين غير ممكنين.
- صنف النطاق Domain class. هذا الحقل هو نفس صنف النطاق المستخدم في سجل السؤال.
- مدة الحياة Time To Live. حقل مكون من 32 بت يعرف مدة صلاحية الجواب بالثانية.
- طول معطيات المورد Resource data length. حقل متغير الطول يحوي الجواب على الاستفسار (ضمن قسم الجواب) أو اسم النطاق للمخدم صاحب السلطة (ضمن قسم السلطة) أو معلومات إضافية (ضمن قسم المعلومات الإضافية). تتعلق صيغة ومحتوى هذا الحقل بقيمة حقل النوع. يمكن أن يأخذ أحد القيم التالية:
  - رقم. يكتب بالبايت كعنوان IP المؤلف من أربعة بايتات.
  - اسم النطاق. عبارة عن سلسلة لصيقات. تُسبق كل لصيقة ببايت واحد يعرف عدد المحارف ضمن اللصيقة. يجب أن تكون قيمة البتين الأكثر دلالة ضمن الحقل الذي يعرف عدد المحارف "00" وذلك للتمييز بين حقل الطول وحقل مؤشر الإزاحة. لا يسبب ذلك أي مشكلة طالما أن طول أي حقل لا يتجاوز 63 محرفاً.
  - مؤشر الإزاحة Offset pointer. يمكن الاستعاضة عن اسم النطاق بمؤشر الإزاحة. يتكون مؤشر الإزاحة من بايتين بحيث تكون قيمة البت الأكثر دلالة لكل بايت هي "1".
  - سلسلة محارف Character string. حقل مكون من بايت واحد يتبعه عدد المحارف المعرفة ضمن حقل الطول. يمكن أن يكون طول سلسلة المحارف 255 محرفاً (بما فيها حقل الطول).

## 5. أمثلة

مثال 1

يرسل الحلال رسالة استفسار إلى مخدم محلي لإيجاد عنوان IP للمضيف "chal.fhda.edu.".
سنناقش رسالة الاستفسار ورسالة الجواب.
يبين الشكل التالي رسالة الاستفسار التي يرسلها الحلال.

| | | | |
|---|---|---|---|
| 0x1333 | | 0x0100 | |
| 1 | | 0 | |
| 0 | | 0 | |
| 4 | 'c' | 'h' | 'a' |
| 'l' | 4 | 'f' | 'h' |
| 'd' | 'a' | 3 | 'e' |
| 'd' | 'u' | 0 | Continued on next line |
| 1 | 1 | | |

الشكل 20– المثال 1 : رسالة الاستفسار

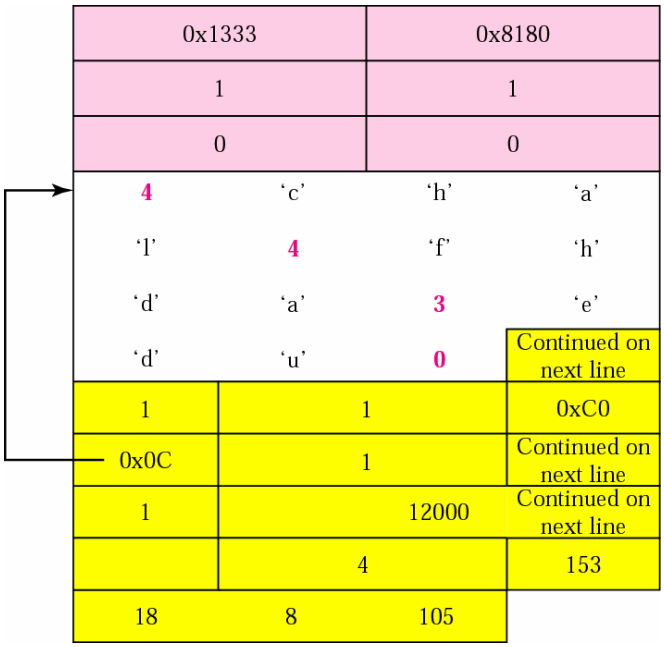
يظهر أول بايتان المعرف 1333 الذي يستخدم كرقم تسلسلي للربط بين استفسار وجواب. بما أنه يمكن للحلال أن يرسل عدة استفسارات إلى مخدم النطاق، فيفيد هذا الحقل في فرز الأجوبة التي تأتي خارج الترتيب. يحوي البايت الثاني الرايات Flags ذات القيمة الست عشرية 0x0100 أو الثنائية 0000000100000000. يمكننا إظهارها أيضاً بالصيغة التالية:

| QR | OpCode | AA | TC | RD | RA | Reserved | rCode |
|---|---|---|---|---|---|---|---|
| 0 | 0000 | 0 | 0 | 1 | 0 | 000 | 0000 |

يعرف بت QR الرسالة على أنها استفسار. يعرف OpCode الاستفسار على أنه معياري. يعلن بت Recursion Desired (RD) على الرغبة بالطلب العودي. تحوي الرسالة سجل سؤال واحد. اسم النطاق هو

4chal4fhda3edu0

يعرف البايتان التاليان نوع الاستفسار على أنه عنوان IP؛ يعرف آخر بايتات الصف على أنه Internet.
يظهر الشكل التالي إجابة المخدم.




الشكل 21– المثال 1: رسالة الجواب

إن الجواب مشابه للاستفسار باستثناء كون الرايات مختلفة وعدد سجلات الجواب هو "1". قيمة الرايات الست عشرية هي 0x8180 أو الثنائية 1000000110000000 التي توزع على الحقول الجزئية كما يلي:

| QR | OpCode | AA | TC | RD | RA | Reserved | rCode |
|---|---|---|---|---|---|---|---|
| 1 | 0000 | 0 | 0 | 1 | 1 | 000 | 0000 |

يعرف بت QR الرسالة على أنها جواب. يعرف حقل OpCode الرسالة على أنها معيارية. يعرف حقل RA أن العودية متاحة. تحوي الرسالة سجل سؤال واحد وسجل جواب واحد. يجري تكرار سجل السؤال من رسالة الاستفسار. قيمة سجل الجواب هي 0xC00C (موزعة على سطرين) والتي تؤشر إلى سجل السؤال بدلاً عن تكرار قيمة اسم النطاق. يعرف الحقل التالي نوع النطاق (عنوان) ويعرف الحقل الذي يليه الصنف (إنترنت). الحقل ذو القيمة 12000 هو حقل TTL (12000 ثانية). الحقل التالي هو طول معطيات المورد التي هي عنوان IP (153.18.8.105).

### مثال 2

استقبل مخدم FTP طرداً من زبون FTP ذو العنوان 153.2.7.9. يريد مخدم FTP التأكد أن زبون FTP هو زبون مخول. يستطيع مخدم FTP مراجعة ملف يحوي أسماء الزبائن المخولين. لكن الملف يحوي فقط أسماء النطاقات. يعرف مخدم FTP عنوان IP للزبون الطالب (وهو عنوان المصدر الموجود ضمن طرد IP). يطلب مخدم FTP من الحلال إرسال استفسار معكوس إلى مخدم DNS لمعرفة اسم زبون FTP.

يظهر الشكل التالي رسالة الاستفسار التي أرسلها الحلال إلى مخدم DNS.

| 0x1200 | | 0x0900 | |
|---|---|---|---|
| 1 | | 0 | |
| 0 | | 0 | |
| 1 | ‘9’ | 1 | ‘7’ |
| 1 | ‘2’ | 3 | ‘l’ |
| ‘5’ | ‘3’ | 7 | ‘i’ |
| ‘n’ | ‘-’ | ‘a’ | ‘d’ |
| ‘d’ | ‘r’ | 4 | ‘a’ |
| ‘r’ | ‘p’ | ‘a’ | 0 |
| 12 | | 1 | |

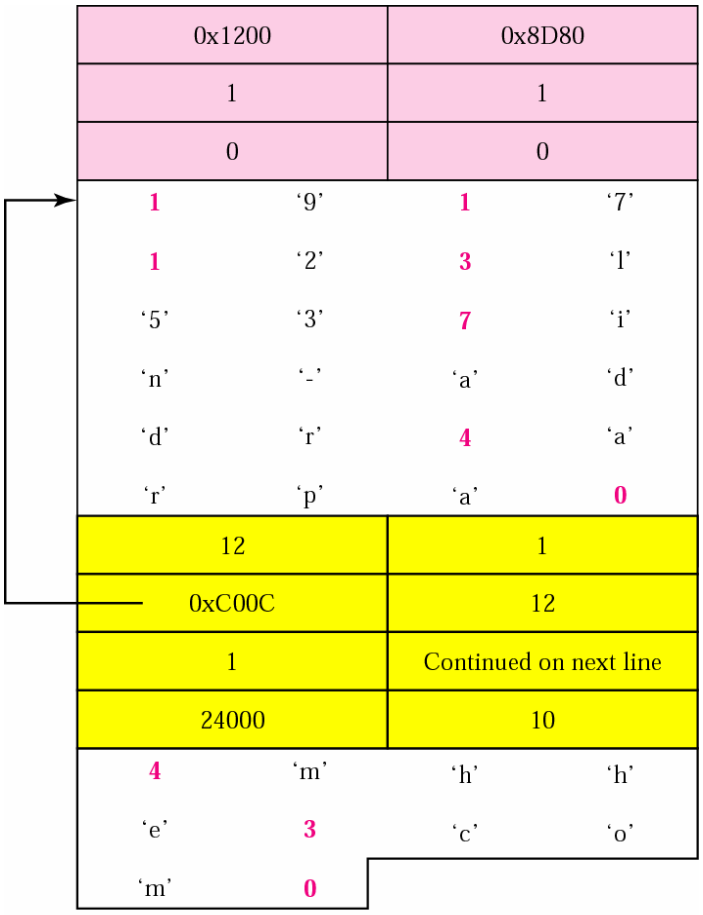
الشكل 22– المثال 2: رسالة استفسار معكوسة

- 0x1200: قيمة المعرف.
- 0x0900: بايتين يكتبان بالصيغة بالثنائية 0000100100000000 التي تدل على الحقول الجزئية التالية:

| QR | OpCode | AA | TC | RD | RA | Reserved | rCode |
|---|---|---|---|---|---|---|---|
| 0 | 0001 | 0 | 0 | 1 | 0 | 000 | 0000 |

- يعرف OpCode أن الاستفسار هو معكوس (حديثاً يجري استخدام OpCode=0000 حتى ولو كان الاستفسار معكوس)
- تحوي الرسالة سجل سؤال واحد.
- اسم النطاق هو 191712315374in-addr4arpa0.
- 12: نوع الاستفسار هو PTR
- 1: الصنف هو إنترنت

يظهر الشكل التالي رسالة الجواب.




الشكل 23– المثال 2: رسالة الجواب

- الراية هي 0x8D80: تكتب بالثنائي 1000110110000000 وتوزع على الحقول الجزئية التالية:

| QR | OpCode | AA | TC | RD | RA | Reserved | rCode |
|---|---|---|---|---|---|---|---|
| 1 | 0001 | 1 | 0 | 1 | 1 | 000 | 0000 |

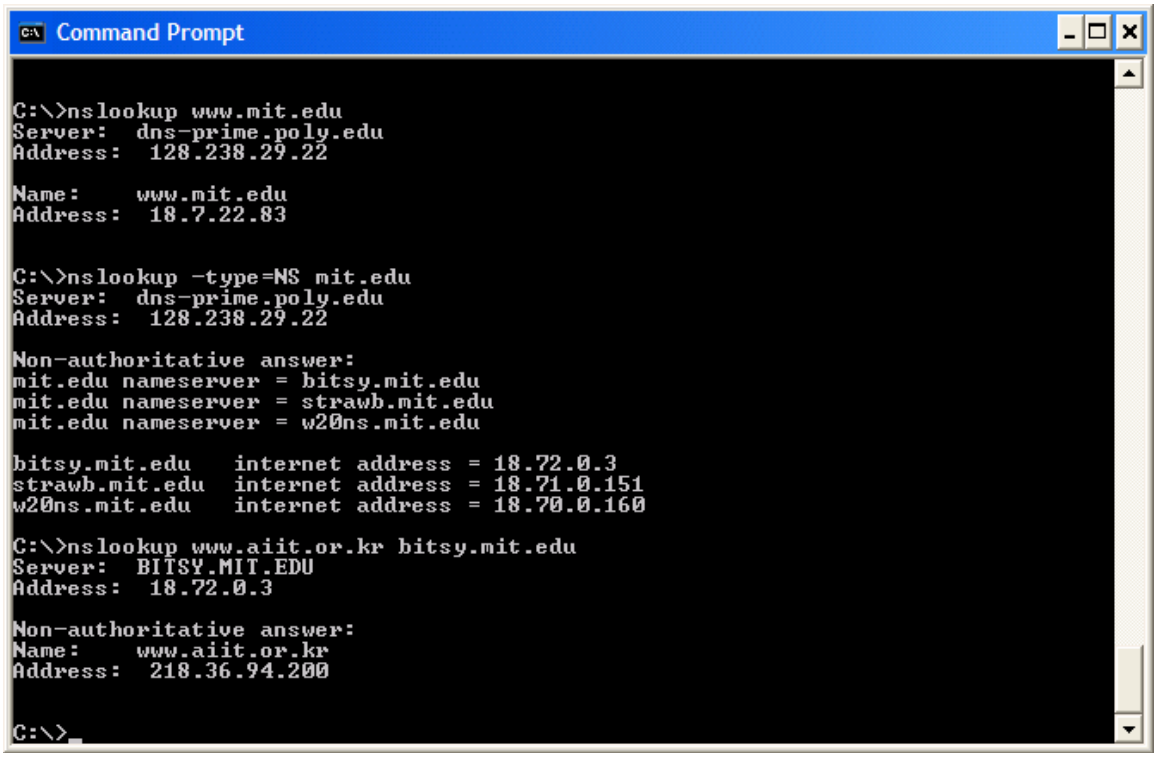
  - تحوي الرسالة على سجل سؤال واحد.
  - يكرر سجل السؤال من رسالة الاستفسار.
  - يحمل سجل الجواب القيمة 0xC00C التي تؤشر إلى سجل السؤال.
- 12: نوع النطاق هو PTR
- 1: الصنف هو الإنترنت.
- 24000 ثانية مدة الحياة.
- 10: طول معطيات المورد.
- اسم النطاق: 4mhhe3com0 أي mhhe.com

## 6. تمارين عملية

### 1.6. استخدام الأداة nslookup الموجودة ضمن سطر الأوامر cmd

التمرين الأول: سنقوم ضمن هذا التمرين باستخدام الأداة nslookup لإرسال طلب استفسار من مخدم نطاق أسماء.

تقوم الاداة nslookup بارسال طلب DNS إلى مخدم نطاق اسماء ومن ثمن تعرض النتائج كما هو موضح في الشكل التالي:




تظهر الصورة السابقة نتيجة ثلاثة طلبات وهي:

- ما هو عنوان IP المقابل للاسم www.mit.edu؟ نلاحظ أن النتيجة هي عنوان واسم مخدم الأسماء وعنوان IP المطلوب وهو 18.7.22.83
- من هو مخدم الأسماء صاحب السلطة للموقع mit.edu؟ نلاحظ أن الجواب يحوي 3 مخدمات وأن الجواب قادم من ذاكرة خابية وليس من المخدم صاحب السلطة بسبب كون الجواب Non-Authoritative answer كما يحوي الجواب أيضاً على عناوين IP لهذه المخدمات علماً أننا لم نطلبهم ضمن الاستفسار.
- نطلب في الاستفسار الأخير عنوان IP للاسم www.aiit.or.kr من مخدم الأسماء bitsy.mit.edu ونلاحظ أن الجواب هو 218.36.94.200

بالنسبة للصيغة النحوية الكاملة لاستفسار nslookup فهي:

```
nslookup -option1 -option2 host-to-find dns-server
```

المسائل المطلوبة: استخدام الأداة nslookup للاستفسار عن المعلومات التالية:

- عنوان IP للمخدم svuonline.org. و mail.svuonline.org. و www.google.com.
- معرفة أسماء وعناوين IP لمخدمي أسماء الجامعة الافتراضية وamazon.com
- معرفة canonical name للمخدم gmail.com
- معرفة الاسم المقابل لكل عنوان من العنوانين: 212.11.196.142 والعنوان 69.171.230.68
- معرفة مخدم البريد الالكتروني لكل من: www.gmail.com و svuonline.org
- تحديد مخدم الأسماء المحلي 8.8.8.8
- وضع الحلال nslookup بوضع التقلية set debug ومن ثم حل العنوان bbc.co.uk. ما هي المعلومات التي تحصل عليها؟

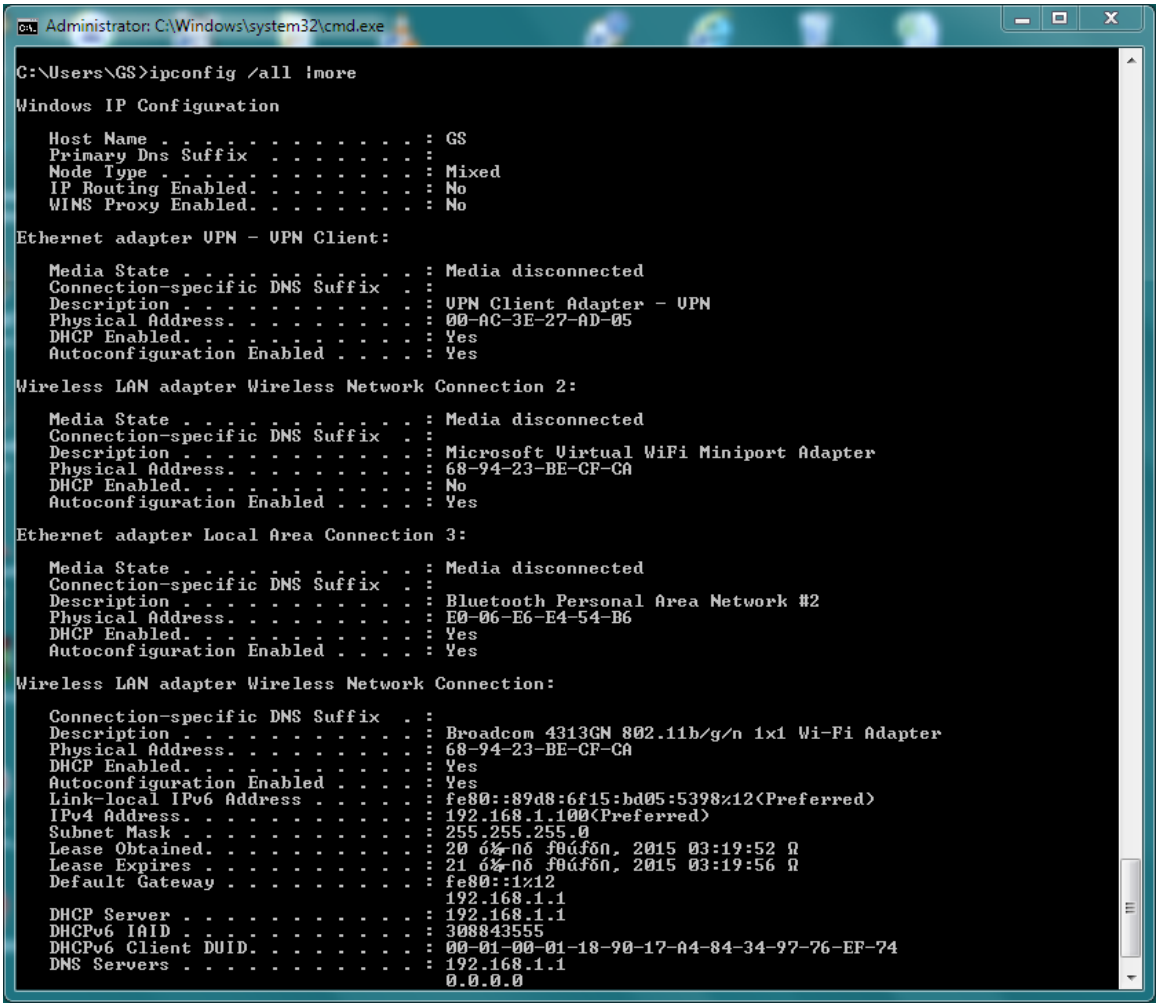
## 2.6. تحليل الطرود DNS

سنستخدم في هذا التمرين الأداة Wireshark والأمر ipconfig:

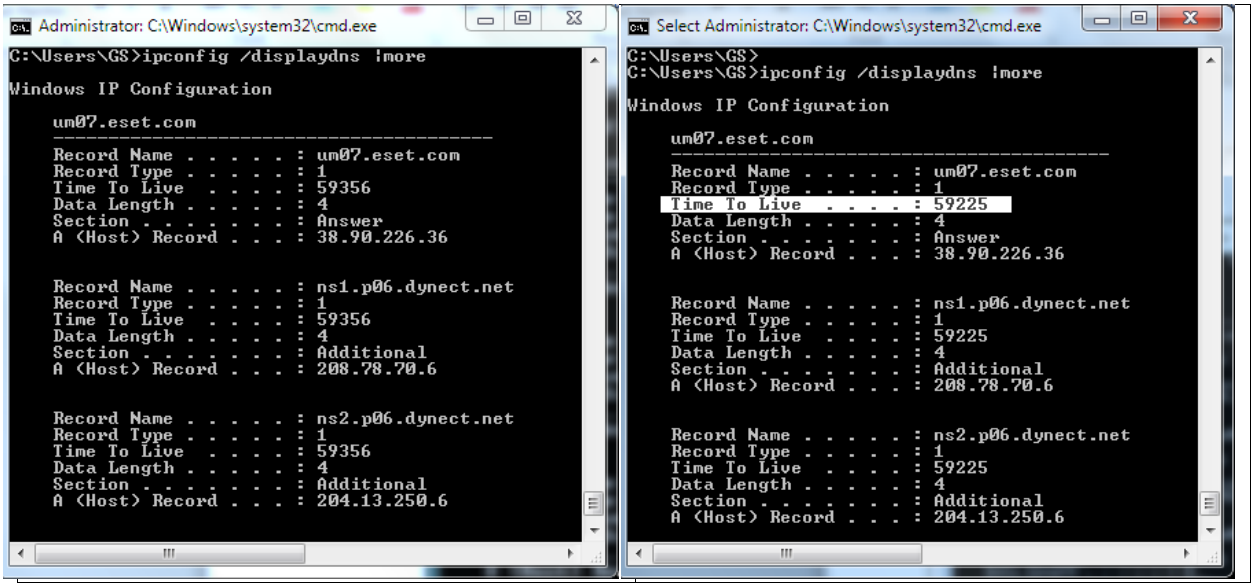
استخدام ipconfig

تعتبر الأداة ipconfig ضمن نظام ويندوز أو ifconfig ضمن نظام لينوكس من أهم الأدوات التي تسمح بمعرفة معلومات عن بروتوكولات TCP/IP ومنها عنوان IP لمخدم الأسماء.

```
ipconfig /all or ipconfig /all | more
```




يمكننا أيضاً استخدام ipconfig لإدارة معلومات DNS المخزنة ضمن للحاسب. فمثلاً إدخال ipconfig /displaydns



تعرض هذه الأداة جميع مدخلات DNS الموجودة ضمن الذاكرة الخابية ولكل مدخل تعرض المدة المتبقية TTL بالثانية. لاحظ تغير قيمة TTL بعد تنفيذ الأمر مرتين متعاقبتين.

```
ipconfig /flushdns
```

حذف جميع المدخلات الموجودة ضمن الذاكرة الخابية للحلال.

## استخدام Wireshark

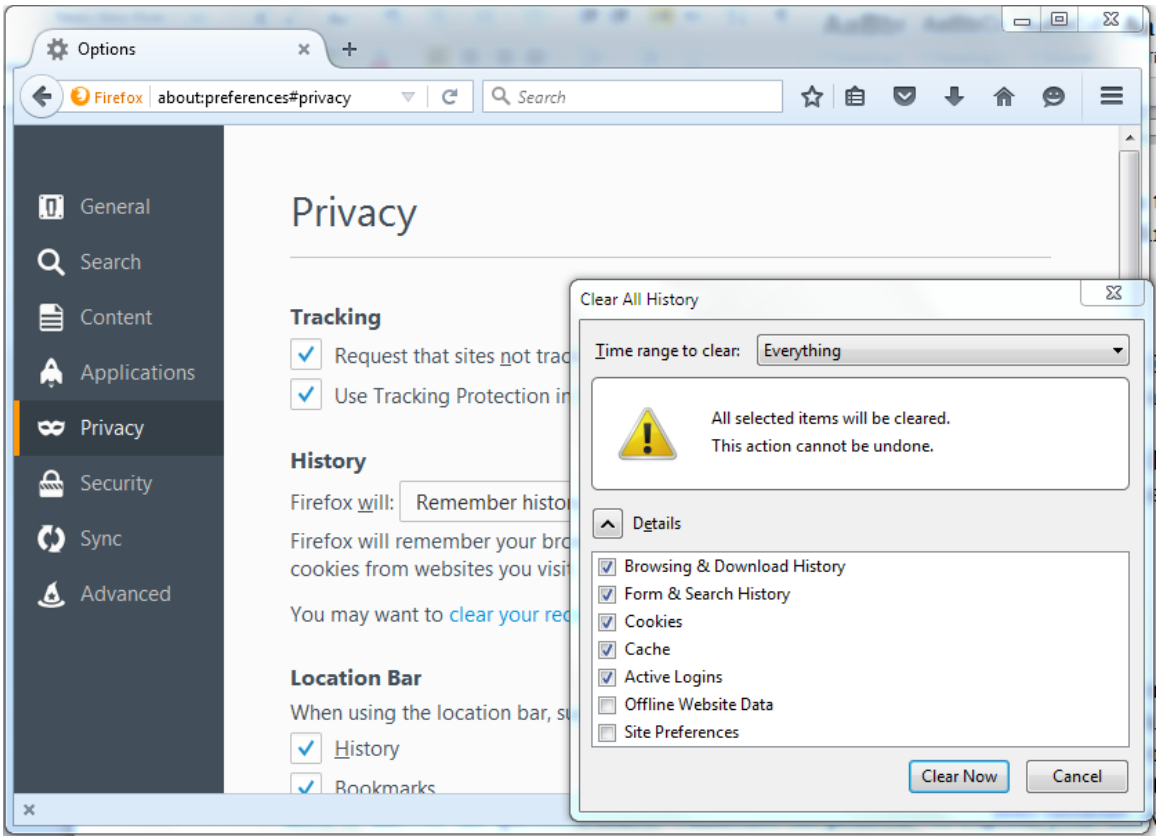
سنحاول الآن التقاط طرود DNS التي تحدث أثناء تصفح الإنترنت بدون تدخل المستخدم.

قم بالعمليات التالية:

حذف معلومات الذاكرة الخابية ipconfig /flushdns

حذف معلومات الذاكرة الخابية من متصفح الإنترنت. مثال عن Firefox :

Option > Privacy > Clear your recent history




شغل wireshark وضع ضمن Filter: ip.addr == your_IP_address والذي يمكن التعرف عليه عن طريق ipconfig /all.

أطلب العنوان www.ietf.org من المتصفح

أوقف التقاط الطرود

أجب على الأسئلة التالية:

أ- أبحث عن طلب DNS وجواب DNS ضمن ملف الأثر؟ هل يعمل بروتوكول DNS فوق UDP أو TCP؟

ب- ما هو عنوان بوابة الوجهة destination port لطلب الاستفسار وما هو رقم بوابة المصدر source port لرسالة الجواب؟

ت- إلى أي عنوان IP جرى إرسال رسالة الاستفسار؟ تأكد من عنوان DNS Server باستخدام ipconfig /all وتأكد أن الرسالة أرسلت له.

ث- افحص DNS query message. ما هو نوع query message? هل تحوي الرسالة أي جوابanswer؟

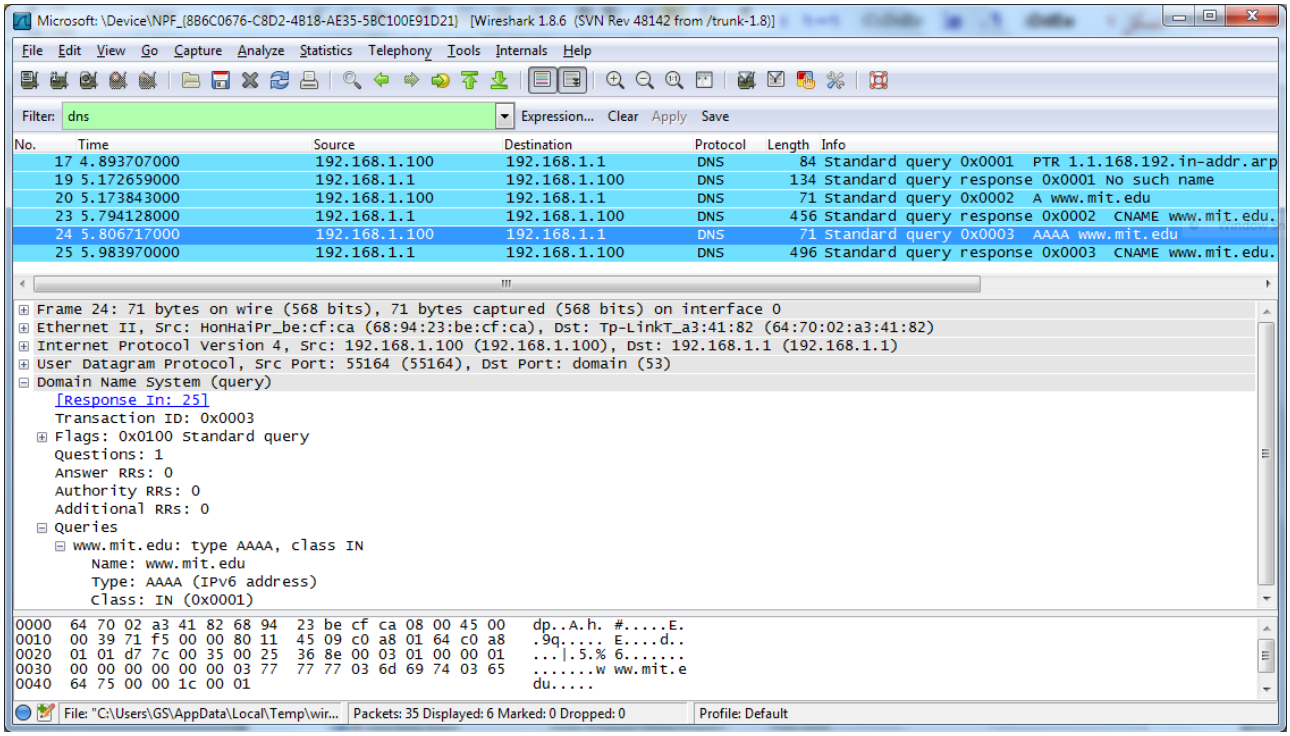
ج- افحص DNS Query response. ما هو عدد الإجابات الموجودة؟ وماذا تحوي كل إجابة منها؟

ح- قارن عنوان IP الموجود ضمن طرد TCP Sync مع العناوين الموجودة ضمن DNS query response. هل يوجد تطابق مع أي منها؟

خ- يحوي الموقع المزار بعض الصور. هل احتاج الحاسب إلى إعادة الاستفسار على عناوين IP المقابلة لمواقع هذه الصور؟

لنحاول الأن استخدام nslookup مع wireshark:

- أبدأ التقاط الطرود
- أدخل nslookup www.mit.edu
- أوقف التقاط الطرود
- ستشاهد ملف أثر يشبه التالي:




أول استفسارين DNS Request queries تطلبهما أداة nslookup هي طلبات غير معيارية ولا توجد ضمن بقية التطبيقات لذلك سنركز على الاستفسار الأخير.

أ- ما هي البوابة الوجهة لاستفسار DNS query request؟ وما هي البوابة المصدر للجواب DNS query response؟

ب- هل جرى إرسال DNS query message إلى عنوان IP لمخدم الأسماء المحلي؟

ت- ما هو نوع DNS query message؟ وهل تحوي أية أجوبة داخلها؟

ث- افحص DNS response message. ما عدد الإجابات المزودة ضمنها؟ وماذا تحوي كل إجابة منها؟

أعد الآن التجربة السابقة لكن باستخدام:

```
nslookup -type=NS mit.edu
```

أجب على الأسئلة التالية:

أ- ما هو عنوان IP التي وجهت له رسالة DNS query message؟ وهل هو عنوان Default DNS server لك؟

ب- افحص DNS query message. ما هو نوع DNS query؟ وهل يحوي الاستفسار على أية أجوبة؟

ت- افحص DNS response message. ما هو مخدم (مخدمات) الأسماء المزود (ة) ضمن الجواب؟ وهل يزود الجواب عناوين IP لمخدم (مخدمات) الأسماء؟

أعد الآن التجربة السابقة لكن باستخدام الأمر:

```
nslookup aol.com. 8.8.4.4
```

أجب على الأسئلة التالية:

أ- ما هو عنوان IP التي وجهت له رسالة DNS query message؟ وهل هو عنوان Default DNS server لك؟ إذا لا، إلى أي عنوان يدل عنوان IP المستخدم؟

ب- افحص DNS query message. ما هو نوع DNS query؟ وهل يحوي الاستفسار على أية أجوبة؟

ت- افحص DNS response message. ما عدد الإجابات المزودة؟ ماذا تحوي كل إجابة منهم؟

# الوب وبروتوكول HTTP

كان استخدام الإنترنت، حتى بداية التسعينيات، محصوراً بالباحثين والأكاديميين وطلاب الجامعات بغية النفاذ عن بعد إلى حواسب بعيدة أو لنقل الملفات أو لتبادل البريد الإلكتروني أو للمناقشة ضمن ما كان يعرف بخدمة الأخبار. جذب دخول الوب إلى الإنترنت انتباه عموم الناس وساهم في تغيير النظرة إلى الإنترنت وطرق التعامل معها الأمر الذي ساهم في تحويل الإنترنت من واحدة من الشبكات الموجودة إلى الشبكة الوحيدة المهيمنة.

ما يميز الوب هو كون الخدمة حسب الطلب on-demand. يستقبل المستخدمون ما يريدون وفي الوقت الذي يريدون بعكس الخدمات التي تقدمها وسائل بث الراديو والتلفاز التي تجبر المتلقين على المتابعة عندما يريد المصدر بدء البث. أضف إلى ذلك أن أي شخص بات قادراً أن يصبح ناشراً بدون كلفة كبيرة. كما ساعدت محركات البحث والنصوص المترابطة في تسهيل عملية الملاحة ضمن الإنترنت.

تجدر الإشارة هنا إلى انتشار بعض التطبيقات مثل YouTube, Gmail, and Facebook والتي جذبت أعداداً كبيرة من المستخدمين بعد عام 2000.

## 1. لمحة عن البروتوكول HTTP

يعتبر بروتوكول HyperText Transfer Protocol (HTTP)، وهو بروتوكول الوب الذي يعمل على مستوى طبقة التطبيقات، قلب الوب. يعرّف هذا البروتوكول ضمن [RFC 1945] و [RFC 2616]. جرى تحقيق HTTP باستخدام برنامجين: برنامج زبون وبرنامج مخدم. يتخاطب هذين البرنامجين مع بعضهما البعض باستخدام رسائل HTTP. يعرّف HTTP بنية الرسائل وطريقة تبادلها. لنذكر بعض المصطلحات المستخدمة ضمن الوب:

تتألف صفحة الوب Web page (أو الوثيقة) من مجموعة من الأغراض. الغرض هو ملف كملف HTML أو صورة JPEG أو مقطع فيديو أو بريمج Java applet والذي يمكن عنونته باستخدام Uniform Resource Locator (URL) وحيد.

تتألف معظم صفحات الوب من ملف HTML أساسي base HTML file ومجموعة من الأغراض المرتبطة بها referenced objects. فمثلاً، إذا كانت صفحة الوب تحوي ملف HTML وخمس صور JPEG فهذا يعني أن صفحة الوب تحوي ستة أغراض: الملف الأساسي مضاف إليه خمس صور. يؤشر الملف الأساسي إلى الصور الخمسة باستخدام URL لكل غرض.

بما أن تحقيق زبون الوب يكون عادةً ضمن متصفح الإنترنت مثل Firefox فيمكننا استخدام المصطلحين معاً للدلالة على زبون الوب.

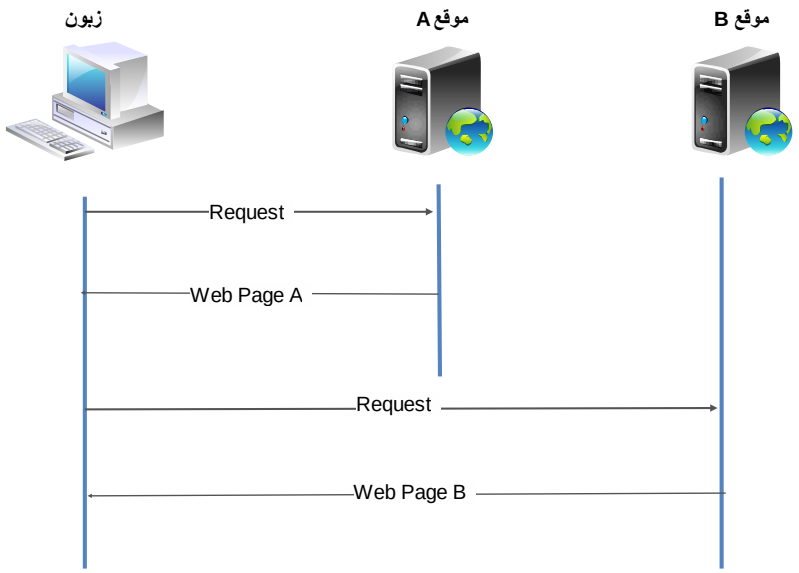
من الأمثلة عن مخدمات الوب المشهورة هناك Apache و Microsoft Internet Information Server (IIS).

عندما يضغط المستخدم على رابط ما أو يدخل عنوان URL ضمن متصفح الإنترنت فإن المتصفح يرسل رسالة طلب HTTP إلى المخدم. يستقبل المخدم رسالة الطلب ويعيد رسالة الجواب Response message والتي تحوي على الغرض المطلوب.

يعتمد بروتوكول HTTP على طبقة TCP لنقل المعلومات المتبادلة مع الزبون. يبدأ زبون HTTP أولاً بفتح اتصال TCP مع المخدم. بعد ذلك، تمر جميع الرسائل المتبادلة بين الطرفين عبر هذا الاتصال.

من المهم الانتباه إلى أن مخدم الوب لا يحتفظ بأي معلومات عن حالة الزبون بعد تخديمه. فإذا طلب الزبون الغرض نفسه مرتين متتاليتين، فالمخدم غير قادر على معرفة ذلك وسيعيد له الغرض المطلوب كل مرة. لذلك يعرف بروتوكول HTTP بأنه عديم الحالة Stateless protocol. يستخدم الوب نموذج زبون–مخدم كما أن المخدم يكون دائماً قيد العمل ومزود بعنوان IP ساكن وهو قادر على تخديم عدة مستخدمين في الوقت نفسه.

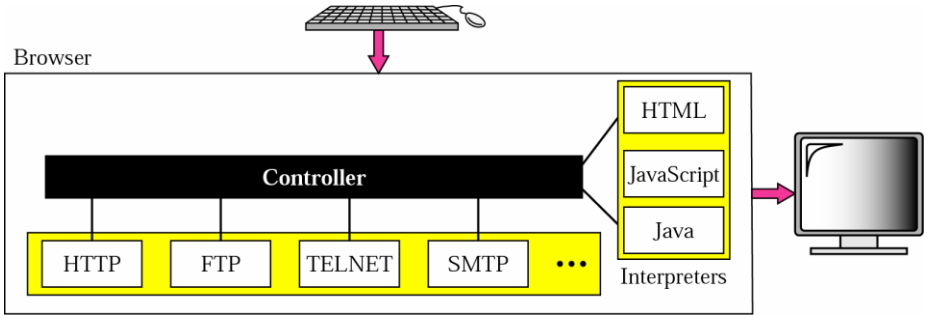
إن شبكة الوب هي خدمة موزعة من نوع مخدم–زبون، حيث يستطيع الزبون، عن طريق المتصفح، النفاذ إلى خدمة باستخدام المخدم. إلا أن الخدمة المزودة تكون موزعة على عدة مواقع كما هو موضح في الشكل التالي.




الشكل 1– خدمة الوب الموزعة

## 1.1. الزبون (المتصفح)

يعتبر المتصفح مسؤولاً عن تفسير وعرض صفحات الوب. يتألف كل متصفح من ثلاثة أجزاء: متحكم Controller وبروتوكول الزبون ومفسر interpreter. يستقبل المتحكم الدخل من لوحة المفاتيح أو الفأرة ويستخدم برامج الزبون للنفاذ إلى الوثيقة. يستخدم بعد ذلك المتحكم أحد المفسرات المتوفرة لعرض الوثيقة على الشاشة. يمكن أن يكون بروتوكول الزبون FTP أو Telnet أو HTTP. أما المفسر، فيمكن أن يكون HTML أو جافا أو Javascript حسب نوع الوثيقة. يبين الشكل التالي تصميماً توضيحياً لبنية المتصفح.




الشكل 2– بنية المتصفح

### 2.1. المخدم

تُخزن صفحة الوب على المخدم. في كل مرة يرسل الزبون لطلب ما فإن المخدم يقوم بإرسال الصفحة المقابلة. تخزن المخدمات عادةً الصفحات المطلوبة ضمن ذاكرة خابية لتحسين الأداء. يمكن أيضاً زيادة فعالية المخدم باستخدام تقانات مثل multithreading أو multitasking حيث يصبح المخدم قادراً على معالجة عدة طلبات في الوقت نفسه.

### 3.1.عنوان مورد نظامي Uniform Resource Locator (URL)

يحتاج الزبون إلى عنوان صفحة وب حتى يستطيع النفاذ إليها. يستخدم بروتوكول HTTP مفهوم العنوان Locator لتسهيل عملية النفاذ إلى الوثائق الموزعة على العالم. يعتبر عنوان URL معياراً لتوصيف أي نوع من الوثائق الموجودة على الإنترنت. يعرف عنوان URL أربعة أشياء: البروتوكول والحاسب المضيف والبوابة والمسار. يأخذ URL الصيغة التالية:

```
Protocol://Host:Port/Path
```

البروتوكول هو برنامج مخدم–زبون المستخدم لاستحضار الوثيقة. يوجد عدة بروتوكولات قادرة على استحضار الوثائق مثل Gopher, FTP, HTTP, News, Telnet. لكن HTTP هو الأكثر شهرةً.

المضيف host هو الحاسب الموجودة عليه الوثيقة أو المعلومة المطلوبة. يمكننا هنا استخدام اسم مستعار alias للمضيف يبدأ عادةً بـ WWW لكن ذلك غير إجباري.

يمكن أن تحوي URL على رقم بوابة المخدم.

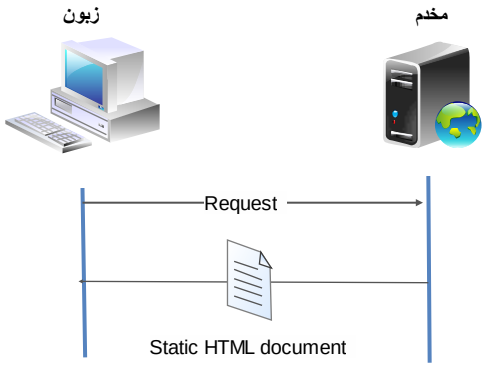
المسار path هو اسم المسار للوصول إلى الملف الذي يحوي الوثيقة المطلوبة. لاحظ أنه يمكن للمسار أن يحوي على slashes التي تفصل، حسب نظام التشغيل UNIX، الدليل عن الدلائل الفرعية أو الملفات.

## 4.1. وثائق الوب

يمكن تصنيف الوثائق ضمن شبكة الوب إلى ثلاثة أصناف: الساكنة static والديناميكية dynamic والنشطة active. يعتمد التصنيف على زمن تحديد محتوى الوثيقة.

### الوثائق الساكنة Static documents

الوثائق الساكنة هي وثائق ثابتة المحتوى، يجري توليدها وتخزينها على المخدم. يمكن للزبون الحصول على نسخة من الوثيقة فقط. أي أنه يجري تحديد محتوى الوثيقة لحظة توليدها وليس لحظة استخدامها. طبعاً، يستطيع المخدم فقط تغيير محتوى الوثيقة أما الزبون فيستطيع، باستخدام المتصفح، عرض محتويات الوثيقة (انظر الشكل التالي).




الشكل 3– الوثائق الساكنة

### لغة تأشير نص ترابطي Hypertext Markup Language (HTML)

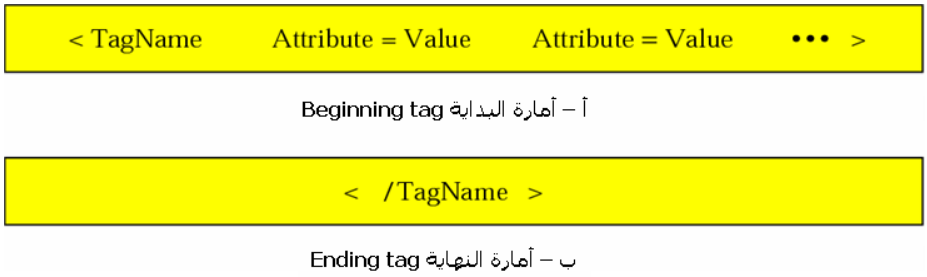
HTML هي لغة تسمح بإنشاء صفحات الوب. يفيد التأشير في صياغة صفحة الوب ليستطيع المتصفح تفسيرها ومن ثم عرضها.

تسمح لغات التأشير مثل HTML بتضمين تعليمات صياغة ضمن النص نفسه. بهذه الطريقة، يستطيع أي متصفح قراءة التعليمات وصياغة النص حسب محطة العمل.

تسمح HTML باستخدام محارف ASCII لكتابة النص نفسه ولتعليمات الصياغة.

تتألف صفحة الوب من قسمين: الرأس والمتن. يحوي الرأس عنوان الصفحة وبعض المتحولات التي يمكن للمتصفح استخدامها. يوضع المحتوى الفعلي للصفحة ضمن المتن الذي يحوي النص والأمارات tags. تُعرِّف الأمارات طريقة إظهار النص. تتكون كل أمارة من اسم ومجموعة اختيارية من الخاصيات، توضع جميعاً ضمن إشارتي أصغر وأكبر (< and >).

عندما توجد الخاصية attribute فإنه يليها إشارة المساواة (=) وقيمة الخاصية. يمكن استخدام بعض الأمارات بشكل وحيد بينما يُستخدم البعض الآخر كزوج (أمارة البداية وأمارة النهاية). يمكن لأمارة البداية أن تمتلك خاصيات أو قيم وتبدأ باسم الأمارة. لا يمكن لأمارة النهاية أن تمتلك خاصيات أو قيم لكن يجب وضع slash (/) قبل اسم الأمارة. يبين الشكل التالي صيغة الأمارة.




الشكل 4 – أمارات البداية والنهاية

من أشهر الأمارات المستخدمة لصياغة النصوص:

- <B> و</B> لإظهار النص بالبنط العريض.
- <I> و</I> لإظهار النص بالبنط المائل.
- <U> و</U> لإظهار النص تحته خط.

### الوثائق الديناميكية

يجري توليد الوثيقة الديناميكية من قبل مخدم الوب عندما يطلب المتصفح الصفحة. عندما يصل الطلب، ينفذ المخدم برنامج تطبيقي ما أو سكريبت يسمح بتوليد الصفحة الديناميكية. يعيد المخدم خرج البرنامج أو خرج السكريبت كجواب على طلب المتصفح. بما أنه يجري توليد صفحة وب جديدة لكل طلب، فيمكن أن تكون الصفحات مختلفة من طلب إلى آخر. لعل أفضل مثال على ذلك، متصفحات تطلب التاريخ والوقت من مخدم.

### واجهة العبّارة المشتركة Common Gateway Interface (CGI)

إن واجهة العبّارة المشتركة هي تقانة تسمح بتوليد ومعالجة الوثائق الديناميكية. CGI هي مجموعة من المعايير التي تُعرِّف كيف يجري كتابة وثيقة ديناميكية، وكيف يجري إدخال المعطيات إلى البرنامج، وكيف يجري استخدام نتائج الخرج.

تعرف CGI مجموعة من القواعد والمصطلحات التي يجب على المبرمج احترامها.

### الوثائق النشطة Active documents

نحتاج في العديد من التطبيقات إلى تنفيذ برنامج أو سكريبت على موقع الزبون. هذا ما يدعى بالوثيقة النشطة. لنأخذ على سبيل المثال تنفيذ برنامج لتوليد بيانيات حية animated graphics على الشاشة أو برنامج آخر يتفاعل مع المستثمر. عندما يطلب المتصفح وثيقة نشطة فإن المخدم يرسل نسخة عن الوثيقة أو السكريبت. يجري الآن تنفيذ الوثيقة على حاسب الزبون (المتصفح).

## 2. بروتوكول HTTP

يُستخدم بروتوكول HTTP بشكل أساسي للنفاذ إلى المعطيات الموجودة على الشبكة العنكبوتية. يعمل بروتوكول HTTP كمزيج من FTP و SMTP. يشبه FTP لأنه ينقل الملفات ويستفيد من خدمات TCP. لكنه أبسط بكثير من FTP لأنه يستخدم ارتباط TCP واحد لنقل المعطيات بين المخدم والزبون.

يشبه أيضاً بروتوكول SMTP لأن المعطيات المنقولة بين المخدم والزبون تشبه رسائل SMTP. لكن بعكس SMTP فإن الرسائل غير معدة للقراءة من قبل الإنسان؛ لأنها تُقرأ وتُفسر من قبل مخدم أو زبون HTTP إضافةً إلى إرسال الرسائل مباشرةً دون تخزين وإعادة توجيه كما هو عليه الحال ضمن SMTP.

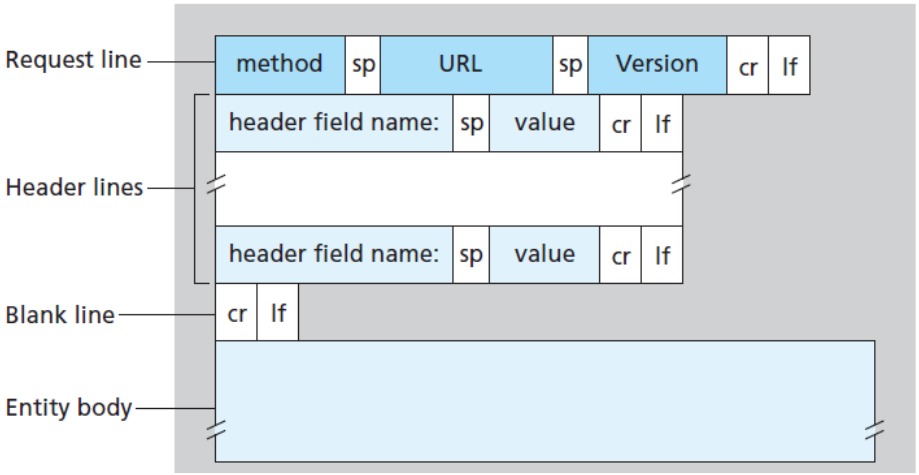
تُضمَّن الطلبات من الزبون إلى المخدم ضمن رسالة طلب Request message بينما يُضمَّن محتوى الملف المطلوب ضمن رسالة الجواب Response message. يستخدم HTTP خدمات TCP على البوابة المعروفة 80.

### 1.2. مداولات HTTP

مع أن HTTP يستخدم خدمات TCP إلا أنه بروتوكول عديم الحالة Stateless. يهيئ الزبون المداولة عن طريق إرسال رسالة الطلب ويرد المخدم عن طريق إرسال الجواب.

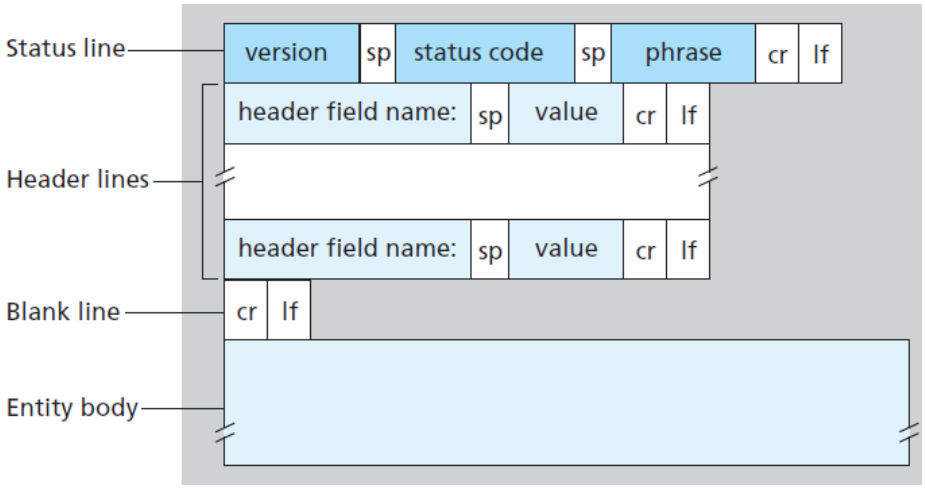
الرسالة

يبين الشكل التالي صيغة الطلب.




الشكل 5– رسالة الطلب

ويبين الشكل التالي رسالة الجواب:

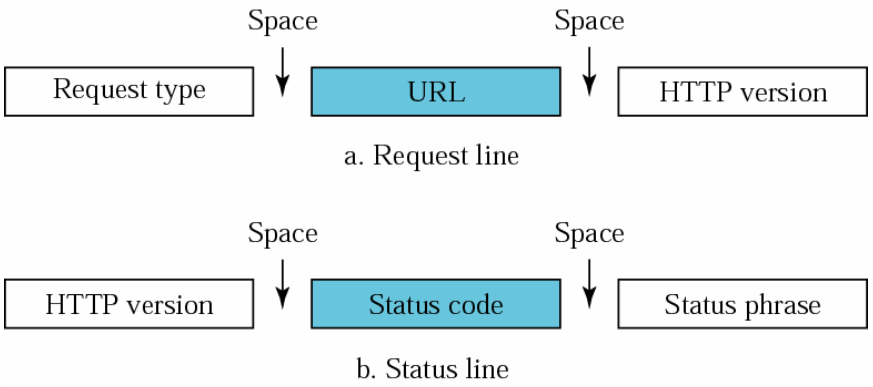



الشكل 6 – رسالة الجواب

تتكون رسالة الطلب من سطر الطلب وترويسة وفي بعض الأحيان، متن. أما رسالة الجواب فتتكون من سطر الحالة وترويسة وفي بعض الاحيان، متن.

## أسطر الطلب والحالة Request and Status lines

يبين الشكل التالي صيغة هذين السطرين.




الشكل 7 – أسطر الطلب والجواب

- نوع الطلب request type. جرى تعريف عدة أنواع خدمة ضمن HTTP 1.1. يُصنف نوع الخدمة حسب طرائق methods كما هو مبين في الشكل التالي.

| Method | Action |
|---|---|
| GET | طلب وثيقة من المخدم |
| HEAD | طلب معلومات عن وثيقة لكن ليس الوثيقة نفسها |
| POST | إرسال الزبون لبعض المعلومات على المخدم |
| PUT | إرسال المخدم لوثيقة إلى الزبون |
| TRACE | إعادة echo الطلب الداخل |
| CONNECT | محجوزة |
| OPTION | الاستفسار عن الاختيارات المتاحة |

الشكل 8– الطرائق methods

- URL. أصبح معروفاً الآن.
- Version. أحدث إصدار للبروتوكول هو HTTP1.1.
- رمز الحالة Status code. يستخدم هذا الحقل ضمن رسالة الجواب. يتألف رمز الحالة من 3 أرقام. تكون جميع الرموز ضمن مجال 100 هي رموز إعلامية informational فقط؛ تشير الرموز ضمن مجال 200 إلى نجاح الطلب؛ تعيد الرموز ضمن مجال 300 توجيه الزبون إلى عنوان URL آخر؛ تشير الرموز ضمن مجال 400 إلى خطأ في طرف الزبون وضمن المجال 500 إلى خطأ في طرف المخدم. يبين الجدول التالي الرموز الأكثر شيوعاً.

| Code | Phrase | الشرح |
|---|---|---|
| **Informational** | | |
| 100 | Continue | جرى استقبال القسم الأول من الطلب ويمكن للزبون المتابعة في طلبه |
| 101 | Switching | يستجيب المخدم لطلب الزبون لتبديل البروتوكولات المعرفة داخل ترويسة التحديث |
| **Success** | | |
| 200 | OK | نجاح الطلب |
| 201 | Created | لقد جرى توليد URL جديدة |
| 202 | Accepted | تم قبول الطلب، لكن لا يجر تنفيذها حالياً |

| | | |
|---|---|---|
| 204 | No content | لا يوجد محتوى ضمن المتن |
| **Redirection** | | |
| 301 | Moved permanently | لم يعد المخدم يستخدم الـ URL المطلوبة |
| 302 | Moved temporarily | تم نقل الـ URL المطلوبة مؤقتاً |
| 304 | Not modified | لم يجر تعديل الوثيقة |
| **Client error** | | |
| 400 | Bad request | يوجد خطأ في التركيب النحوي للطلب |
| 401 | Unauthorized | ينقص الطلب أذن الدخول |
| 403 | Forbidden | الخدمة ممنوعة |
| 404 | Not found | الوثيقة غير موجودة |
| 405 | Method not allowed | الطريقة غير مدعومة ضمن هذه الـ URL |
| 406 | Not accepted | الصيغة المطلوبة غير مقبولة |
| **Server error** | | |
| 500 | Internal server error | يوجد خطأ على موقع المخدم أو كونه معطل |
| 501 | Not implemented | لا يمكن تحقيق العملية المطلوبة |
| 503 | Service unavailable | الخدمة غير متاحة مؤقتاً، لكن يمكن إعادة الطلب لاحقاً |

الشكل 9– رموز الحالة

### الترويسة Header

تستخدم الترويسة لتبادل معلومات إضافية بين المخدم والزبون، كأن يطلب الزبون إرسال الملف بصيغة خاصة أو أن يرسل المخدم معلومات إضافية عن الملف. تتكون الترويسة من سطر أو أكثر. يحوي كل سطر اسم الترويسة وعلامة الترقيم ":" وفراغ وقيمة للترويسة. أي Header name: Header value.

ينتمي سطر الترويسة إلى أحد الأصناف التالية: ترويسة عامة، ترويسة طلب، ترويسة جواب، ترويسة كيان. يمكن أن تحوي رسالة الطلب Request message إما الترويسة العامة أو الطلب أو الكيان. أما رسالة الجواب فيمكن أن تحوي الترويسة العامة أو الجواب أو الكيان.

- الترويسة العامة General header. تعطي معلومات عامة عن الرسالة. يبين الشكل التالي بعض الترويسات العامة.

| Header | الوصف |
|---|---|
| Cache-control | توصيف معلومات عن caching |
| Connection | إظهار إذا كان يجب إغلاق الارتباط أو لا |
| Date | إظهار التاريخ الحالي |
| MIME-version | إظهار إصدار MIME |
| Upgrade | توصيف بروتوكول الاتصال المفضل |

الشكل 10– الترويسات العامة

- ترويسة الطلب Request header. تحدد إعدادات الزبون وصيغة الوثائق التي يفضلها الزبون. يبين الشكل التالي بعض ترويسات الطلب.

| Header | الوصف |
|---|---|
| Accept | إظهار صيغة الوسائط التي يقبلها الزبون |
| Accept-charset | إظهار طقم المحارف التي يستطيع الزبون معالجتها |
| Accept-encoding | إظهار الترميز الذي يستطيع الزبون معالجته |
| Accept-language | إظهار اللغة التي يستطيع الزبون قبولها |
| Authorization | إظهار السماحيات التي يمتلكها الزبون |
| From | إظهار العنوان البريدي الإلكتروني للمستثمر |
| Host | إظهار اسم المضيف ورقم البوابة للزبون |
| If-modified-since | أرسل الوثيقة إذا كانت أحدث من تاريخ محدد |
| If-match | أرسل الوثيقة إذا كانت تطابق أمارة محددة |
| If-not-match | أرسل الوثيقة إذا كانت لا تطابق أمارة محددة |
| If-range | أرسل الجزء الناقص من الوثيقة |
| If-unmodified-since | أرسل الوثيقة إذا لم تتغير منذ التاريخ المحدد |
| Referrer | توصيف URL للوثيقة المرتبطة |
| User-agent | تعريف برنامج الزبون |

الشكل 11– ترويسات الطلب

- ترويسة الجواب Response header. تحدد إعدادات المخدم ومعلومات خاصة عن الطلب. يبين الشكل التالي بعض ترويسات الجواب.

| Header | الوصف |
|---|---|
| Accept-range | إظهار إذا كان المخدم يقبل المجال الذي حدده الزبون |
| Age | إظهار عمر الوثيقة |
| Public | إظهار لائحة بالطرائق المدعومة |
| Retry-after | تحديد الوقت الذي بعده يصبح المخدم متاح |
| Server | إظهار اسم المخدم ورقم الإصدار |

الشكل 10– ترويسات الجواب

- ترويسة الكيان Entity header. تعطي ترويسة الكيات معلومات عن متن الوثيقة. توجد هذه الترويسة غالباً ضمن رسائل الجواب غير أن بعض رسائل الطلب التي تحوي متن مثل PUSH و PUT تستخدم هذه الترويسة. يبين الشكل التالي بعض ترويسات الكيان.

| Header | الوصف |
|---|---|
| Allow | إدراج الطرائق المقبولة التي يمكن استخدامها مع URL |
| Content-encoding | تحديد طريقة الترميز |
| Content-language | تحديد اللغة |
| Content-length | تحديد طول الوثيقة |
| Content-range | تحديد مجال الوثيقة |
| Content-type | تحديد نوع الوسائط |
| Etag | إعطاء أمارة للكيان Entity tag |
| Expires | تحديد تاريخ ووقت انتهاء صلاحية المحتوى |
| Last-modified | تحديد تاريخ ووقت التعديل الأخير |
| Location | تحديد موضع الوثيقة المولدة أو المنقولة |

الشكل 11– ترويسات الكيان

- المتن Body. يمكن أن يكون المتن موجوداً ضمن رسالة الطلب أو رسالة الجواب. يحوي عادةً المعلومات المطلوب إرسالها أو توصيلها.

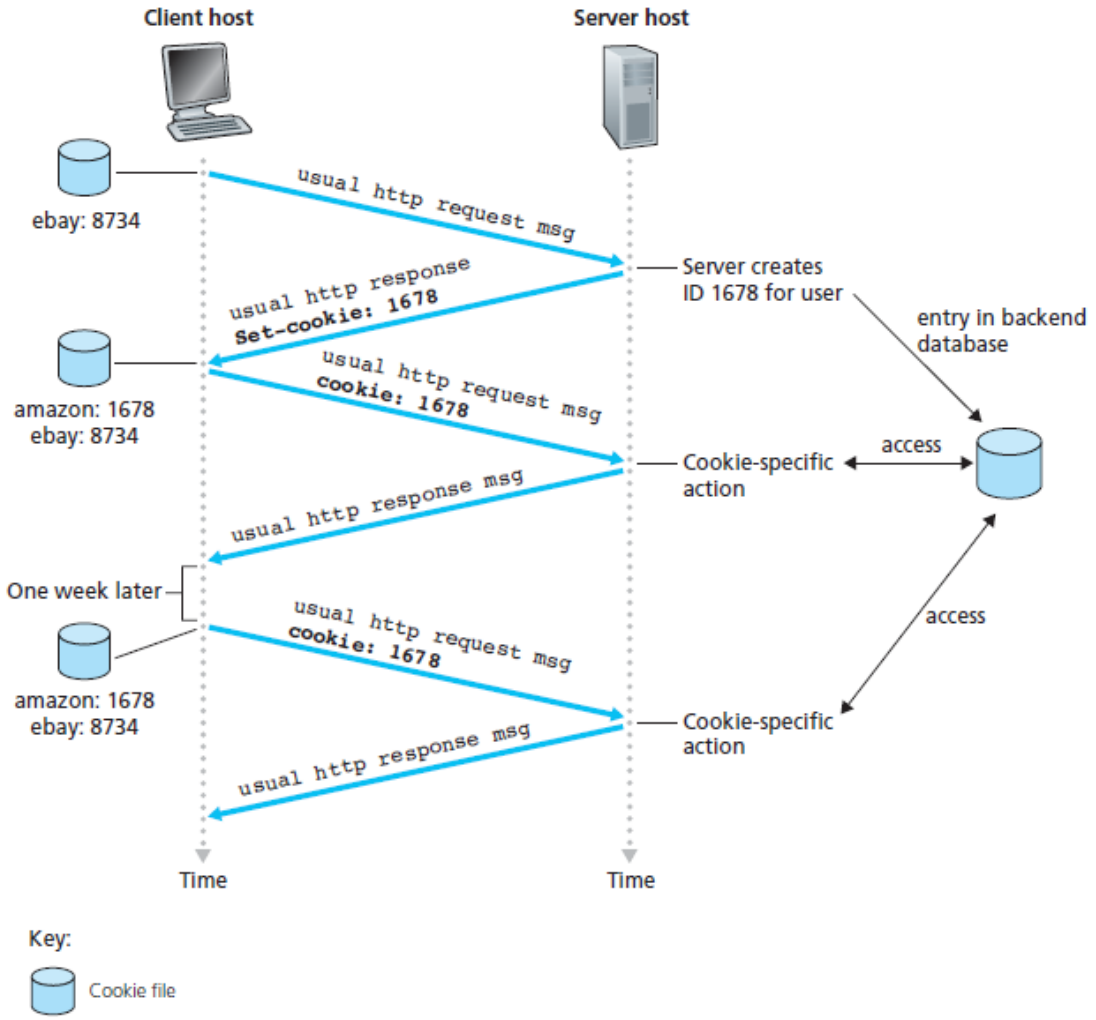
### 2.2. استخدام Cookies

لقد قلنا سابقاً أن بروتوكول HTTP هو عديم الحالة، طبعاً هذا يفيد في تبسيط العمليات التي يقوم بها الأمر الذي يحسن أداء المخدم مما يسمح له بمعالجة آلاف الطلبات في الوقت نفسه. لكن مخدم الوب، في بعض الحالات، يحتاج إلى التعرف على المستثمرين ومتابعتهم بغية توجيه المحتوى حسب المستثمر مثلاً. لذلك تقوم مخدمات HTTP باستخدام cookies. تسمح الكوكيز، حسب المعيار [RFC 6265]، للمواقع بمتابعة المستثمرين. كما أن معظم مواقع الإنترنت التجارية الحالية تستخدم الكوكيز.

تتألف تقنية الكوكيز من أربعة مكونات:

- وضع الكوكي ضمن سطر ترويسة رسالة الجواب
- وضع ترويسة الكوكي ضمن رسالة الطلب
- يحتفظ متصفح الإنترنت بالكوكي ويقوم بإدارتها
- يقوم مخدم الوب بإضافة مدخل ضمن قاعدة المعطيات خاصة بهذه الكوكي

يبين الشكل التالي تسلسل عمليات إنشاء الكوكيز ومتابعتها ضمن المتصفح أو المخدم.




الشكل 14– متابعة حالة المستثمر باستخدام الكوكيز

نلاحظ من الشكل السابق أن المستثمر كان يملك كوكي سابقة برقم 8734 تابعة لموقع ebay ثم دخل إلى موقع amazon الذي خلق كوكي جديدة برقم 1678 وأضافها إلى رسالة الجواب set-cookie: 1678. عندما يعاود المستثمر الدخول من المتصفح نفسه إلى موقع amazon بعد أسبوع مثلاً فإن المتصفح (زبون HTTP) يرسل مع رسالة الطلب HTTP Request message رقم الكوكي المرتبطة بموقع amazon وهي Cookie: 1678. عندها يستطيع المخدم التعرف على المستثمر وعلى عملياته السابقة وإعادة المحتوى المناسب له.

### 3.2.المخدم الوكيل Proxy server وتخبئة الوب Web caching

يدعم بروتوكول HTTP المخدم الوكيل. المخدم الوكيل هو حاسوب يحفظ نسخة عن أجوبة الطلبات الحديثة. يرسل زبون HTTP طلب إلى المخدم الوكيل الذي يقوم باختبار الذاكرة الخابية لديه فإذا كان الجواب غير مخزن

ضمن الذاكرة الخابية فإنه يرسل الطلب إلى المخدم المطلوب. يجري أيضاً إرسال الأجوبة القادمة إلى المخدم الوكيل الذي يخزنها من أجل الطلبات المقبلة من زبائن أخرى.

يقلص المخدم الوكيل العبء عن المخدمات الأصلية ويقلل حركة المرور ويقلل زمن الاستجابة. يتطلب استخدام المخدم الوكيل إعداد الزبون للنفاذ إلى المخدم الوكيل بدلاً عن مخدمات الإنترنت.

### الجلب الشرطي Conditional Get

تتمثل إحدى المشاكل التي تطرحها عملية التخبئة في التأكد من كون النسخة الموجودة لدى مخدم التخبئة ما تزال صالحة ولم تتغير بعد عند المخدم الأصلي. يوجد ضمن بروتوكول HTTP آلية تسمح لمخدم التخبئة أو لأي زبون وب من التأكد من كون النسخة التي بحوزتهما ما تزال صالحة. تدعى هذه الآلية بالجلب الشرطي Conditional Get. يدعى طلب HTTP بالاسم Conditional Get إذا (1) استخدم الطلب طريقة Get و (2) احتوت رسالة الطلب على سطر الترويسة If-Modified-Since.

لنأخذ المثال التالي: لنفرض أن المخدم الوكيل Proxy Server، الذي يعمل عادة كمخدم تخبئة، أرسل الطلب التالي إلى مخدم الوب:

```
GET /fruit/kiwi.gif HTTP/1.1
Host: www.exotiquecuisine.com
```

ثم أرسل مخدم الوب رسالة الجواب التالية:

```
HTTP/1.1 200 OK
Date: Sat, 8 Oct 2011 15:39:29
Server: Apache/1.3.0 (Unix)
Last-Modified: Wed, 7 Sep 2011 09:23:24
Content-Type: image/gif
(data data data data data ...)
```

يرسل مخدم التخبئة الرسالة إلى المتصفح الطالب ويحتفظ بنسخة ضمن الخابية المحلية. تحتفظ الخابية أيضاً بالتاريخ الموجود ضمن سطر الترويسة Last-Modified مع الرسالة. إذا طلب متصفح ما الصفحة نفسها بعد عدة أيام وإذا كانت الصفحة ما تزال ضمن الخابية فإن مخدم التخبئة يقوم بفحص صلاحية النسخة المحلية عن طريق إرسال طلب Conditional Get على الشكل التالي:

```
GET /fruit/kiwi.gif HTTP/1.1
Host: www.exotiquecuisine.com
If-modified-since: Wed, 7 Sep 2011 09:23:24
```

عندما يستلم مخدم الوب رسالة Conditional Get فإنه يتأكد فيما إذا كانت الصفحة قد تغيرت منذ التاريخ المطلوب وفي حالة الإيجاب يرسل الصفحة الحديثة أما إذا لم تتغير الصفحة فإن مخدم الوب يرسل المعلومات التالية:

```
HTTP/1.1 304 Not Modified
Date: Sat, 15 Oct 2011 15:39:29
Server: Apache/1.3.0 (Unix)
(empty entity body)
```

لاحظ أن مخدم الوب لا يرسل الصفحة المطلوبة إذا لم تتغير الأكر الذي من شأنه تقليل هدر الموارد بدون حاجة وتقليل زمن وصول الصفحة إلى الزبون وخاصةً إذا كان حجم الصفحة كبير.

تجدر الإشارة هنا إلى أن المخدم الأصلي يحدد مع كل جواب مدة صلاحية الجواب عن طريق الترويسة Expires: أو Max-age ويستطيع مخدم التخبئة معرفة حداثة المعلومة المخزنة بالعودة إلى أحدهما علماً أن Max-age هو أقوى من Expires.

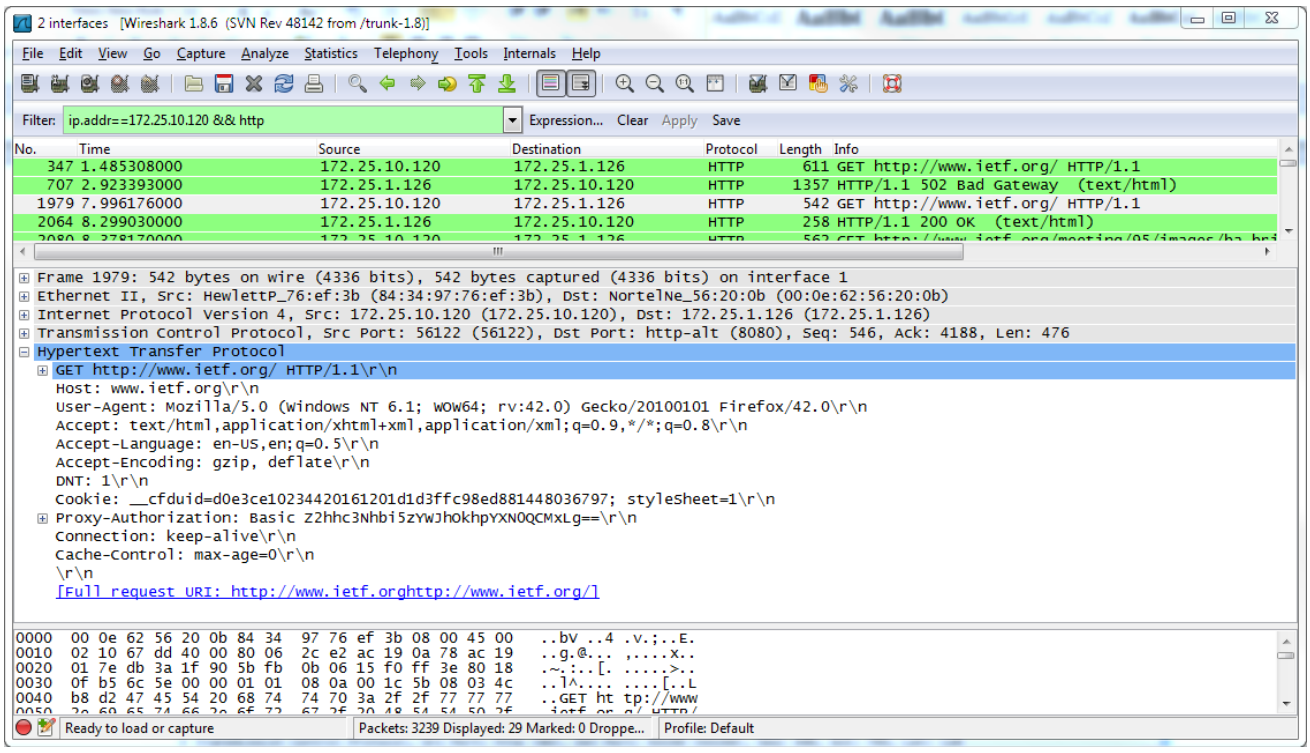
## 3. عملي

### 1.3. رسائل Get/response

سنحاول في هذا التمرين تحليل طريقتي Get وresponse باستخدام أداة wireshark كما يلي:

1. افتح متصفح الإنترنت
2. شغل wireshark وضمن Filter ضع http
3. افتح الصفحة www.ietf.org
4. أوقف تحليل الطرود ضمن wireshark

ستكون صفحة wireshark مشابهة للشكل التالي



أجب على الأسئلة التالية بعد النظر ضمن طرد السؤال Get من المتصفح إلى المخدم وطرد الجواب Response من المخدم إلى المتصفح:

1. هل يستخدم متصفحك النسخة HTTP 1.0 or 1.1؟ وما هي النسخة التي يستخدمها المخدم؟
2. ما هي اللغة التي يقبل استخدامها متصفحك؟
3. ما هو عنوان IP لحاسبك؟ وللمخدم؟
4. ما هي قيمة Status code التي أرجعها المخدم؟
5. متى جرى آخر تعديل على الصفحة المطلوبة؟
6. ما عدد البايتات التي أرجعها المخدم إلى المتصفح؟

## 2.3. استخدام HTTP conditional Get/response

حتى تنفذ التمرين بالشكل الصحيح، يجب عليك أولاً حذف معلومات المتصفح الموجودة ضمن الذاكرة الخابية عن طريق تنفيذ ما يلي: Tools-> Clear Recent History ومن ثم اختيار Cache box بالنسبة للمتصفح Firefox وTools-> Internet Options->Delete File بالنسبة للمتصفح IE. يمكنك الآن البدء بالتمرين:

1. شغل المتصفح
2. شغل Wireshark
3. أدخل العنوان التالي ضمن URL: www.ietf.org
4. أضغط على الزر F5 لطلب "refresh"
5. أوقف wireshark وصفي مدخلات wireshark بحيث يكون البروتوكول هو HTTP

أجب عن الأسئلة التالية:

1. أبحث ضمن أول طلب Get مرسل من قبل متصفحك. هل ترى سطر "IF-modified-since" ضمن HTTP GET؟
2. افحص رسالة الجواب القادمة من المخدم. هل أرسل المخدم المحتوى المطلوب؟ كيف يمكنك التأكد من ذلك؟
3. أفحص الآن محتوى ثاني HTTP Get مرسلة إلى المخدم نفسه. هل ترى السطر "if-modified-since" ضمن HTTP Get؟ في حالة الإجابة بنعم، ما هي المعلومات الإضافية التي تتبع هذه الترويسة؟
4. ما هي حالة الجوب Status Code المرسلة كجواب للاستفسار الثاني؟ هل أرسل المخدم المحتوى المطلوب؟

### 3.3. عرض ملف كبير

سنحاول في هذا التمرين عرض ملف كبير ونسجل الملاحظات.

1. شغل المتصفح
2. شغل wireshark
3. أدخل العنوان التالي: www.hiast.edu.sy
4. أوقف wireshark

أجب على الأسئلة التالية:

1. ما عدد الاستفسارات من نوع Get request التي أرسلها المتصفح؟
2. ما ترتيب الاستفسار Get الذي يطلب الغرض

```
GET /sites/all/modules/languageicons/flags/ar.png
```

3. ما هو رقم الطرد الذي يحمل رسالة الجواب Status code على أول رسالة GET
4. ما هي قيمة الجواب والجملة Status code and phrase
5. ما هو عدد الطرود الكلية بما فيها طرود TCP اللازمة لإرسال الطلبات واستلام الأجوبة؟
6. ما هي أرقام البوابات المستخدمة TCP source and destination ports ضمن رسائل طلبات Get ورسائل الجواب عليها؟

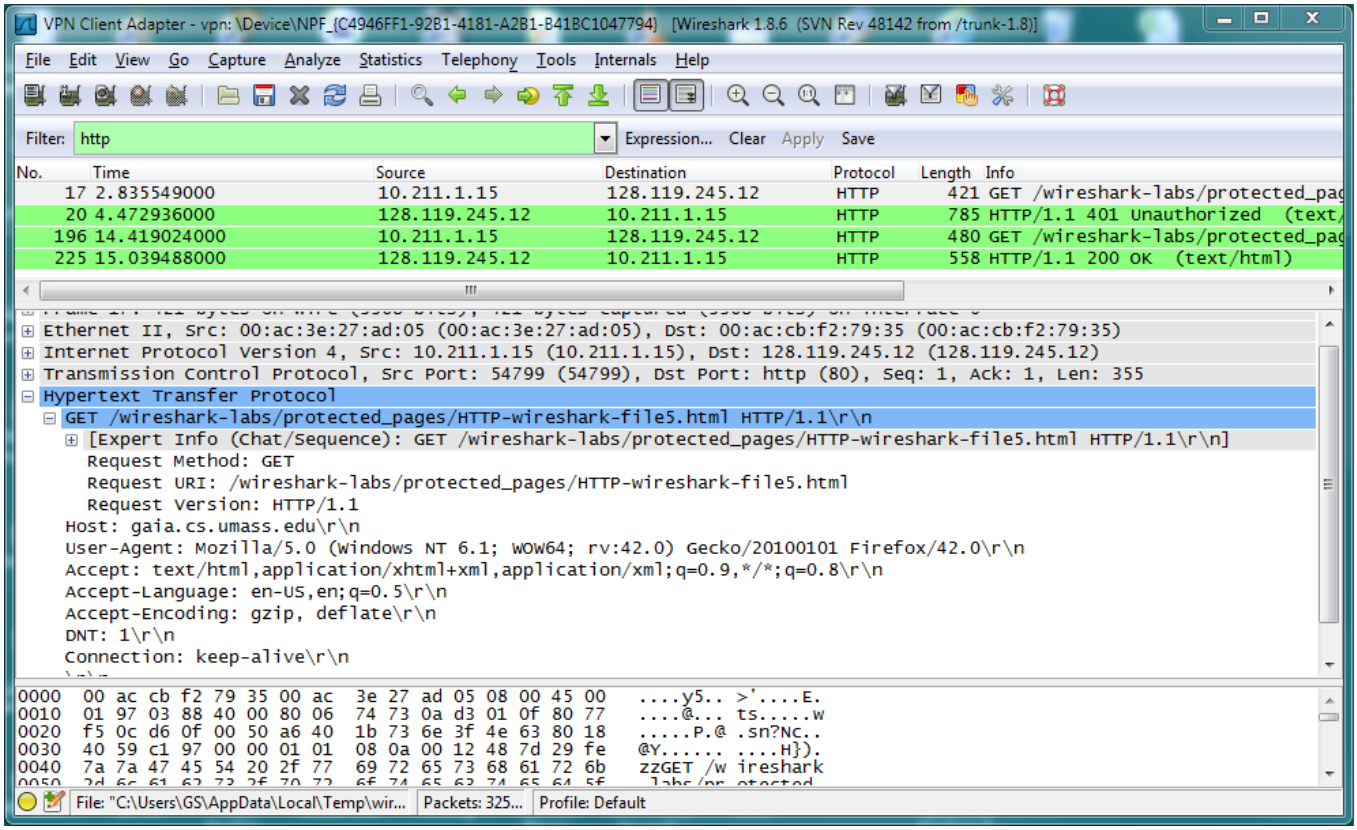
### 4.3. استيقان HTTP

لنحاول زيارة موقع وب يطلب اسم مستخدم وكلمة مرور ونفحص تسلسل رسائل HTTP المتبادلة.

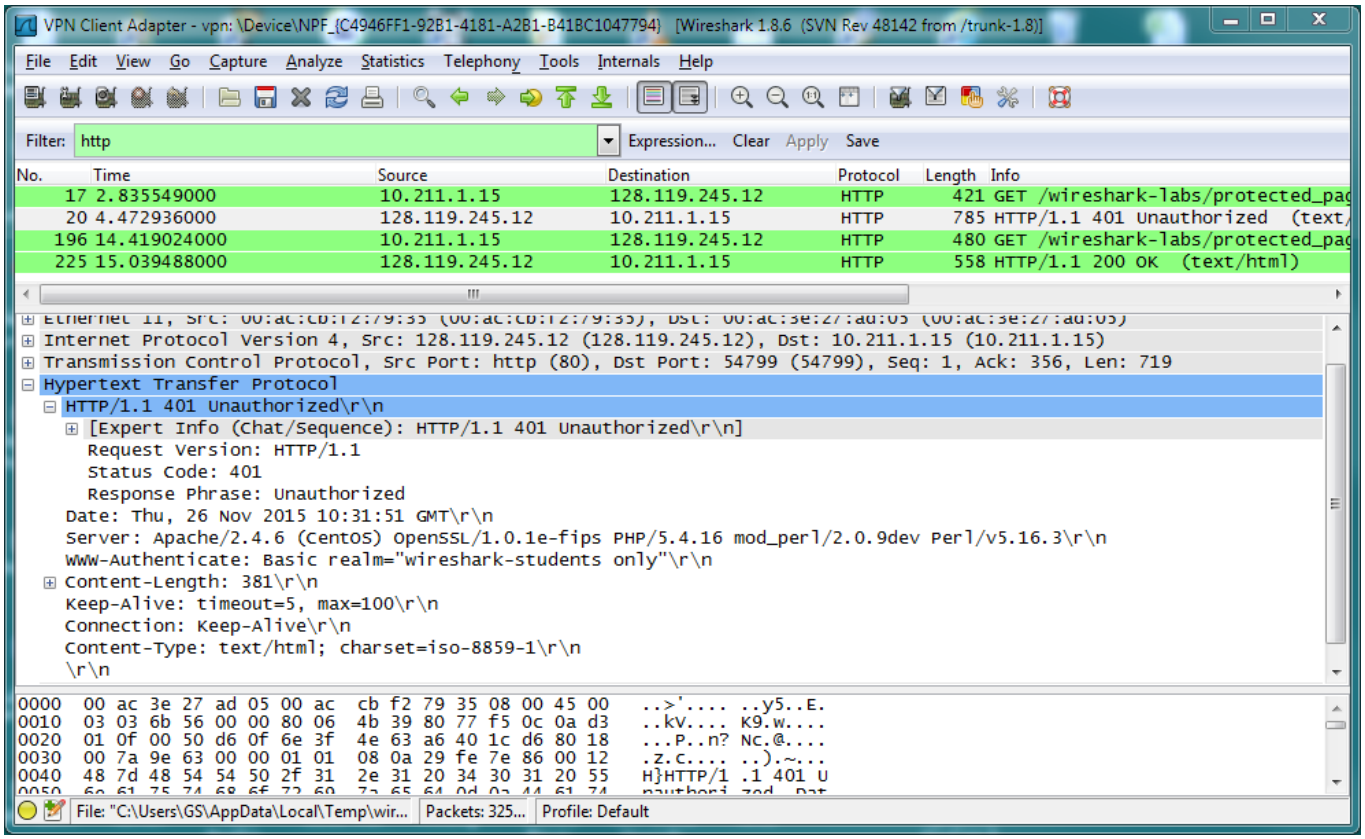
طبعاً سوف نشغل wireshark (بعد حذف history مع كل محاولة جديدة) ثم ندخل العنوان التالي ضمن المتصفح

```
http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html
```

أدخل اسم المستخدم "wireshark-students" وكلمة المرور "network" وافحص نتائج محلل الطرود wireshark. تبين الأشكال التالية تسلسل إرسال الرسائل:

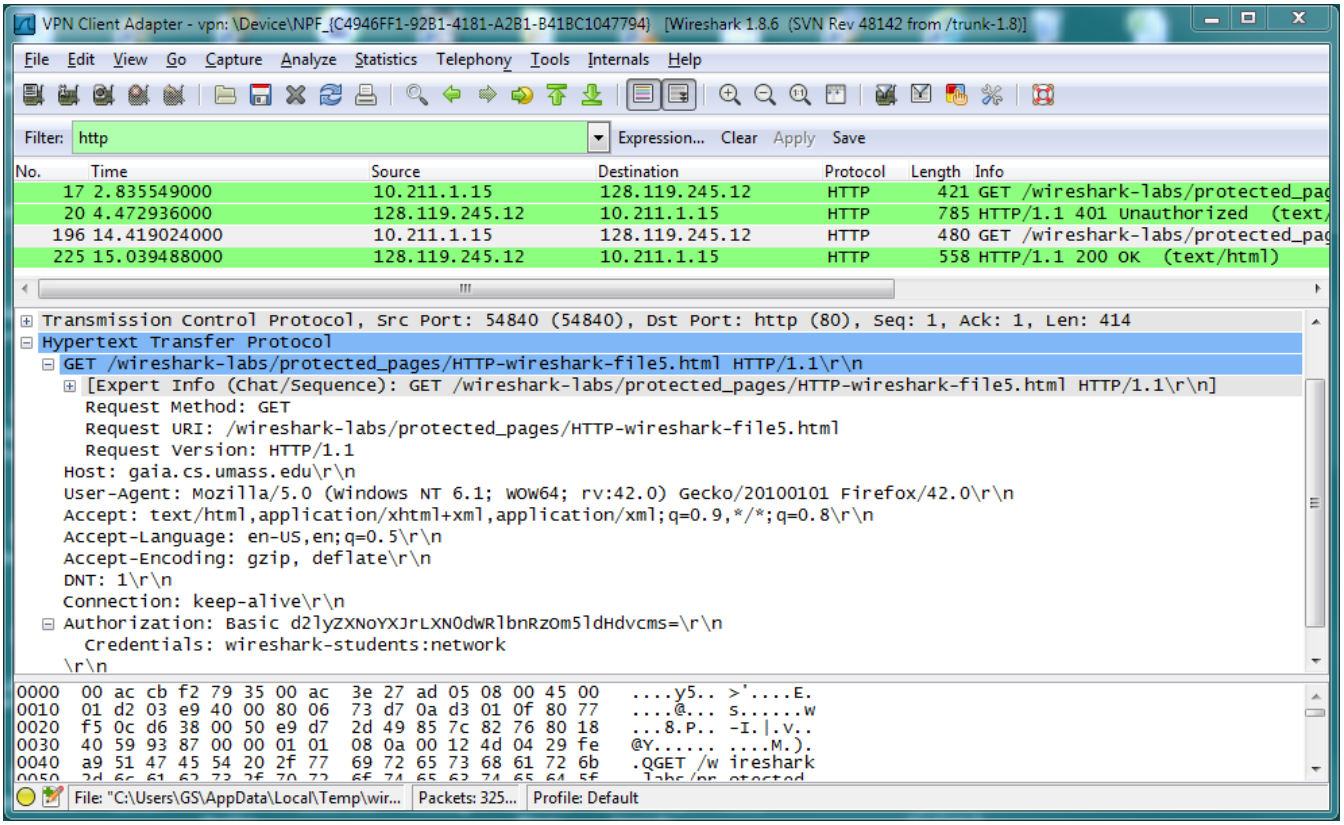



الشكل 15– الطرد الأول




الشكل 16– طرد الجواب

نلاحظ أن طرد الجواب يحمل الرقم 401 Unauthorized




الشكل 17– طرد HTTP يحمل معلومات الاستيقان

الطرد المبين في الشكل السابق يحمل اسم المستخدم وكلمة المرور المطلوبة للموقع. لاحظ أن معلومات الاستيقان d2lyZXN…. عندما تكون من النوع Basic تكون مرمزة وفق الترميز Base64 وليس مشفرة حيث يعرض السطر التالي نتيجة فك الترميز. بفضل هنا استخدام بروتوكول Digest الذي لا يرسل كلمة المرور عبين الطرفين ويكون أكثر أمناً.

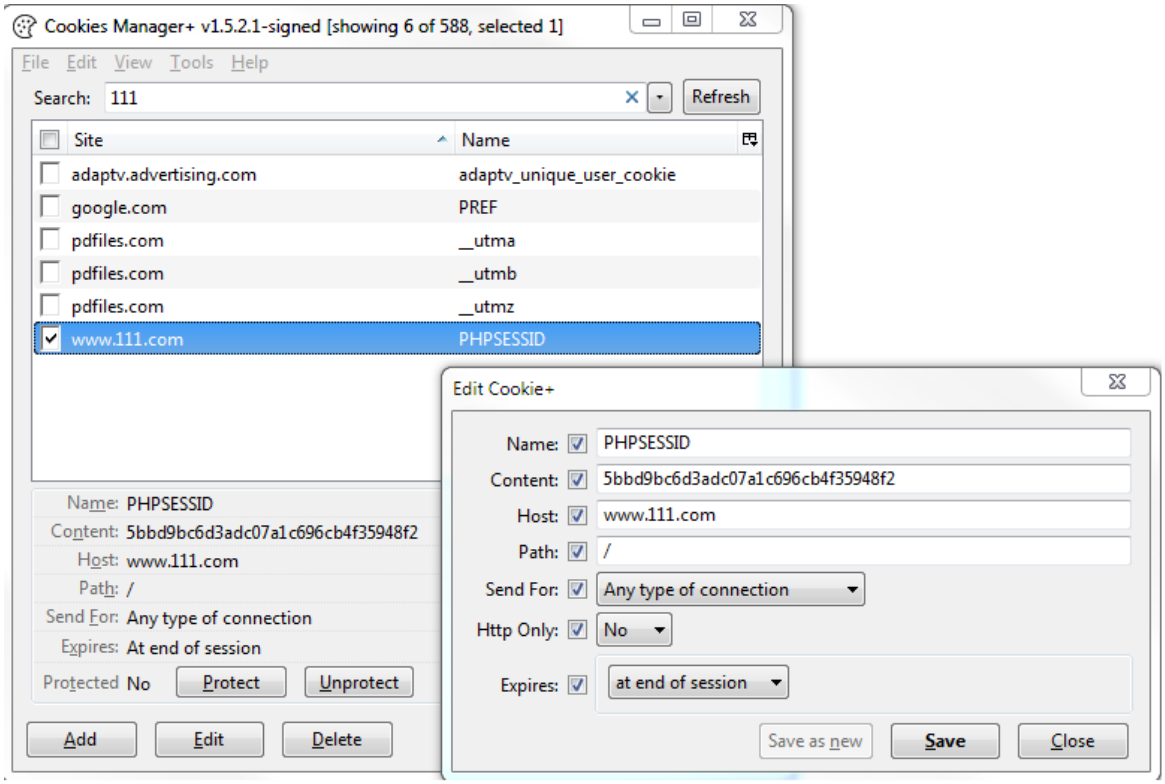
الطرد الأخير يظهر نجاح عملية الاستيقان والسماح بالدخول وعرض المعلومات المطلوبة.

يطلب هنا من الطالب إعادة الخطوات السابقة والتأكد من الطرود الأربعة التي يجري إرسالها ومعلومات الاستيقان.

### 5.3.إدارة الكوكيز

ينصح هنا بتحميل Firefox Cookies Manager+ add on.

ندخل إلى الموقع www.111.com ثم نستدعي Cookies Manager+ فتظهر لنا صفحة مثل الشكل التالي:




يبين الشكل السابق الكوكيز المضافة ومعلومات عنها.

على الطالب هنا أن يشغل wireshark ومن ثم الدخول إلى الموقع www.111.com/dns وفحص تسلسل الطرود وكتابة المعلومات الخاصة بالكوكيز عند أول دخول والجواب من المخدم وعند ثاني دخول إلى المخدم نفسه.

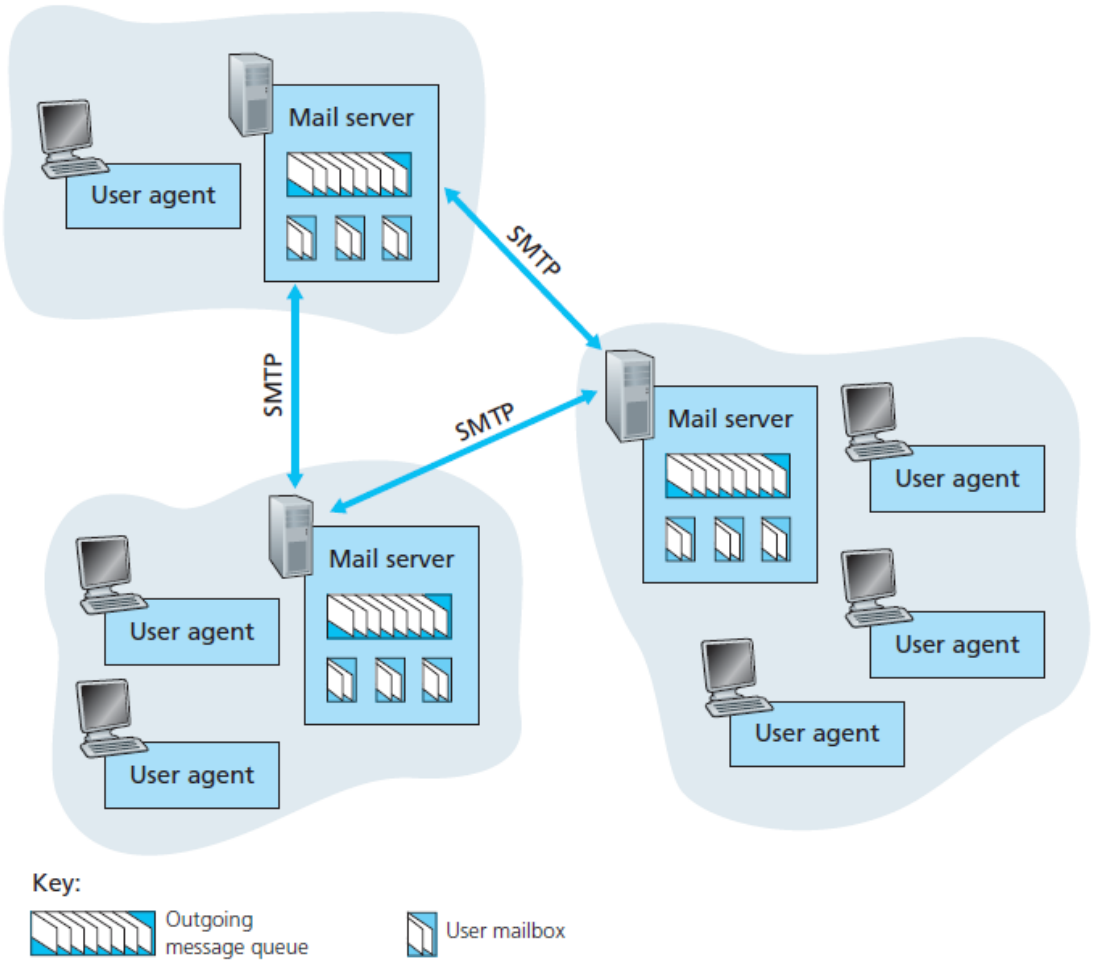
# البريد الإلكتروني وبروتوكول SMTP

## 1. البريد الالكتروني:

البريد الالكتروني موجود منذ بداية الإنترنت وكان من أكثر التطبيقات استخداماً وما يزال يحظى بأهمية كبيرة في أوقاتنا هذه. يعتبر البريد الالكتروني من وسائط الاتصال غير المتزامنة – نرسل ونقرأ البريد الالكتروني عندما نريد دون الحاجة للتنسيق مع الآخرين– وسريعة التوزيع وسهلة الاستخدام وغير مكلفة.

سنهتم في هذا الفصل ببروتوكول البريد الالكتروني SMTP المستخدم بين المخدمات وبروتوكول POP3 المستخدم لاسترجاع الرسائل من علب البريد إضافةً إلى التطرق إلى البريد الالكتروني المعتمد على الوب Web–based mail.

سنبدأ بالمكونات الأساسية لنظام البريد الالكتروني.




الشكل 1– لمحة عامة عن مكونات نظام البريد الالكتروني

نلاحظ من الشكل السابق ثلاثة مكونات أساسية للبريد الالكتروني: مخدم البريد الالكتروني Mail Server أو ما يعرف بعميل نقل الرسائل Message Transfer Agent (MTA) وبروتوكول نقل الرسائل المبسط Simple Mail Transfer Protocol (SMTP) وعميل المستثمر User Agent (UA).

عندما تريد Alice إرسال بريد الكتروني إلى Bob فإنها تستخدم أحد برامج البريد الالكتروني (أي UA) مثل Microsoft Outlook أو Apple Mail الذي يفيد في قراءة الرسائل والإجابة عليها أو إعادة توجيهها أو حفظها أو كتابة رسالة جديدة. عند الانتهاء من كتابة الرسالة، يقوم UA بإرسالها إلى مخدم البريد الالكتروني التي تملك حساباً لديه حيث يجري وضع الرسالة ضمن رتل خاص يدعى Outgoing message queue. بعد ذلك، يتصل مخدم بريد Alice مع مخدم بريد Bob ويرسل له الرسالة باستخدام بروتوكول SMTP. عندما يريد Bob قراءة الرسائل الموجهة له، فإنه يستخدم برنامج UA أيضاً لاسترجاع رسائله من علبة البريد المخصصة له ضمن مخدم البريد الالكتروني الذي يملك حساباً فيه. يقوم مخدم البريد بالاستيقان من Bob عن طريق اسم وكلمة مرور. إذا لم يستطع مخدم Alice من الاتصال بمخدم Bob فإنه يحتفظ بالرسائل ضمن message queue حتى يستطيع المحاولة في وقت لاحق. عادةً، إعادة المحاولة تجري كل 30 دقيقة ولعدة أيام مستمرة، فإذا لم ينجح بالإرسال فإنه يحذف الرسالة من الرتل ويعلم Aliceعن طريق رسالة الكترونية تفيده بعدم نجاح الإرسال.

يستخدم البروتوكول SMTP بروتوكول TCP الموثوق لنقل الرسائل من برنامج المرسل UA إلى المخدم المصدر وبين المخدم المصدر والمخدم الوجهة.

### 1.1. وكيل المستثمر User Agent (UA):

يعتبر UA المكون الأولي من مكونان نظام البريد الإلكتروني. يزود المستثمر بخدمة تسهيل عملية إرسال واستقبال الرسائل.

الخدمات التي يزودها UA

يقوم برنامج UA بتحرير composes وقراءة reads والرد على replies to وإعادة توجيه الرسائل forwards إضافةً إلى معالجة علب البريد.

- تحرير الرسائل Composing Messages

تجري عملية التحرير عن طريق تزويد قالبtemplate يقوم المستثمر بملئه. تزود بعض UAs محرر نصوص قادراً على اكتشاف الأخطاء الإملائية والنحوية.

- قراءة الرسائل Reading Messages

تسمح غالبية UAs للمستثمر بمشاهدة البريد الوارد مزود بملخص عن كل رسالة مستقبلة. تحوي الرسالة على الحقول التالية:

  - حقل خاص برقم الرسالة.
  - راية تدل على حالة الرسالة: جديدة، أو مقروءة سابقاً لكن بدون رد، أو مقروءة مع رد.
  - حجم الرسالة.

- المرسل.
- عنوان الرسالة الاختياري.

- قراءة الرسائل Reading Messages

إضافةً إلى السماح للمستثمر بقراءة رسائله فإن معظم UAs تتيح إمكانية عرض ملخص عن الرسائل المستقبلة. تحوي الرسائل الإلكترونية الحقول التالية:

- رقم الرسالة.
- راية تدل على كون الرسالة جديد أو تمت قراءتها بدون الرد عليها أو تمت قراءتها وتم الرد عليها.
- حجم الرسالة.
- المرسل.
- حقل العنوان الاختياري.

- الرد على الرسائل Replying to Messages

يستطيع المستقبل، بعد قراءة الرسالة، أن يرد عليها باستخدام UA. يسمح UA عادةً للمستقبل أن يوجه الرد إلى المرسل الأصلي للرسالة أو إلى جميع المستقبلين.

إعادة توجيه الرسالة Forwarding Messages

يسمح UA للمستثمر بتوجيه الرسالة إلى طرف ثالث مختلف عن المرسل الأصلي أو المستقبلين.

معالجة علب البريد Handling Mailboxes

يخلق UA عادةً علبتي بريد: علبة الوارد inbox وعلبة الصادر outbox. علبة البريد هي ملف ذو صيغة خاصة تسمح بمعالجته من قبل UA. كما تسمح أيضاً غالبية UAs بخلق علب بريد إضافية حسب الحاجة.

- إرسال الرسائل Sending Mail

يستخدم المستثمر برنامج UA لتحرير ولإرسال الرسالة. تتألف الرسالة من مغلف Envelope ورسالة Message.

يحوي المغلف عادةً على عنوان المرسل وعنوان المستقبل وبعض المعلومات الإضافية. أما الرسالة فتتألف من ترويسة header ومتن body. تُعرِّف ترويسة الرسالة المرسل والمستقبل وعنوان الرسالة وبعض المعلومات الإضافية. يحوي متن الرسالة على المعلومات المطلوب قراءتها من قبل المستقبل.

- استقبال الرسائل Receiving Mails

يجري تشغيل UA عن طريق المستثمر أو عن طريق مؤقت زمني. يقوم UA بإعلام المستثمر في حال وجود رسالة له. عندما يكون المستثمر جاهزاً لقراءة الرسائل، يقوم UA بإظهار لائحة يحوي كل سطر منها ملخص عن رسالة واحدة. يحوي الملخص على عنوان المرسل وموضوع الرسالة وزمن إرسال أو استقبال الرسالة. يستطيع المستقبل اختيار رسالة ما وإظهار محتوياتها.

- العناوين Addresses

يتألف العنوان من جزأين: جزء محلي واسم نطاق، مفصولين عن بعضهما باستخدام الرمز @.

يعرف الجزء المحلي اسم ملف خاص، يدعى علبة البريد، حيث يجري تخزين جميع الرسائل الواردة إلى مستثمر معين بغية استحضارها من قبل UA.

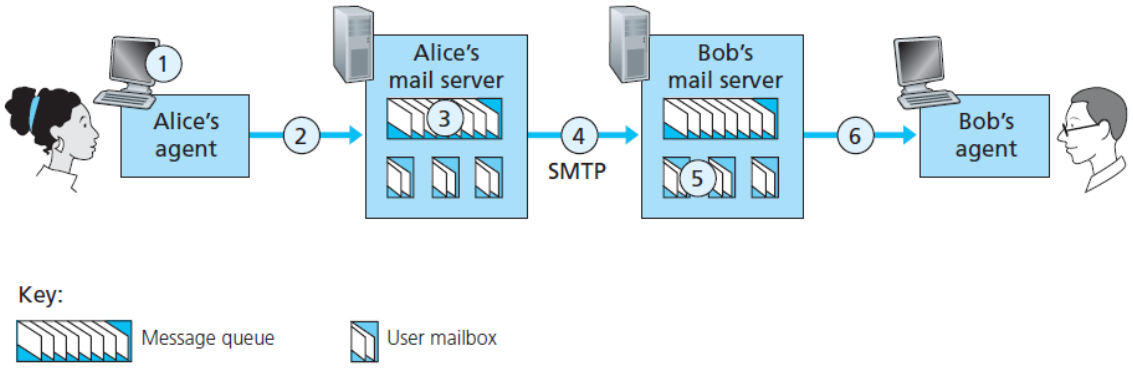
تختار كل مؤسسة عادةً مضيف أو أكثر لمهمة استقبال وإرسال الرسائل؛ يطلق عليهم اسم مخدمات البريد الإلكتروني.

- اللائحة البريدية Mailing List

يسمح البريد الإلكتروني باستخدام اسم واحد، alias، لتمثيل عدة عناوين بريدية؛ هذا ما يدعى باللائحة البريدية. في كل مرة يتم إرسال رسالة ما، يقارن النظام اسم المستقبل مع قاعدة المعطيات الخاصة بـ aliases؛ فإذا وجد لائحة بريدية موافقة للاسم يجري تحضير رسالة مستقلة إلى كل عنوان من عناوين اللائحة وتسليمها لـ MTA. أما في حال عدم وجود لائحة بريدية موافقة فيجري إرسال الرسالة إلى العنوان نفسه عن طريق Mail Transfer Entity.

## 2.1. وكيل نقل الرسائل SMTP: MTA

تجري عملية نقل الرسائل فعلياً عن طريق MTAs. يجري الإرسال عن طريق زبون MTA والاستقبال عن طريق مخدم MTA. يدعى البروتوكول الذي يعرف زبون ومخدم MTA ضمن الإنترنت ببروتوكول نقل الرسائل المبسط Simple Mail Transfer Protocol (SMTP). يبين الشكل التالي المجال الذي يغطيه بروتوكول SMTP.




الشكل 2– Alice ترسل رسالة إلى Bob

لنأخذ المثال المبين في الشكل السابق. Alice تريد إرسال رسالة بسيطة مكونة من Ascii Code إلى Bob.

1. تشغل Alice برنامج UA وتدخل عنوان Bob ومن ثم تدخل نص الرسالة وتضغط على أمر الرسال Send.
2. يرسل عميل المستثمر Alice الرسالة إلى مخدم البريد الخاص بها وتوضع الرسالة ضمن رتل Outgoing messages
3. يعمل مخدم بريد Alice كزبون لبروتوكول SMTP حيث يرى الرسالة الموجهة إلى Bob ضمن رتل الرسائل الصادرة فيفتح اتصال TCP مع مخدم Bob للبريد الالكتروني
4. بعد مجموعة من عمليات المصافحة بين الطرفين Handshaking، يرسل مخدم Alice الرسالة إلى مخدم Bob عبر اتصال TCP
5. يستقبل مخدم Bob الرسالة ويضعها ضمن علبة بريد Bob
6. يشغل Bob عميله الخاص UA ويستقبل الرسالة في الوقت الذي يريده.

نلاحظ من المثال السابق أن الاتصال بين مخدمي البريد لا يمر عبر مخدمات وسيطة

نلاحظ أيضاً أنه يجري استخدام SMTP مرتين، بين المرسل وبين مخدم بريد المرسل وبين مخدمي البريد. كما سنلاحظ لاحقاً، فإننا نحتاج إلى بروتوكول آخر بين مخدم البريد وبين المستقبل.

سندرس الآن بقليل من التفصيل الأوامر المتبادلة بين مخدمي البريد الالكتروني

يعرف بروتوكول SMTP طريقة تبادل الأوامر والإجابات بين MTA client وبين MTA server.

### الأوامر والإجابات Commands and Responses:

تهدف الأوامر والإجابات إلى نقل الرسائل بين الزبون والمخدم. يتم إنهاء كل أمر أو كل إجابة بعلام نهاية السطر مكون من محرفين (carriage return and line feed).

### الأوامر Commands:

تُرسل الأوامر من الزبون إلى المخدم. يتألف كل أمر من قسمين: كلمة مفتاحية keyword يتبعها مجموعة من arguments وهو على الشكل `keyword: argument(s)`.

يعرف SMTP 14 أمراً: الخمسة الأوائل إجبارية على جميع التحقيقات أن تدعمها. يفضل أن تكون الأوامر الثلاثة التالية موجودة Highly recommended. أما الأوامر الباقية، فهي غير إجبارية. يبين الشكل التالي هذه الأوامر.

| Keyword | Argument(s) |
|---|---|
| HELO | Sender's host name |
| MAIL FROM | Sender of the message |
| RCPT TO | Intended recipient of the message |
| DATA | Body of the mail |
| QUIT | |
| RSET | |
| VRFY | Name of recipient to be verified |
| NOOP | |
| TURN | |
| EXPN | Mailing list to be expanded |
| HELP | Command name |
| SEND FROM | Intended recipient of the message |
| SMOL FROM | Intended recipient of the message |
| SMAL FROM | Intended recipient of the message |

الشكل 3– أوامر SMTP

- HELO. يستخدم الزبون هذا الأمر بغية التعريف عن نفسه. يكون المحدد المستخدم هو اسم نطاق مضيف (حاسب) الزبون. يأخذ الأمر الصيغة

```
HELO: mail.svuonline.org
```

- MAILFROM. يستخدم الزبون هذا الأمر للتعريف عن مرسل الرسالة. يكون المحدد المستخدم هو العنوان البريدي للمرسل. يأخذ الأمر الصيغة

```
MAIL FROM: alice@svuonline.org
```

- RCPT TO. يستخدم الزبون هذا الأمر للتعريف عن المستقبل المقصود بالرسالة. يكون المحدد المستخدم هو العنوان البريدي للمستقبل. في حال وجود مجموعة مستقبلين فإنه يجري تكرار الأمر. يأخذ الأمر الصيغة

```
RCTP TO: bob@gmail.com
```

- DATA. يستخدم هذا الأمر لإرسال متن الرسالة. تجري معالجة جميع الأسطر التي تلي هذا الأمر على أنها متن Body الرسالة. يتم إنهاء الرسالة باستخدام سطر يحوي نقطة فقط. يأخذ الأمر الصيغة

```
DATA
This is the message
To be sent to Alice
.
```

- QUIT. ينهي هذا الأمر جلسة الاتصال. الصيغة المستخدمة

```
QUIT
```

- RSET. يقطع هذا الأمر جلسة البريد الحالية ويجري حذف المعلومات المتعلقة بالمرسل وبالمستقبل. يأخذ الأمر الصيغة

```
RSET
```

- VRFY. يستخدم هذا الأمر للتحقق من كون عنوان المستقبل هو عنواناً صالحاً. يأخذ الأمر الصيغة

```
VRFY: bob@gmail.com
```

- NOOP. يستخدم الزبون هذا الأمر لاختبار حالة المستقبل وهو يتطلب جواباً من المستقبل. يأخذ الأمر الصيغة

```
NOOP
```

- TURN. يسمح هذا الأمر بتبديل الأدوار بين المرسل والمستقبل حيث يصبح المرسل مستقبلاً والعكس بالعكس. يأخذ الأمر الصيغة

```
TURN
```

- EXPN. يطلب هذا الأمر من المستقبل أن يوسع اللائحة البريدية المرسلة كمحددات arguments وإعادة العناوين البريدية للمستقبلين الذين ينتمون إلى اللائحة. يأخذ هذا الأمر الصيغة

```
EXPN: x  y  z
```

- HELP. يطلب هذا الأمر من المستقبل إرسال معلومات ضمن المحددات عن الأمر المرسل. يأخذ الأمر الصيغة

```
HELP: mail
```

- SEND FROM. يستخدم هذا الأمر لتوصيل الرسالة إلى طرفية المستقبل وليس علبة البريد. إذا لم يكن المستقبل على الخط فتعود الرسالة إلى المرسل. يحوي محدد هذا الأمر عنوان المرسل. يأخذ الأمر الصيغة

```
SEND FROM: bob@hotmail.com
```

- SMOL FROM. يستخدم هذا الأمر لتوصيل الرسالة إلى طرفية أو علبة بريد المستقبل. هذا يعني أنه إذا كان المستقبل على الخط فيجري توصيل الرسالة إلى الطرفية وإذا لم يكن على الخط فيجري توصيل الرسالة إلى علبة البريد. يكون المحدد هو عنوان المرسل. يأخذ الأمر الصيغة

```
SMOL FROM: bob@hotmail.com
```

- SMAL FROM. يستخدم هذا الأمر لتوصيل الرسالة إلى طرفية وعلبة بريد المستقبل

```
SMAL FROM: bob@hotmail.com
```

- الإجابات Responses

ترسل الإجابات من المخدم إلى الزبون. تتكون الإجابة من رمز ذو 3 خانات مع إمكانية إضافة معلومات نصية. تعني الخانة الأولى ما يلي:

- 2yz (positive completion reply). هذا يعني أن الأمر المطلوب قد تم تنفيذه بنجاح ويمكن إرسال أمراً جديداً.
- 3yz (positive intermediate reply). هذا يعني أنه قد تم قبول الأمر لكن المستقبل يحتاج إلى معلومات إضافية قبل البدء بالتنفيذ.
- 4yz (transient negative completion reply). هذا يعني أنه قد تم رفض الأمر بسبب خطأ مؤقت ويمكن إعادة إرسال الأمر مجدداً.

- 5yz (permanent negative completion reply). هذا يعني أنه قد تم رفض الأمر ولا توجد فائدة من إعادة الأمر مجدداً.

تزود الخانتان الثانية والثالثة تفاصيل إضافية عن الإجابة. يبين الشكل التالي مجموعة من الإجابات.

| Code | Description |
|---|---|
| **Positive Completion Reply** | |
| 211 | System status or help reply |
| 214 | Help message |
| 220 | Service ready |
| 221 | Service closing transmission channel |
| 250 | Request command completed |
| 251 | User not local; the message will be forwarded |
| **Positive Intermediate Reply** | |
| 354 | Start mail input |
| **Transient Negative Completion Reply** | |
| 421 | Service not available |
| 450 | Mailbox not available |
| 451 | Command aborted |
| 452 | Command aborted; insufficient storage |
| **Permanent Negative Completion Reply** | |
| 500 | Syntax error; unrecognized command |
| 501 | Syntax error in parameters or arguments |
| 502 | Command not implemented |
| 503 | Bad sequence of commands |
| 504 | Command temporarily not implemented |
| 550 | Command is not executed; mailbox unavailable |
| 551 | User not local |
| 552 | Requested action aborted; exceeded storage location |
| 553 | Requested action not taken; mailbox name not allowed |
| 554 | Transaction failed |

الشكل 4– الإجابات

### 3.1. طور نقل الرسائل:

تمر عملية نقل الرسالة بثلاثة أطوار: إنشاء الارتباط ثم نقل الرسالة ثم إنهاء الارتباط.

إنشاء الارتباط:

بعد أن ينشئ الزبون ارتباطاً إلى البوابة المعروفة رقم 25، يبدأ مخدم SMTP بطور إنشاء الارتباط. يتألف هذا الطور من ثلاث مراحل مبينة في الشكل التالي.

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
```

يرسل المخدم الرمز 220 (service ready) لإعلام الزبون بكونه جاهزاً لاستقبال الرسائل. إذا لم يكن المخدم جاهزاً فإنه يرسل الرمز 421 (service not ready).

يرسل الزبون رسالة HELO للتعريف عن نفسه باستخدام اسم النطاق الذي ينتمي إليه. تجدر الإشارة هنا أن المخدم والزبون كانا قد تعرفا على عناوينهم المنطقية IP addresses خلال فتح ارتباط TCP.

يرد المخدم بالرمز 250 (request command completed) أو أي رمز آخر حسب الحالة.

نقل الرسالة:

بعد تحقيق الارتباط بين مخدم SMTP وزبون SMTP، يمكن تبادل رسالة واحدة بين المرسل والمستقبل أو بين المرسل وعدة مستقبلين.يتطلب هذا الطور ثمان خطوات. يجري تكرار الخطوتين 3 و4 إذا كان يوجد أكثر من مستقبل للرسالة (انظر الشكل التالي).

1. يرسل الزبون رسالة MAIL FROM بغية التقديم عن نفسه. تفيد هذه الخطوة المخدم في حال إعادة رسائل الخطأ إلى المرسل.
2. يرد المخدم بالرمز 250 أو أي رمز آخر حسب الحالة.
3. يرسل الزبون رسالة RCPT TO التي تحوي العنوان البريدي للمستقبل.
4. يرد المخدم بالرمز 250 أو أي رمز آخر حسب الحالة.
5. يرسل الزبون رسالة المعطيات لبدء نقل الرسالة.
6. يرد المخدم بالرمز 354 (start mail input) أو أي رمز آخر حسب الحالة.
7. يرسل الزبون محتوى الرسالة سطراً سطراً. ينتهي كل سطر بالثنائية CR+LF. تنتهي الرسالة بسطر يحوي نقطة.
8. يرد المخدم بالرمز 250 (OK) أو أي رمز آخر حسب الحالة.

```
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr ... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
```

```
C: How about pickles?
C: .
S: 250 Message accepted for delivery
```

إنهاء الارتباط:

ينهي الزبون الارتباط بعد إكمال نقل الرسالة بنجاح. يتطلب هذا الطور خطوتين.

1. يرسل الزبون أمر QUIT.
2. يرد المخدم بالرمز 221 أو أي رمز آخر حسب الحالة.

طبعاً، يجري إغلاق ارتباط TCP بعد إنهاء ارتباط SMTP.

```
C: QUIT
S: 221 hamburger.edu closing connection
```

### 4.1. صيغ رسائل البريد الالكتروني:

يعرف المعيار [RFC 5322] الترويسات الإضافية التي يمكن إرفاقها مع نص الرسالة مثل التاريخ والموضوع والمرسل والمستلم وغيرها. يجري فصل الترويسات الإضافية عن نص الرسالة باستخدام سطر فارغ (أي CRLF). يوصف المعيار السابق صيغ هذه الترويسات وتفسير كل صيغة منها. مثل الترويسات المستخدمة في بروتوكول HTTP يجري هنا استخدام كلمة مفتاحية Keyword متبوعة بالنقطتين ":" ومن ثم يوضع قيمة. بعض الكلمات المفتاحية مطلوبة بينما الأخرى اختيارية.

القيم المطلوبة هي:

```
From:
To:
```

القيم الاختيارية:

```
Subject:
```

من الضروري الانتباه هنا إلى أن هذه القيم مختلفة عن القيم التي رأيناها ضمن أوامر SMTP مثل Mail from و Rcpt to.

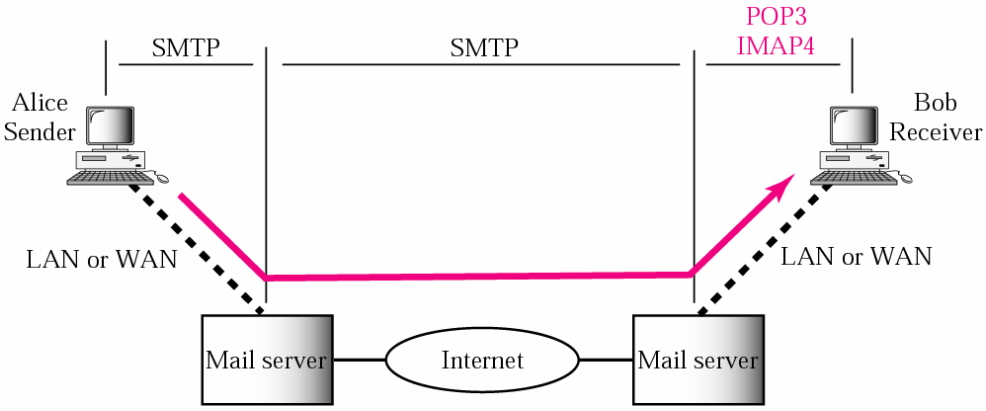
مثال عن هذه الترويسات:

```
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Searching for the meaning of life.
```

### 5.1. وكلاء النفاذ إلى الرسائل: POP و IMAP

يجري استخدام بروتوكول SMTP في المرحلتين الأولى والثانية من مراحل توصيل الرسائل. لكن بما أن هذا البروتوكول هو من النوع الدفعي Push فيلزمنا بروتوكولاً آخر، لتحقيق المرحلة الثالثة، يفيد في استحضار الرسالة ويكون من نوع pull. نستخدم هنا وكيل النفاذ إلى الرسائل.

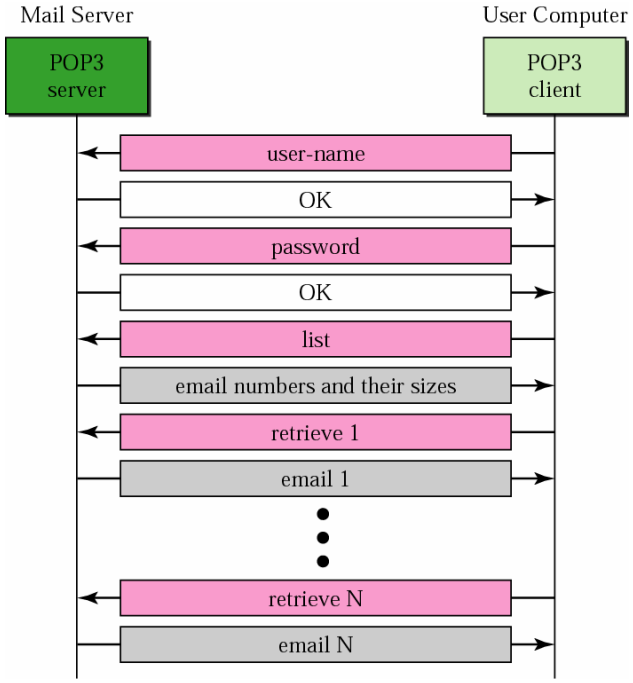
يتوفر حالياً بروتوكولان للنفاذ إلى الرسائل: Post Office protocol Version 3 (POP3) و Internet Mail Access Protocol version 4 (IMAP4). يبين الشكل التالي مكان هذين البروتوكولين.




الشكل 5– POP4 و IMAP4

### 1.5.1. بروتوكول POP3 :

يجري تنصيب زبون POP3 لدى حاسب المستقبل بينما يجري تنصيب مخدم POP3 لدى مخدم البريد.
تبدأ عملية النفاذ إلى البريد عندما يريد المستثمر تحميل بريده من علبة البريد الموجودة على مخدم البريد. يفتح الزبون ارتباط TCP على البوابة رقم 110. يرسل بعد ذلك اسم المستثمر وكلمة المرور. يستطيع المستثمر الآن سرد واستحضار رسائل البريد واحدة بواحدة. يبين الشكل التالي مثالاً عن تحميل الرسائل باستخدام POP3.




الشكل 6– POP3

يعمل POP3 وفق نمطين: نمط الحذف ونمط المحافظة. يجري، وفق نمط الحذف، حذف الرسالة من علبة البريد بعد استحضارها. بينما يجري، وفق نمط المحافظة، المحافظة على الرسالة ضمن علبة البريد بعد استحضارها. يستخدم النمط الأول عندما ينفذ المستثمر إلى بريده عن طريق حاسبه الأساسي حيث يستطيع حفظ وتنظيم الرسائل الواردة بعد قراءتها بينما يستخدم النمط الثاني المستثمر الذي ينفذ إلى بريده عن بعد وبشكل مؤقت أو عن طريق الحاسب المحمول.

### 2.5.1. بروتوكول IMAP4 :

يعتبر بروتوكول IMAP4 أكثر فعالية من بروتوكول POP3 لكنه أكثر تعقيداً في الوقت نفسه.
من مساوئ POP3 أنه لا يسمح للمستثمر بحفظ وتنظيم رسائله على المخدم كما لا يستطيع المستثمر امتلاك عدة مجلدات على المخدم. إضافةً إلى ما سبق، لا يسمح POP3 للمستثمر بمعاينة محتويات الرسائل جزئياً قبل تحميلها.
يزود IMAP4 الوظائف الإضافية التالية:

- يستطيع المستثمر معاينة ترويسة الرسالة قبل التحميل.
- يستطيع المستثمر البحث ضمن محتوى الرسالة على سلسلة محارف قبل التحميل.
- يستطيع المستثمر تحميل الرسالة جزئياً.
- يستطيع المستثمر خلق وحذف وإعادة تسمية علب البريد على المخدم.
- يستطيع المستثمر خلق تراتب علب بريد Hierarchy of mailboxes ضمن مجلد لتخزين الرسائل.

### 3.5.1. البريد الإلكتروني المعتمد على الوب Web-based MAIL:

تزود بعض مواقع الوب خدمة البريد الإلكتروني لأي شخص قادر على النفاذ إلى الموقع مثل موقعي hotmail و Gmail و Yahoo. يجري هنا نقل الرسالة من متصفح Alice إلى مخدم بريدها عبر بروتوكول HTTP. يجري نقل الرسائل من مخدم البريد المرسل إلى مخدم البريد المستقبل باستخدام SMTP. أخيراً، يجري استقبال الرسالة من مخدم البريد المستقبل باستخدام HTTP.
عندما يريد Bob استحضار رسائله فإنه يقوم بإرسال رسالة إلى موقع الوب. يقوم الموقع بإرسال استمارة مطلوب ملئها من قبل Bob تحوي على اسم المستثمر وكلمة المرور. في حال تطابق المعلومات المدخلة من قبل Bob يقوم الموقع بنقل الرسالة إلى متصفح Bob بصيغة HTML.

## 2. القسم العملي:

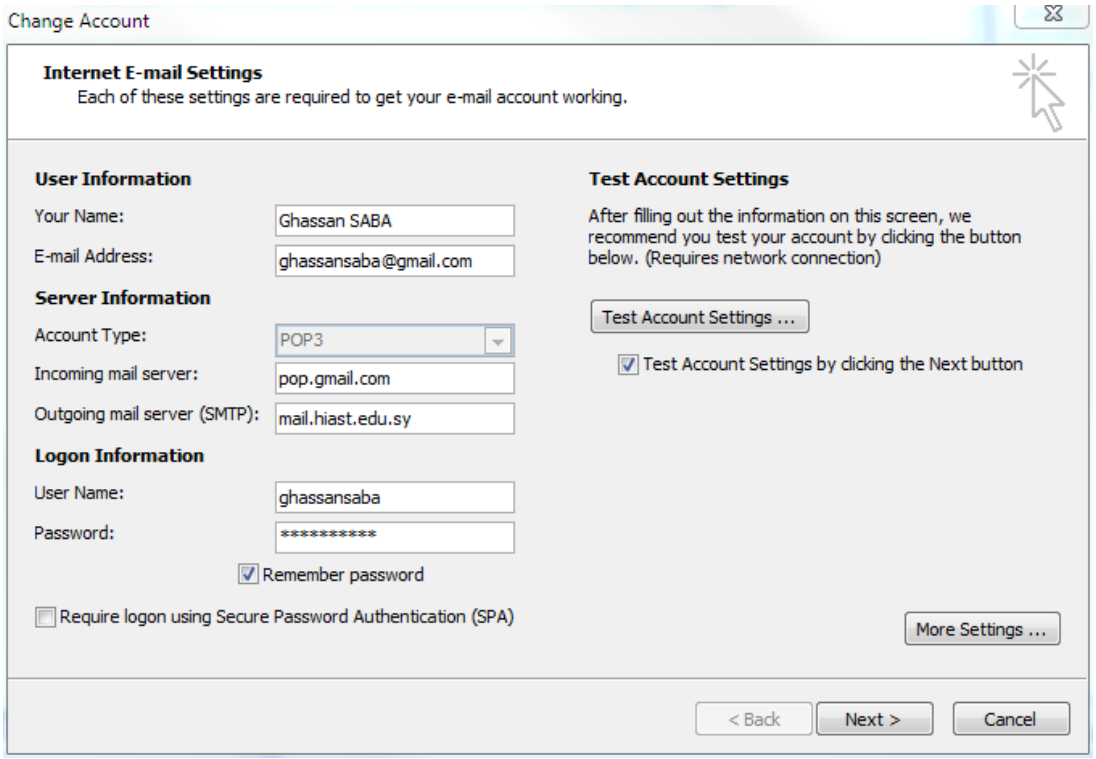
### 1.2. إعداد Outlook:

سنقوم ضمن هذا الجزء بإعداد Microsoft Outlook 2010 للتعامل مع مخدم البريد الالكتروني الصادر والوارد للمخدم gmail.com.

1. نفتح برنامج Outlook
2. من File نختار Add Account
3. ندخل الاسم وعنوان البريد الالكتروني وليكن ghassansaba@gmail.com وكلمة المرور مرتين
4. أدخل المعلومات التالية:

- Account Type: POP3
- Incoming mail server: pop.gmail.com
- Outgoing mail server (SMTP): mail.hiast.edu.sy **يتعلق بمزود الخدمة التي تتصل منه**

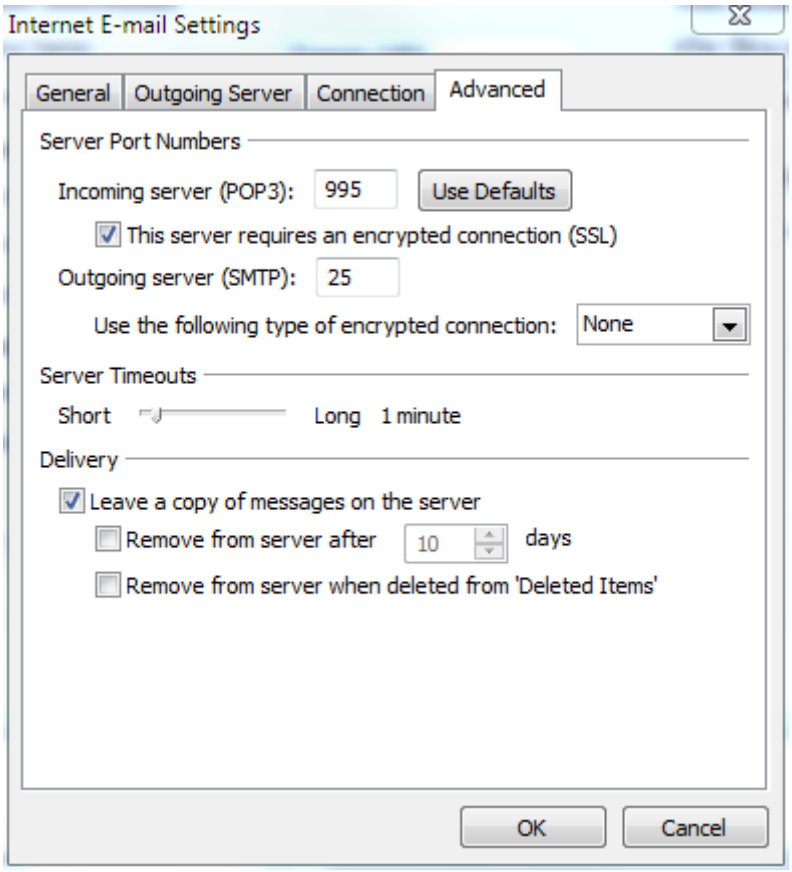
كما هو موضح في الشكل التالي:




الشكل 7– إعداد حساب gmail

ملاحظة: يجب تعريف Outgoing mail server (SMTP) حسب مخدم البريد الذي تستخدمه للإرسال (أنا مثلاً استخدم mail.hiast.edu.sy) فإذا كنت تستخدم مخدم الافتراضية فعليك استخدام mail.svuonline.org أو إذا كنت تتصل من المنزل فيجب عليك معرفة مخدم البريد الصادر حسب مزود خدمة الإنترنت التي تتعامل معها).

1. أضغط على more settings ومن ثم اختر Advanced وأدخل القيم المبينة في الشكل التالي:




الشكل 8– إعدادات متقدمة لمخدم gmail

2. يمكنك الآن اختبار الحساب بالضغط على Test Account Settings

نلاحظ هنا أن لكل مخدم بريد الكتروني إعدادات بريد صادر ووارد مختلفة عن غيره ويجب معرفتها عن طريق الإنترنت أو مدير مزود خدمة الإنترنت أو عن طريق

```
nslookup -type=MX svuonline.org
```

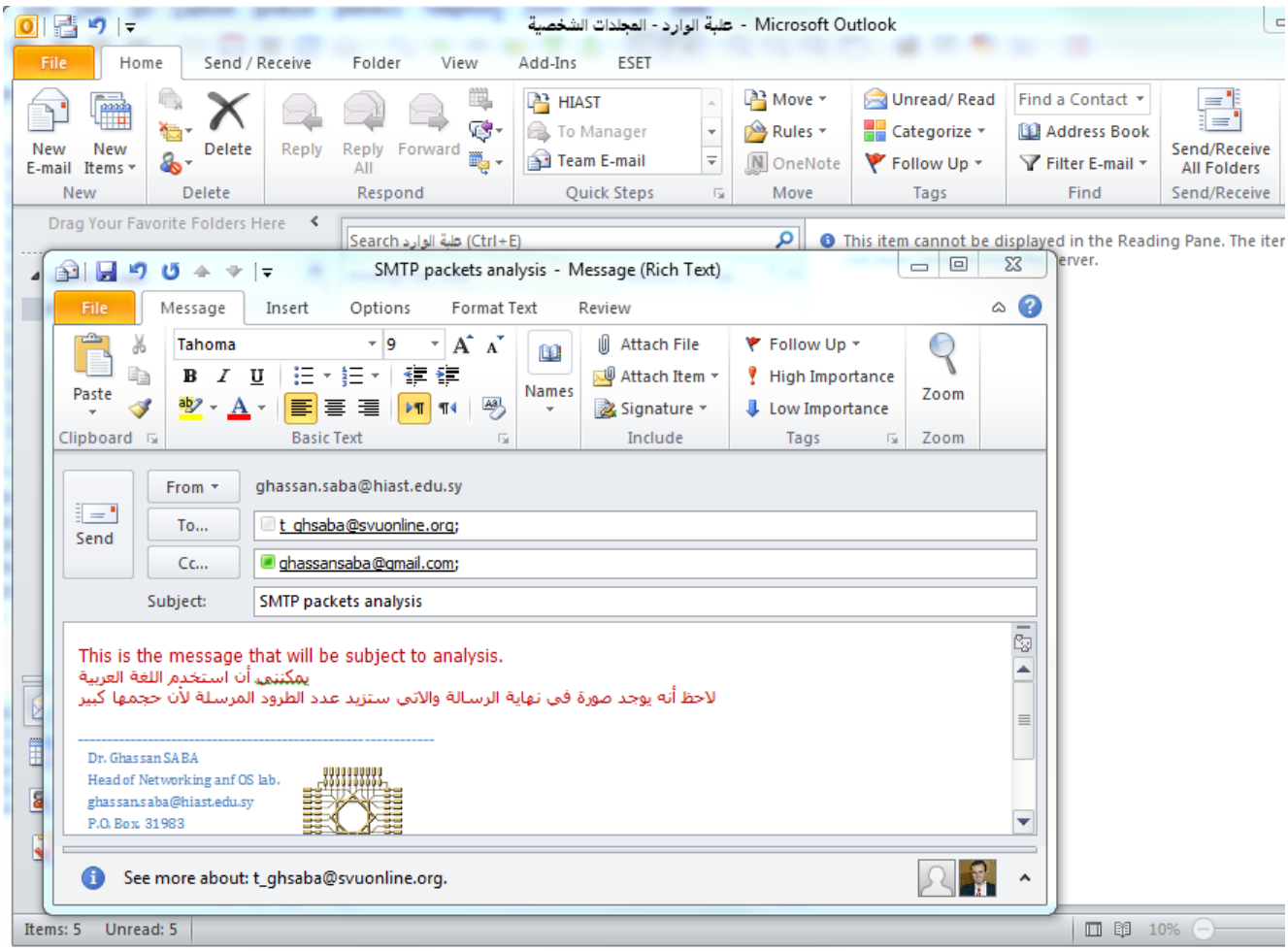
المطلوب: يجب على الطالب إضافة حساب البريد الالكتروني في الجامعة الافتراضية حيث:

```
Incoming mail server: mail.svuonline.org
More settings -> Advanced -> incoming server (POP3): 995
Check "This server requires an encrypted connection (SSL)"
Outgoing server (SMTP): 25
```

### 2.2. تحليل طرود SMTP عند الإرسال:

سنحاول ضمن هذا التمرين تحليل الطرود التي يرسلها Outlook عند إرسال رسالة ما باستخدام wireshark.

بعد تشغيل outlook أضغط على New E-mail فيظهر لك صندوق حوار لإدخال رسالة جديدة كما هو موضح في الشكل التالي:

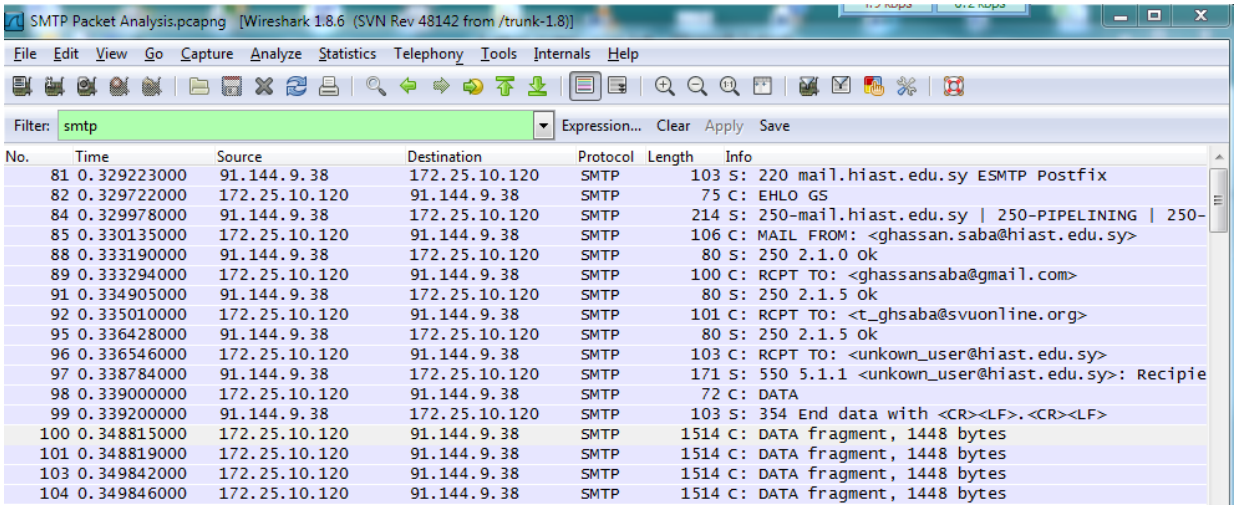



الشكل9– إضافة رسالة جديدة

لاحظ المعلومات التالية:

```
Sender address: ghassan.saba@hiast.edu.sy
Receiver address1: t_ghsaba@svuonline.org
Receiver address 2:unknown_user@hiast.edu.sy أي مستثمر غير معرف
Receiver address 3 (cc): ghassansaba@gmail.com
```

يبين الشكل التالي صورة مأخوذة من برنامج wireshark للتفاعل بين المخدم mail.hiast.edu.sy والزبون وهو برنامج outlook للحاسب GS.




الشكل 10– ملف أثر SMTP packet

ملاحظة: بما أن بريد المعهد العالي HIAST يستخدم Extended SMTP (ESMTP) فعبارة التأهيل أصبحت EHLO بدلاً عن HELO المستخدمة في نسخة SMTP الأصلية.

العمل المطلوب من الطالب:

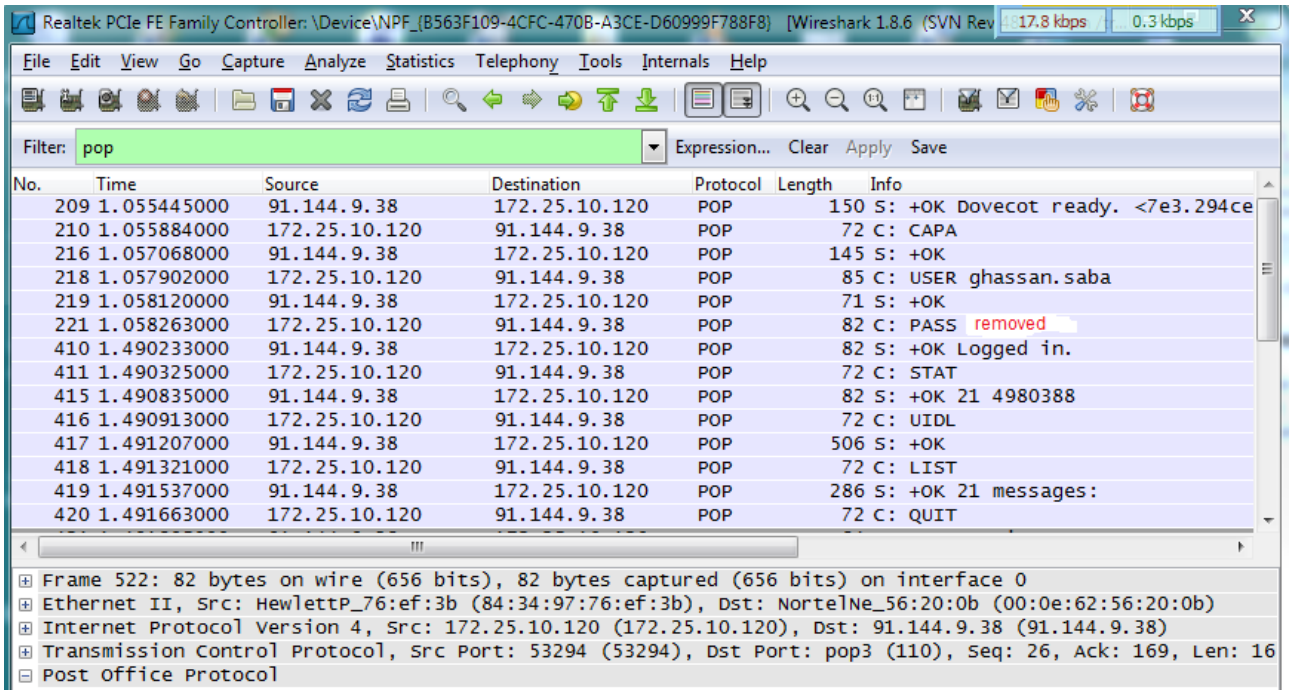
يقوم الطالب بإرسال رسالة من عنوانه في الجامعة الافتراضية إلى أي عنوانين الأول صحيح والثاني غير صحيح باستخدام Outlook ويلتقط الطرود عبر wireshark ومن ثم يجيب على الأسئلة التالية:

1. ما هو اسم مخدم بريد الموقع الذي يستخدمه وما عنوان IP له؟
2. ما هي أرقام بوابات TCP المستخدمة بين المخدم والزبون وهل تتغير حسب الطرود؟
3. ما هو الجواب الذي جرى استقباله من المخدم عند إرسال عنوان غير صحيح؟
4. ما هي آخر رسالة أرسلها الزبون؟
5. ما طبيعة طرد TCP الذي أرسله الزبون بعد الرسالة السابقة وما هي TCP flags؟

### 3.2. تحليل طرود استرجاع البريد باستخدام POP3:

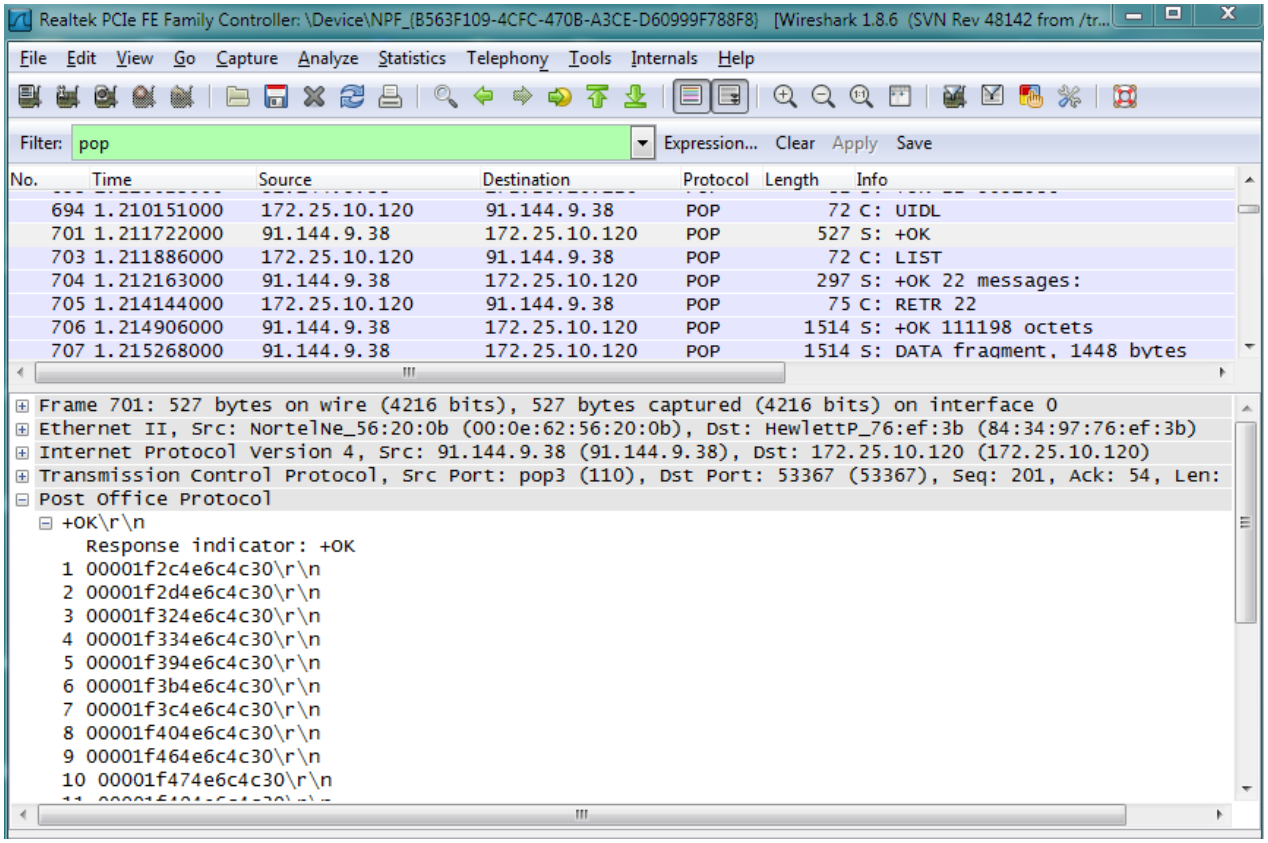
سنحاول ضمن هذا التمرين تشغيل Outlook الذي سيتصل مباشرة بمخدم POP3 المعرف عليه وفتح علب البريد حسب الحسابات المعرفة لديه. مثلاً لدي 3 حسابات كما رأينا في أحد التمارين السابقة.

بعد تشغيل Outlook و wireshark تلاحظ الواجهة التالية:




الشكل 11– تحليل طرود POP3

لاحظ أن الإضافتين CAPA (bility) هي لإظهار الوظائف التي يدعمها مخدم POP3 بينما أمر UniqueIdentifier (UIDL) يدل على أن المخدم قادر على نسب معرف وحيد لكل رسالة تصله الأمر الذي يفيد بالاحتفاظ بالرسالة عند المخدم بعد تحميلها وإمكانية تحميلها مرة أخرى عن طريق UIDL الخاص بها من قبل الزبون. يوضح الشكل التالي استخدام UIDL:




الشكل 12– استخدام UIDL مع POP3

Message 1 UIDL is 0001f2c4e6c4c30

المطلوب من الطالب:

تشغيل Outlook وwireshark ومن ثم الضغط على الزر Send/Receive All Folders والإجابة على الأسئلة التالية:

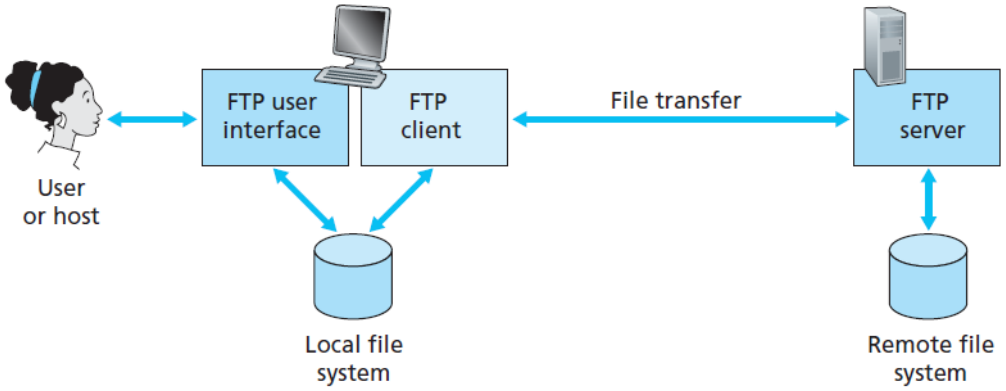
1. ما هي بوابات TCP المستخدمة؟
2. ما هو عنوان IP لمخدم POP3 وهل هو نفسه عنوان مخدم SMTP؟
3. هل جرى استخدام الأمرين: CAPA و UIDL؟ وماذا كانت النتيجة؟
4. ما هو الأمر المستخدم لإظهار الرسائل الموجودة ضمن علبة البريد؟

ملاحظة: تترك للطالب الفرصة لمحاولة الدخول عن طريق Telnet إلى مخدمات SMTP port 25 و POP3 (port110) والتأكد من الأوامر السابقة أو إرسال وتلقي الرسائل. غير أن إمكانية Telnet غير ممكنة دوماً حسب السياسات الأمنية لكل مخدم.

# نقل الملفات وبروتوكولي FTP and TFTP

## 1. بروتوكول نقل الملفات FTP:

يكون المستثمر، عند تنفيذ جلسة FTP، جالساً بمقابل المضيف المحلي ويريد إرسال ملف إلى أو من مضيف بعيد. يجب على المستثمر التعريف عن نفسه باستخدام اسم مستخدم وكلمة مرور. أي أن تفاعل المستثمر مع المخدم يجري عن طريق وكيل للمستثمر وتبادل الملفات يجري بين نظام ملفات محلي ونظام ملفات بعيد حسب ما هو موضح في الشكل التالي:




الشكل 1– نقل الملفات باستخدام بروتوكول FTP

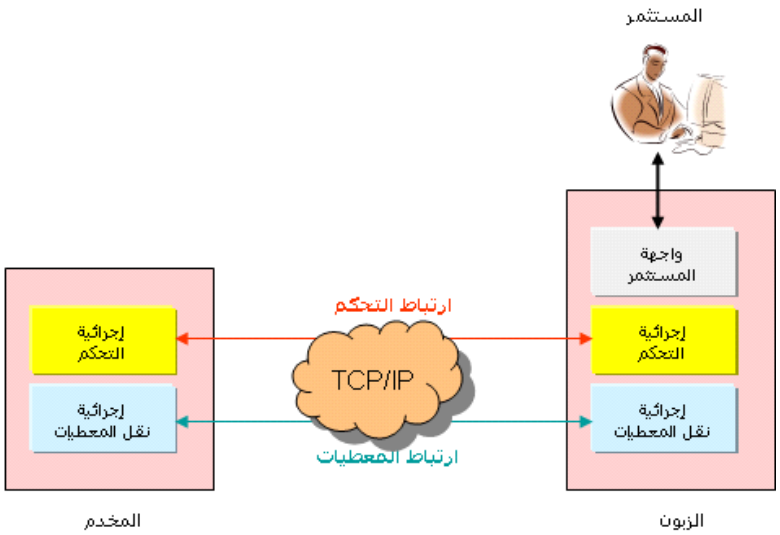
نذكر من الصعوبات التي يمكن أن يواجهها بروتوكول نقل الملفات بين مضيفين ينتميان إلى نظامي تشغيل مختلفين والتي استطاع FTP حلها:

- يمكن أن تكون اصطلاحات تسمية الملفات مختلفة.
- يمكن أن يستخدم كل نظام تشغيل آلية تمثيل معطيات مختلفة عن الآخر.
- يمكن أن تكون أنظمة المجلدات أو الدلائل Directory structure مختلفة.

يختلف FTP عن غيره بكونه ينشئ ارتباطي TCP(2 TCP connections) بين المضيفين: يُستخدم الارتباط الأول لنقل المعطيات بينما يُستخدم الارتباط الثاني لنقل معلومات التحكم (أوامر وأجوبة). يعود سبب إنشاء ارتباطين إلى أن قواعد إدارة معلومات التحكم تكون بسيطة لأنها لا تحوي أكثر من سطر أوامر أو أجوبة عليها بينما تكون قواعد نقل المعطيات المتعددة الأشكال أكثر تعقيداً.

يستخدم FTP بوابتين معروفتين Well-known TCP ports: البوابة 21 للارتباط الخاص بالتحكم والبوابة 20 للارتباط الخاص بالمعطيات.

يبين الشكل التالي النموذج الأساسي لبروتوكول FTP.




الشكل 2– آلية عمل FTP

يحوي الزبون على ثلاثة مكونات: واجهة المستثمر وإجرائية التحكم وإجرائية نقل المعطيات. أما المخدم فهو يحوي على مكونين: إجرائية التحكم وإجرائية نقل المعطيات.

يبقى ارتباط التحكم موصولاً طالما أن جلسة FTP التفاعلية مفتوحة. أما ارتباط المعطيات فيجري فتحه عند طلب إرسال ملف ويجري إغلاقه عند انتهاء عملية الإرسال.

### 1.1. الارتباطات Connections:

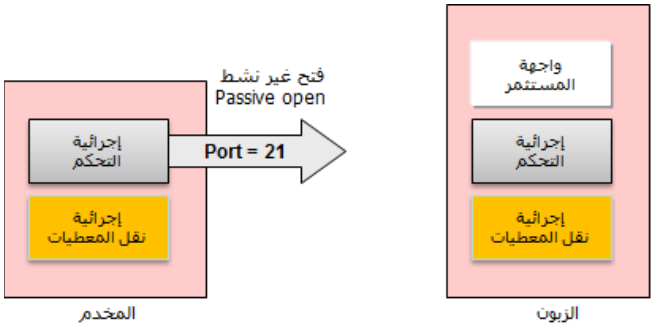
يستخدم كل نوع من الارتباطات استراتيجية ورقم بوابة مختلف.

ارتباط التحكم:

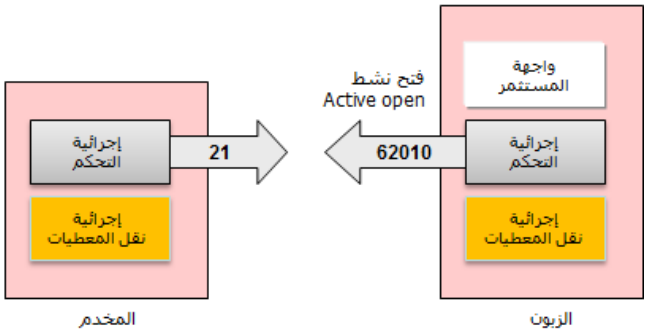
تمر عملية فتح ارتباط التحكم بخطوتين:

1. يطلب المخدم فتح غير نشط passive open للبوابة المعرفة مسبقاً رقم 21 وينتظر الزبون.
2. يستخدم الزبون رقم بوابة مؤقتة ويطلب فتح نشط Active Open.

يبقى الارتباط مفتوحاً طيلة فترة الاتصال. تكون الخدمة المستخدمة على مستوى بروتوكول IP من نوع "تقليل التأخير" minimize delay وذلك لأن الاتصال تفاعلي بين المستثمر والمخدم. يبين الشكل التالي الاتصال بين المخدم والزبون.




أ – فتح غير نشط من قبل المخدم



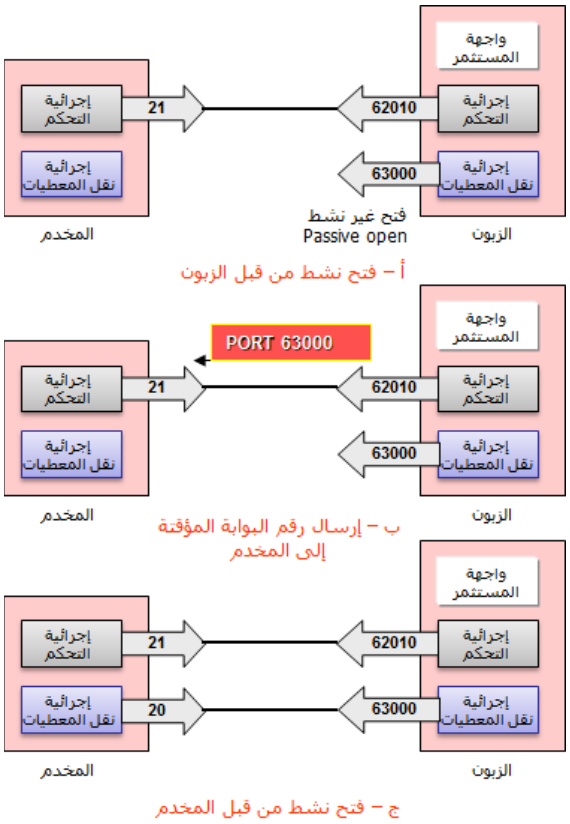

ب – فتح نشط من قبل الزبون

الشكل 3– فتح ارتباط التحكم

ارتباط المعطيات:

يستخدم ارتباط المعطيات البوابة المعروفة رقم 20 في طرف المخدم. يجري إنشاء ارتباط المعطيات حسب ما يلي:

1. يطلب الزبون فتح غير نشط لبوابة مؤقتة.
2. يرسل الزبون رقم البوابة السابقة إلى المخدم باستخدام أمر PORT Command.
3. يستقبل المخدم رقم البوابة ويطلب فتح نشط للبوابة المعروفة رقم 20 وبرقم البوابة المستقبلة.




الشكل 4– إنشاء ارتباط المعطيات

التراسل عبر ارتباط التحكم:

يستخدم FTP مجموعة محارف NVT ASCII. يجري التراسل باستخدام أوامر وأجوبة. يمتد الأمر على سطر واحد ينتهي بمحرفين: Carriage return and line feed (CR+LF).

التراسل عبر ارتباط المعطيات:

يختلف ارتباط المعطيات عن ارتباط التحكم بكون الأول مستخدماً لنقل الملفات. لذلك يجب على الزبون تعريف نوع الملف المطلوب نقله وبنية المعطيات ونمط الإرسال. يجري التحضير لعملية إرسال الملف من خلال ارتباط التحكم.

نوع الملف:يدعم FTP أنواع الملفات التالية:

- ASCII File: وهي الصيغة الافتراضية لنقل الملفات النصية. يجري هنا ترميز المحارف وفق NVT ASCII. يحول المرسل الملف من صيغته الأصلية إلى صيغة NVT ASCII بينما يحول المستقبل المعطيات المستقبلة إلى تمثيله الخاص.
- EBCDIC File: يمكن أيضاً نقل الملف باستخدام ترميز EBCDIC إذا كان أحد الأطراف يستخدم هذا الترميز.

- Image File: وهي الصيغة الافتراضية لنقل الملفات وفق الصيغة الثنائية Binary files. يجري هنا إرسال الملف كتدفق مستمر من البتات بدون أي تفسير أو ترميز. يفيد هذا النوع في نقل الملفات الثنائية مثل البرامج المترجمة Compiled programs.

يجب، عند استخدام ASCII أو EBCDIC لترميز الملفات، إضافة خاصية أخرى لتعريف قابلية طباعة الملف.

- غير مطبوع Nonprint. وهي الصيغة الافتراضية لنقل الملفات النصية. لا يحوي الملف توصيفات عمودية خاصة بالطباعة. أي أنه لا يمكن طباعة الملف بدون معالجة بسبب عدم وجود محارف خاصة بالحركة العمودية لرأس الطباعة.
- Telnet: يحوي الملف محارف عمودية من نوع NVT ASCII مثل CR (Carriage Return) و LF (Line feed) و NL (New Line) و VT (Vertical tab). أي يكون الملف قابل للطباعة بعد النقل.

بنيةالمعطيات: يمكن أن تأخذ بنية المعطيات أحد الأنواع التالية:

- بنية ملف File structure. وهي القيمة الافتراضية حيث يكون الملف عبارة عن تدفق مستمر من البايتات.
- بنية سجل Record structure. يجري هنا تقسيم الملف إلى سجلات. يمكن استخدام هذا النوع مع الملفات النصية.
- بنية صفحات Page structure. يجري تقسيم الملف إلى صفحات تحوي كل منها رقم الصفحة وترويسة. يمكن تخزين أو النفاذ إلى الصفحات بطريقة عشوائية أو تسلسلية.

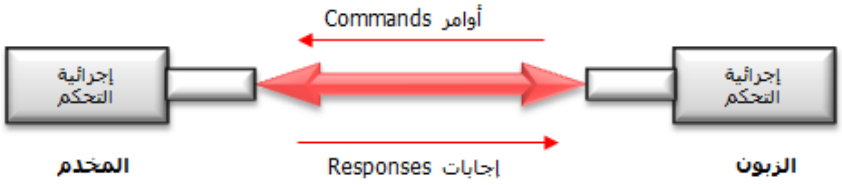
نمط الإرسال: يوجد ثلاثة أنماط لنقل الملفات:

- النمط الدفقي stream mode. وهو النمط الافتراضي. يجري هنا توصيل المعطيات من FTP إلى TCP على أنها دفقة بايتات مستمرة. تكون طبقة TCP مسؤولة عن وضع المعطيات ضمن مقاطع ذات أطوال مناسبة. لا توجد هنا حاجة لتحديد نهاية الملف وذلك لأن إغلاق المرسل لارتباط المعطيات يؤشر إلى نهاية الملف. إذا كانت المعطيات مقسمة إلى سجلات فيجب أن يحوي كل سجل محرفاً مكوناً من بايتاً واحداً يؤشر إلى نهاية السجل EOR (End Of Record) إضافةً إلى محرفاً آخر مكوناً أيضاً من بايتاً واحداً يؤشر إلى نهاية الملف EOF (End Of File).

- النمط الكتلي Block mode: يجري هنا توصيل المعطيات من FTP إلى TCP كتلة بكتلة. يجب هنا استخدام ترويسة لكل كتلة مكونة من 3 بايتات. يدعى البايت الأول "واصف الكتلة" Block descriptor بينما يعرف البايتان الباقيان قياس الكتلة بالبايت.
- النمط المضغوط Compressed mode. يمكن ضغط الملف إذا كان حجمه كبيراً. طريقة الضغط المستخدمة هي الترميز الطولي run-length encoding حيث يجري استبدال المعلومات المكررة بوضع نسخة واحدة من المعلومة وإضافة عدد التكرارات. في حالة الملفات النصية، فإن التكرار يحدث في الفراغات Spaces بينما يحدث في الملفات الثنائية في المحرف عديم القيمة null character.

### 2.1. معالجة الأوامر Command processing:

يستخدم FTP ارتباط التحكم لإنشاء وصلة تراسل بين إجرائية التحكم لدى الزبون وإجرائية التحكم لدى المخدم. خلال فترة التراسل هذه، يقوم الزبون بإرسال الأوامر إلى المخدم بينما يقوم المخدم بإرسال الإجابات إلى الزبون (انظر الشكل التالي).




الشكل 5– معالجة الأوامر

الأوامر:

تكون الأوامر على صيغة ASCII uppercase مع أو بدون محددات. يمكننا تقسيم الأوامر إلى ست مجموعات: أوامر النفاذ access commands وأوامر إدارة الملفات file management commands وأوامر صياغة الملفات file formatting commands وأوامر تعريف البوابات port defining commands وأوامر نقل الملفات file transferring commands وأوامر متفرقة miscellaneous commands.

1. أوامر النفاذ Access commands. تسمح هذه الأوامر للمستثمر بالنفاذ إلى المخدم البعيد. يبين الجدول التالي الأوامر المستخدم ضمن هذه المجموعة.

| الأمر | المحدد(ات) | الوصف |
|---|---|---|
| USER | تعريف المستثمر User id | معلومات عن المستثمر |
| PASS | كلمة المرور | كلمة مرور المستثمر |
| ACCT | الحساب المطلوب تحميله | معلومات عن الحساب |
| REIN | | إعادة التهيئة Reinitialize |
| QUIT | | الخروج من النظام |
| ABOR | | إيقاف الأمر السابق |

الشكل 6– أوامر النفاذ

2. أوامر إدارة الملفات File Management commands.تسمح هذه الأوامر للمستثمر بالنفاذ إلى نظام الملفات الموجود على الحاسب البعيد. فهي تسمح له بالملاحة عبر بنية الدليل Directory وخلق دليل جديد وحذف الملفات وغيرها. يبين الجدول التالي الأوامر الأساسية الموجودة ضمن هذه المجموعة.

| الأمر | المحدد(ات) | الوصف |
|---|---|---|
| CWD | اسم الدليل | الانتقال إلى دليل جديد |
| CDUP | | الانتقال إلى الدليل الأب |
| DELE | اسم الملف | حذف ملف |
| LIST | اسم الدليل | إدراج لائحة أسماء الدلائل الجزئية أو الملفات |
| NLIST | اسم الدليل | إدراج لائحة أسماء الدلائل الجزئية أو الملفات بدون خاصيات أخرى |
| MKD | اسم الدليل | إنشاء دليل جديد |
| PWD | | إظهار اسم الدليل الحالي |
| RMD | اسم الدليل | حذف مجلد |
| RNFR | اسم الملف (الاسم القديم) | الاسم القديم لملف مراد تغيير اسمه |
| RNTO | اسم الملف (الاسم الجديد) | تغيير الاسم إلى اسم جديد |
| SMNT | اسم نظام الملفات | تنصيب نظام ملفات |

الشكل 7– أوامر إدارة الملفات

3. أوامر صياغة المعطيات Data Formatting commands. تسمح هذه الأوامر للمستثمر بتعريف بنية المعطيات ونمط الملفات ونمط الإرسال. يجري استخدام الصياغة المعرفة من قبل أوامر نقل الملفات. يبين الشكل التالي بعض الأوامر التي تستخدمها هذه المجموعة.

| الأمر | المحدد(ات) | الوصف |
|---|---|---|
| **TYPE** | A (ASCII)<br>E (EBCDIC)<br>I (Image)<br>N (Nonprint)<br>T (Telnet) | تعريف نوع الملف وفي حال الحاجة طريقة الطباعة |
| **STRU** | F (File)<br>R (Record)<br>P (Page) | تعريف طريقة تنظيم المعطيات |
| **MODE** | S (Stream)<br>B (Block)<br>C (Copressed) | تعريف نمط الإرسال |

الشكل 8– أوامر صياغة المعطيات

4. أوامر تعريف البوابة Port defining commands. تعرف هذه الأوامر رقم البوابة لارتباط المعطيات من جهة الزبون. يوجد طريقتان لتحقيق ذلك: يمكن للزبون، وفق الطريقة الأولى التي تستخدم PORT Command، اختيار رقم بوابة مؤقت ويرسله إلى المخدم بواسطة "فتح غير نشط". يستخدم المخدم رقم البوابة وينشئ "فتح نشط". أما في الطريقة الثانية المعتمدة على PASV Command، فيطلب الزبون من المخدم اختيار رقم البوابة. يحقق المخدم "فتح غير نشط" على هذه البوابة ويرسل رقم البوابة ضمن الجواب (انظر الجواب رقم 227 ضمن الشكل 12). ينشئ الزبون "فتح نشط" باستخدام رقم البوابة تلك. يبين الشكل التالي أوامر تعريف البوابات.

| الأمر | المحدد(ات) | الوصف |
|---|---|---|
| **PORT** | معرِّف مكون من 6- digit | يختار الزبون البوابة |
| **PASV** | | يختار المخدم البوابة |

الشكل 9– أوامر تعريف البوابة

5. أوامر نقل الملفات File Transfer commands. تسمح هذه الأوامر للمستثمر بنقل الملفات. يبين الشكل التالي الأوامر الشائعة ضمن هذه المجموعة.

| الأمر | المحدد(ات) | الوصف |
|---|---|---|
| RETR | اسم(اء) الملف(ات) | استحضار الملفات؛ نقل الملف(ات) من المخدم إلى الزبون |
| STOR | اسم(اء) الملف(ات) | تخزين الملفات؛ نقل الملف(ات) من الزبون إلى المخدم |
| APPE | اسم(اء) الملف(ات) | مشابه لـ STOR لكن إذا كان الملف موجوداً فتضاف المعطيات إليه |
| STOU | اسم(اء) الملف(ات) | مشابه لـ STOR لكن اسم الملف يجب أن سيكون وحيداً ضمن الدليل؛ ويجب أن لا تجري الكتابة فوق الملف الموجود |
| ALLO | اسم(اء) الملف(ات) | تخصيص مساحة تخزينية لدى المخدم من أجل الملف(ات) |
| REST | اسم(اء) الملف(ات) | وضع مؤشر الملف على نقطة محددة |
| STAT | اسم(اء) الملف(ات) | إعادة حالة الملفات |

الشكل 10– أوامر نقل الملفات

6. أوامر متفرقة Miscellaneous commands. تعطي هذه الأوامر بعض المعلومات إلى مستثمر FTP من جهة الزبون. يبين الشكل التالي بعض هذه الأوامر.

| الأمر | المحدد(ات) | الوصف |
|---|---|---|
| HELP | | طلب معلومات عن المخدم |
| NOOP | | اختبار كون المخدم على قيد الحياة |
| SITE | أوامر | تحديد الأوامر الخاصة بالموقع |
| SYST | | السؤال عن نظام التشغيل لدى المخدم |

الشكل 11– الأوامر المتفرقة

### الأجوبة Responses:

يولد كل أمر FTP جواباً على الأقل. يشمل الجواب على قسمين: رقم مؤلف من 3 خانات يتبعه نص. يعرف الرقم رمز الجواب؛ يعرف النص الوسطاء المطلوبة أو تفسيرات إضافية. سنمثل الخانات الثلاث بالرمز xyz ونبين فيما يلي مدلول كل خانة منها:

الخانة الأولى. تُعرف الخانة الأولى إلى اليسار حالة الأمر. يمكن استعمال 5 قيم محتملة:

- 1yz (positive preliminary reply) . تدل على أن العملية قد بدأت وأن المخدم سيرسل جواباً آخر قبل قبول أمر آخر.
- 2yz (positive completion reply). تدل على اكتمال العملية وأن المخدم سيقبل أمراً جديداً.
- 3yz (positive intermediate reply). تدل على قبول الأمر لكننا نحتاج إلى معلومات إضافية.
- 4yz (transient negative completion reply). تدل على عدم إجراء العملية لكن الخطأ مؤقت ويمكن إعادة الأمر لاحقاً.
- 5yz (permanent negative completion reply). لم يتم قبول الأمر ولا توجد فائدة من إعادة إرساله مرة أخرى.

الخانة الثانية. تعرف الخانة الثانية أيضاً حالة الأمر. يمكن هنا استخدام 6 حالات محتملة:

- x0z (Syntax).
- x1z (information).
- x2z (connection).
- x3z (authentication and accounting).
- x4z (unspecified).
- x5z (filesystem).

الخانة الثالثة. تزود الخانة الثالثة معلومات إضافية كما هو مبين في الشكل التالي.

| Code | Description |
|---|---|
| **Positive Preliminary Reply** | |
| 120 | Service will be ready shortly |
| 125 | Data connection open; data transfer will start shortly |
| 150 | File status is OK; data connection will be open shortly |
| **Positive Completion Reply** | |
| 200 | Command OK |
| 211 | System status or help reply |
| 212 | Directory Status |
| 213 | File status |
| 214 | Help message |
| 215 | Naming the system type (operating system) |
| 220 | Service ready |
| 221 | Service closing |
| 225 | Data connection open |
| 226 | Closing data connection |
| 227 | Entering passive mode; server sends its IP address and port number |
| 230 | User login OK |
| 250 | Request file action OK |
| **Positive Intermediate Reply** | |
| 331 | User name OK; password is needed |
| 332 | Need account for logging |
| 350 | The file action is pending; more information needed |
| **Transient Negative Completion Reply** | |
| 425 | Cannot open data connection |
| 426 | Connection closed; transfer aborted |
| 450 | File action not taken; file not available |
| 451 | Action aborted; local error |
| 452 | Action aborted; insufficient storage |
| **Permanent Negative Completion Reply** | |
| 500 | Syntax error; unrecognized command |
| 501 | Syntax error in parameters or arguments |
| 502 | Command not implemented |
| 503 | Bad sequence of commands |
| 504 | Command parameters not implemented |
| 530 | User not logged in |
| 532 | Need account for storing file |
| 550 | Action is not done; file unavailable |
| 552 | Requested action aborted; exceeded storage allocation |
| 553 | Requested action not taken; file name not allowed |

الشكل 12– الإجابات Responses

### نقل الملفات:

يجري نقل الملفات عبر ارتباط المعطيات وتحت إشراف أوامر التحكم المرسلة عبر ارتباط التحكم. يجب الانتباه هنا إلى أن نقل الملفات عنطريق FTP يعني أحد الحالات التالية:

- يجب نسخ ملف من المخدم إلى الزبون. هذا ما يدعى باستحضار الملف Retrieving a file. يجري تحقيقه بإشراف أمر RETR.
- يجب نسخ الملف من الزبون إلى المخدم. هذا ما يدعى بتخزين الملف Storing a file. يجري تحقيقه بإشراف أمر STOR.
- يجب إرسال لائحة بالدلائل أو بأسماء الملفات من المخدم إلى الزبون. يجري تحقيقه بإشراف أمر LIST. لاحظ أن FTP يرسل لائحة الدلائل على أنها ملف عبر ارتباط المعطيات.

### مثال 1:

يبين الشكل التالي مثالاً عن استخدام FTP لاستحضار لائحة العناصر الموجودة داخل دليل.

| Control connection | | Data connection | |
|---|---|---|---|
| Client | Server | Client | Server |
| Control Process | Control process | Data Process | Data Process |
| 220 (service Ready) | | | |
| User Alice | | | |
| 331 (User Name OK. Password?) | | | |
| Pass ****** | | | |
| 230 (User Login OK) | | | |
| PORT 8888 | | | |
| 150 (Data Connection open shortly) | | | |
| List /usr/Alice/documents | | | |
| 125 (Data connection OK) | | | |
| | | List of files or directories | |
| 226 (Closing data connection) | | | |
| QUIT | | | |
| 221 (Service Closing) | | | |

الشكل 13– المثال 1

مثال 2:

يبين الشكل التالي جلسة FTP حقيقية لما شاهدناه في المثال 1. توضح الكلمات الملونة بالأحمر إدخالات المستخدم على ارتباط التحكم.

```
ftp ietf.org
Connected to ietf.org
220 FTP server ready
User <ietf.org:<none>>: anonymous
331 Anonymous Login ok, send your complete email address as your
password
Password: 123@gmail.com
230 Anonymous access granted, restriction apply
pwd
257 "/" is the current directory
cdrfc
250-*=================================================================*
 *                                                                   *
 * This directory is maintained by the RFC Editor.  If you experience *
 * any problems, please report them to rfc-editor@rfc-editor.org.     *
 *                                                                   *
 *=================================================================*
250 CWD command successful
ftp>quote "pasv"
227 Entering Passive Mode (4,31,198,44,234,112).
```

الشكل 14– المثال 2

### 3.1. FTP المغفل AnonymousFTP:

كما لاحظنا سابقاً، يحتاج المستثمر على اسم مستثمر وكلمة مرور حتى يستطيع الدخول إلى مخدم FTP. لكن بما أن الكثير من المواقع يمتلك ملفات متاحة للعموم، أي أنه لا توجد حاجة لإدخال اسم مستثمر وكلمة مرور للحصول على هذه الملفات. لذلك يستطيع المستثمر، في هذه الحالة، استخدام الاسم anonymous وكلمة المرور guest.

تكون صلاحيات المستثمر محدودة جداً من حيث الملفات أو الأوامر كامتلاك صلاحية نقل الملفات دون إمكانية الملاحة بين الدلائل.

### 4.1. برنامج SFTP:

Secure FTP هو برنامج تنفيذي يعمل ضمن بيئة Unix/Linux ويكون عادةً مضمناً ضمن Secure Shell (SSH). بعد إقامة ارتباط آمن بين طرفين باستخدام SSH، يمكننا استخدام SFTP كأحد البرامج التطبيقية التي تستخدم هذا الارتباط الآمن. يعمل SFTP بطريقة تفاعلية مثل FTP عن طريق تحقيق مجموعة من الأوامر.

## 2. بروتوكول TFTP :

نحتاج في بعض الحالات إلى نسخ ملف ما دون تعقيدات بروتوكول FTP. عندما يقلع مسير Router أو محطة دون قرص صلب فإنهما يحتاجان إلى تحميل ملف إقلاع. جرى تصميم بروتوكول Trivial File Transfer Protocol (TFTP) لهذا النوع من الملفات. بساطته تجعله قادراً على الوضع ضمن ذاكرة ROM مع بروتوكولات بسيطة أخرى مثل UDP وIP. يستطيع TFTP قراءة ملف أو كتابته: القراءة تعني نسخ الملف من المخدم (تحميل) بينما الكتابة (رفع) تعني نسخ الملف إلى المخدم.

يستخدم بروتوكول TFTP خدمات UDP على البوابة رقم 69.

### 1.2. الرسائل Messages:

يوجد 5 أنواع من رسائل TFTP: RRQ, WRQ, DATA, ACK, and ERROR.

رسالة RRQ:

رسالة القراءة Read Request (RRQ) يستخدمها الزبون لإنشاء ارتباط لقراءة المعطيات من المخدم. يبين الشكل التالي صيغة هذه الرسالة:

| OpCode = 1 | File name | All 0s | Mode | All 0s |
|---|---|---|---|---|
| 2 bytes | Variable | 1 byte | Variable | 1 byte |

الشكل 15– صيغة رسالة RRQ

حقول رسالة RRQ هي:

- OpCode. يحوي رمز العملية وتكون قيمة الحقل هي "1" في حالة RRQ.
- File name. سلسلة متغيرة الطول مرمزة وفق ASCII لتعريف اسم الملف المطلوب قراءته. يجري استخدام بايت مؤلف من 8 أصفار للدلالة على نهاية اسم الملف.
- Mode. سلسلة متغيرة الطول للدلالة على نمط النقل وهي تنتهي أيضاً ببايت مؤلف من 8 أصفار. النمط هو إما "netascii" أو "octet" للملفات الثنائية.

رسالة WRQ:

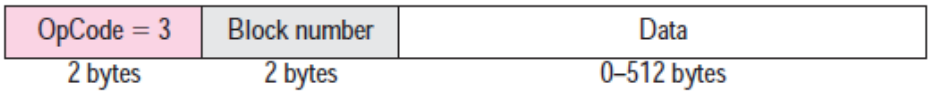
رسالة الكتابة Write Request (WRQ) يستخدمها الزبون لإنشاء ارتباط مع المخدم لكتابة معلومات على المخدم. حقول الرسالة هي:

| OpCode = 2 | File name | All 0s | Mode | All 0s |
|---|---|---|---|---|
| 2 bytes | Variable | 1 byte | Variable | 1 byte |

الشكل 16– صيغة رسالةWRQ

رسالة DATA:

يجري استخدام رسالة DATA من قبل الزبون أو المخدم لإرسال كتل المعطيات. يبين الشكل التالي صيغة هذه الرسالة:

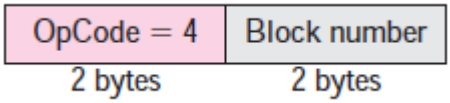



الشكل 17– صيغة رسالة DATA

- OpCode=3 للدلالة على حقل من نوع DATA
- Block number.يجري استخدام هذا الحقل للأرقام التسلسلية لكتل المعطيات من قبل الزبون أو المخدم. يبدأ الترقيم عند القيمة "1" وهو ضروري للإقرارات.
- Data. يجب أن يكون قياس هذا الحقل 512 Bytes تماماً لجميع كتل المعطيات ما عدا الكتلة الأخيرة الذي يتراوح قياسه بين 0-511 Bytes. أي أن استقبال كتلة بقياس أقل من 512 Bytes يدل على كون الكتلة هي الأخيرة أو مؤشر End-of-file.

رسالة ACK:

يجري استخدامه من المخدم أو من الزبون الإقرار استلام كتلة معطيات. طول الإقرار هو 4 Bytes وصيغتها مبينة في الشكل التالي:



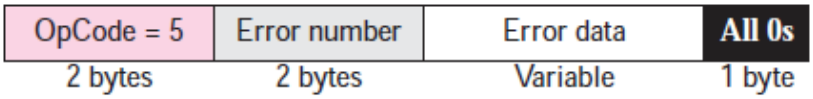

الشكل 18– صيغة رسالة ACK

- OpCode=4 للدلالة على رسالة الإقرار
- Block number. تؤشر على رقم الكتلة المستلمة

يجري أيضاً إرسال إقرار كجواب على رسالة WRQ. يرسلها المخدم للدلالة على كونه جاهزاً لاستقبال المعطيات من الزبون. تكون في هذه الحالة قيمة حقل Block number مساوية للصفر.

رسالة ERROR:

يجري استخدامها من قبل المخدم أو الزبون في حالة عدم إمكانية إنشاء ارتباط بينهما أو عند وجود مشكلة أثناء الاتصال. يمكن إرسالها جواباً على WRQ أو RRQ. يبين الشكل التالي صيغة هذه الرسالة:



الشكل 19– صيغة رسالة ERROR

- Opcode=5
- Error number. يعرف نوع الخطأ، راجع الجدول التالي لمعرفة دلالات الأرقام المختلفة للأخطاء:

| *Number* | *Meaning* | *Number* | *Meaning* |
|---|---|---|---|
| 0 | Not defined | 5 | Unknown port number |
| 1 | File not found | 6 | File already exists |
| 2 | Access violation | 7 | No such user |
| 3 | Disk full or quota exceeded | | |
| 4 | Illegal operation | | |

الشكل 20– أرقام الأخطاء ومعانيها

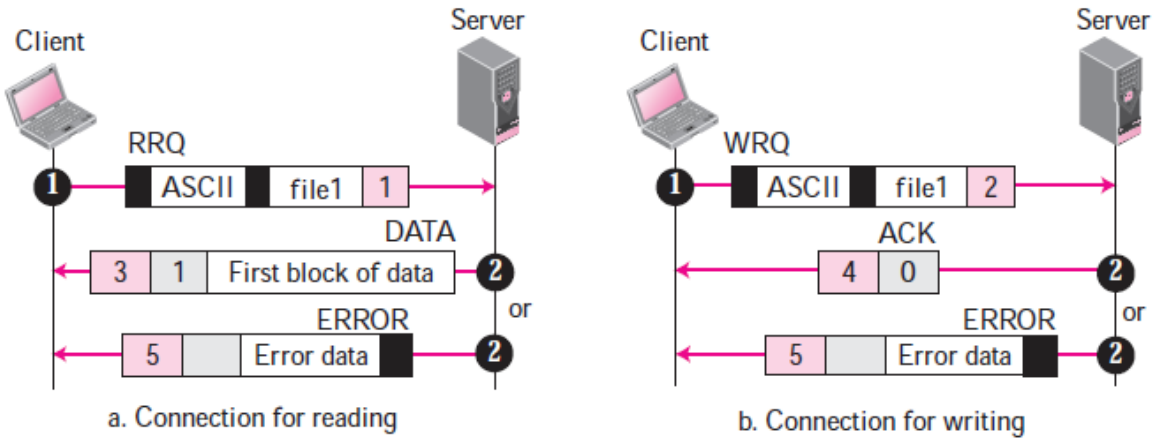
- Error Data. يحوي هذا الحقل القيمة النصية للخطأ.

### 2.2. الارتباط Connection:

بما أن TFTP يستخدم خدمات UDP فلا يوجد مفهوم إنشاء الارتباط أو انهائه حسب مفهوم TCP أي المصافحة الثلاثية. يرسل UDP كل كتلة معطيات بعد تغليفها بطرد UDP. نحتاج إلى ارتباط ضمن TFTP لضمان أن جميع الكتل المشكلة للملف ستصل إلى الطرف الآخر.

#### 1.2.2. إنشاء الارتباط Connectionestablishment:

يختلف إنشاء الارتباط عند قراءة ملف عن تلك عند كتابة ملف (راجع الشكل التالي).




الشكل 21– إنشاء الارتباط

نلاحظ أنه عند القراءة، يرسل الزبون RRQ مبيناً معها اسم الملف المطلوب ونوع الإرسال. في حال عدم وجود مشاكل، يجيب المخدم بإرسال أول كتلة معطيات. أما في حال وجود مشكلة ما فإنه يجيب برسالة ERROR حسب نوع الخطأ.

أما عند الكتابة، فيستخدم الزبون رسالة WRQ مبيناً عليها اسم الملف ونوعه. هنا يجيب المخدم برسالة ACK مع الرقم التسلسلي "0" للدلالة على استعداده لاستقبال الملف أو رسالة خطأ.

### 2.2.2. انهاء الارتباط Connection Termination:

يجري انهاء الارتباط عند استقبال كتلة معطيات طولها أقل من 512 bytes الذي يؤشر على آخر كتلة معطيات.

### 3.2.2. نقل المعطيات DataTransfer:

تجري مرحلة نقل المعطيات بعد إنشاء الارتباط وقبل انهائه. لتجاوز عدم وثوقية بروتوكول UDP، يجري هنا تجزئة الملف المنقول إلى كتل كل منها ذو طول 512 bytes ما عدا الكتلة الأخيرة. يتعامل TFTP مع آليات اكتشاف وتصحيح الأخطاء والتحكم بالتدفق.

#### التحكم بالتدفق Flow Control:

يقوم TFTP بإرسال كتلة معطيات وتشغيل مؤقت زمني وينتظر الإقرار. إذا استقبل المرسل الإقرار قبل انقضاء المؤقت الزمني فإنه يرسل الكتلة التالية. أمل إذا انقضى المؤقت الزمني قبل استقبال الإقرار فإنه يعيد إرسال الكتلة السابقة ويعيد تشغيل المؤقت. وهكذا يتم تحقيق التحكم بالتدفق عن طريق ترقيم كتل المعطيات وانتظار الإقرارات.

#### تحميل ملف:

عندما يريد الزبون قراءة (تحميل) ملف، فإنه يرسل RRQ. يجيب المخدم عن طريق رسالة DATA تحوي أول كتلة معطيات مع الرقم التسلسلي "1" إذا لم توجد مشاكل.

#### تخزين ملف:

عندما يريد الزبون تخزين Store ملف، فإنه يرسل رسالة WRQ. يجيب المخدم برسالة ACK باستخدام الرقم التسلسلي "0" أو يرسل رسالة الخطأ المناسبة. بعد أن يستقبل الزبون للإقرار فإنه يبدأ بإرسال كتل المعطيات بدءاً بالرقم "1".

#### التحكم بالأخطاء Error Control:

يعتمد TFTP أسلوب تحكم بالأخطاء تناظري، أي أن كلاً من المرسل والمستقبل يستخدم مؤقت زمني. المرسل يستخدم المؤقت لكتل المعطيات بينما يستخدم المستقبل المؤقت للإقرارات. عندما تضيع كتلة معطيات فإن

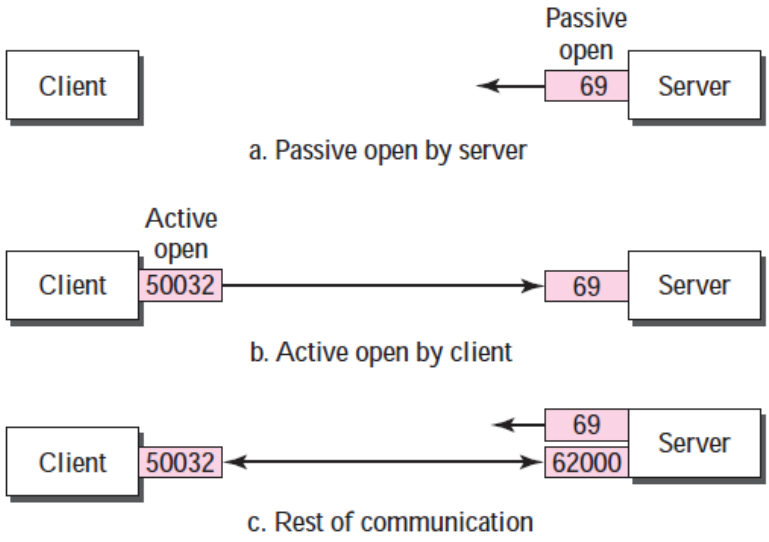
المرسل يعيد إرسالها بعد انقضاء المؤقت الزمني وكذلك الأمر بالنسبة للإقرار الذي نشغل له مؤقت ونعيد إرسال الإقرار في حال انقضى هذا المؤقت.

نحتاج إلى التحكم بالأخطاء لتجاوز مشاكل تشوه الرسائل أو ضياعها أو وصول نسخ مكررة منها أو تكرار الإقرارات.

### 3.2. بوابات UDP المستخدمة:

عند التعامل مع بوابات UDP، فإن إجرائية المخدم تستدعي PassiveOpen على البوابة المعروفة وتنتظر الزبون ليطلب Activeopen على بوابة متغيرة. بعد انشاء الارتباط بين المخدم والزبون، يمكن تحقيق تبادل المعطيات بينهما على هاتين البوابتين.

يحجز بروتوكول TFTP البوابة رقم 69 لتحقيق الاتصال الأولي مع الزبون ومن ثم يبدل إلى بوابة متغيرة لمتابعة الاتصال كما هو موضح في الشكل التالي:



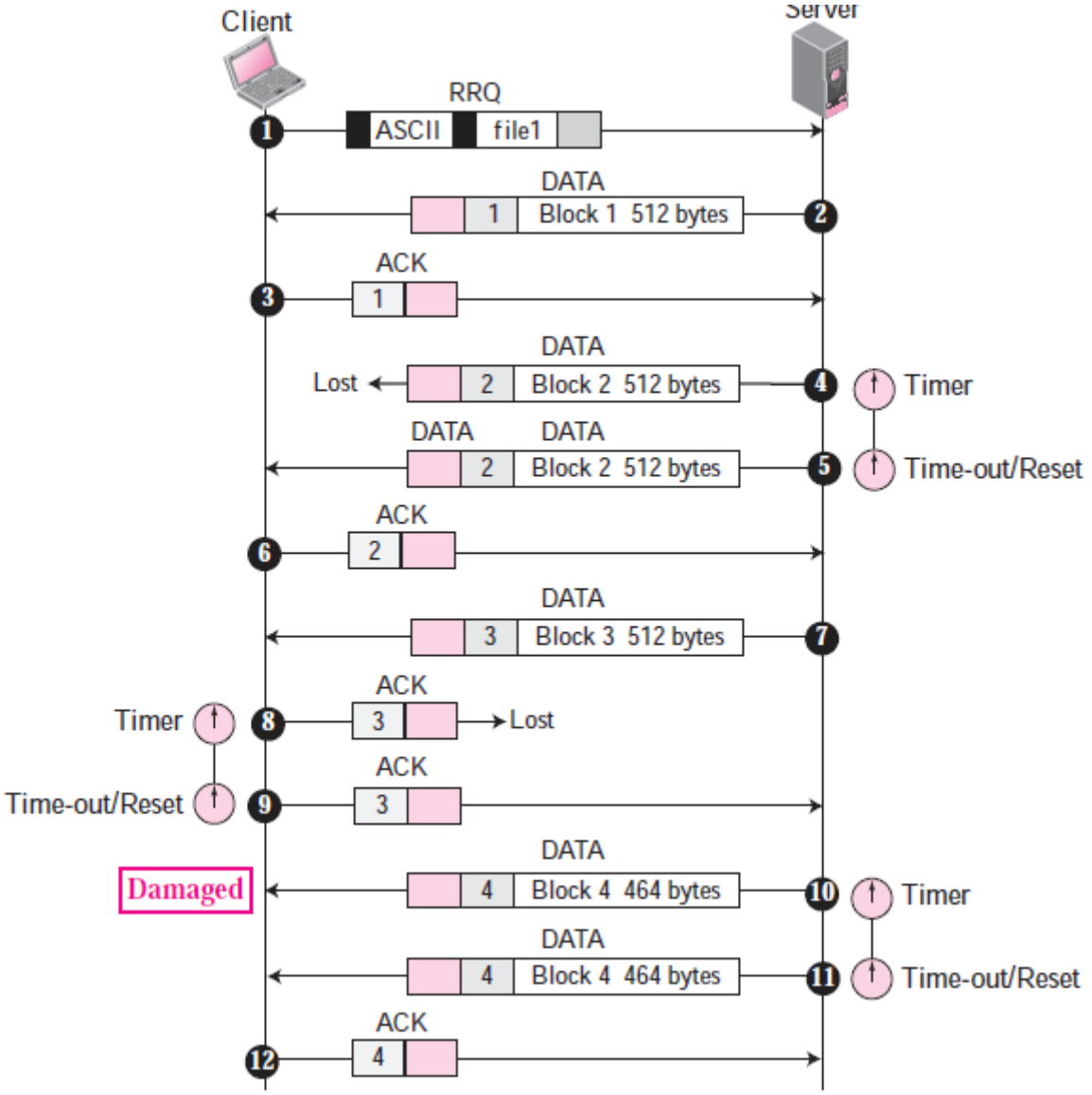

الشكل 22– أرقام البوابات التي يستخدمها TFTP

الخطوات هي:

- يتنصت (أي Passiveopen) مخدم TFTP على البوابة 69
- يفتح الزبون الارتباط باستخدام أحد البوابات المصدر العشوائية مع البوابة الوجهة 69. يجري تحقيق ذلك عن طريق رسالة RRQ أو WRQ.
- يفتح المخدم الارتباط باستخدام رقم بوابة مصدر يكون عشوائي ويستخدم عنوان البوابة التي استقبلها كبوابة وجهة. يرسل المعطيات أو الإقرارات أو رسائل ERROR على هذه البوابات.

### 4.2. مثال عن TFTP:

يوضح الشكل التالي مثالً عن نقل ملف باستخدام بروتوكول TFTP.

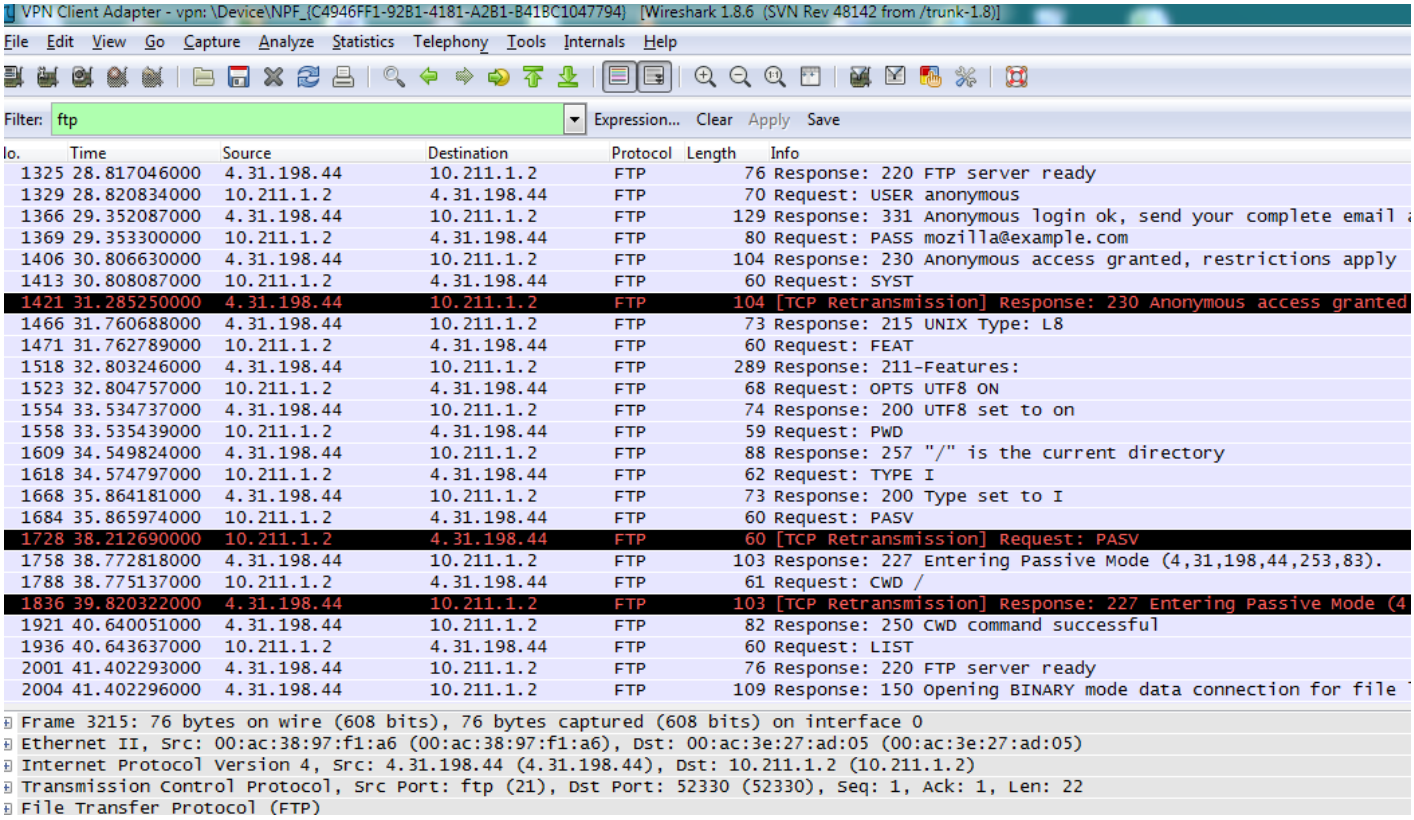



الشكل 23– مثال عن TFTP

## 3. تمارين عن FTP:

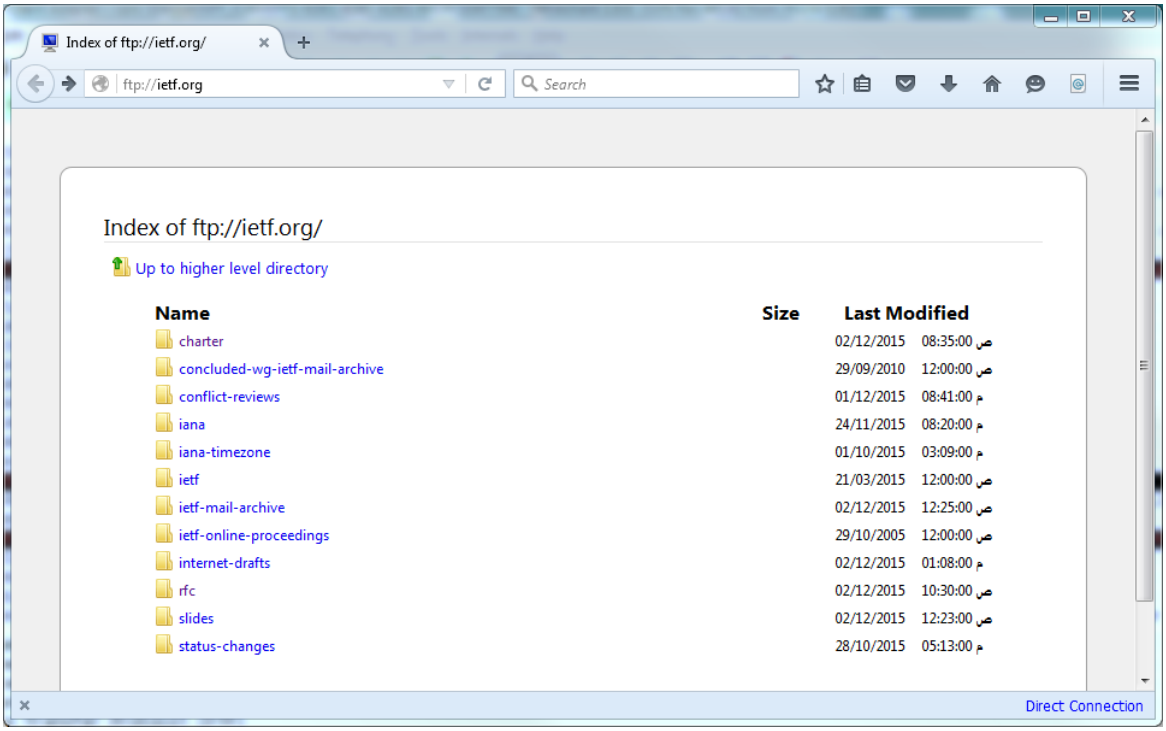
سنحاول الاتصال بمخدم FTP التالي: ftp://ietf.org عن طريق متصفح Mozilla firefox

وسطر الأوامر cmd. ستظهر لك شاشة شبيهة بالشاشة التالية:




الشكل 24– تحليل رسائل FTP

أما عند فتح موقع IETF فتظهر لك الشاشة التالية:




الشكل 25– موقع ftp://ietf.org

الخطوات المتبعة:

1. شغل متصفح Firefox أو أي متصفح آخر
2. شغل wireshark
3. أدخل العنوان URL: ftp://ietf.org
4. أضغط على المجلد Charter
5. أضغط على أول ملف نصي من الشاشة
6. أوقف تحليل الطرود

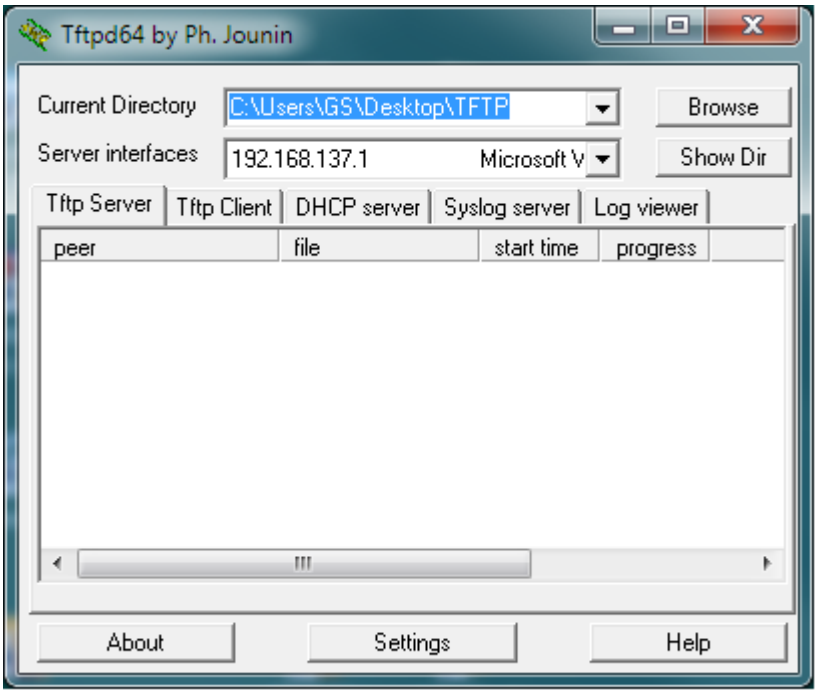
أجب على الأسئلة التالية:

1. ما هو رقم بوابة المصدر والوجهة لأول رسالة FTP من المخدم إلى الزبون؟
2. ما هو عنوان IP لمخدم FTP؟
3. ما هو اسم المستخدم وكلمة المرور التي أرسلها الزبون (أي المتصفح)؟
4. ما هي البوابة التي اختارها المخدم بعد استقبال أمر PASV؟

## 4. تمارين عن TFTP:

يلزمنا لتحقيق هذا التمرين برنامج TFTP Server وTFTP Client وWireshark لتحليل الطرود المتبادلة.

يمكنك تحميل مخدم TFTP التالي: tftpd32 or tftpd64 حسب إصدار نظام التشغيل لديك من الموقع http://tftpd32.jounin.net ومن ثم تشغيله على حاسبك الشخصي فتحصل على النافذة التالية:




الشكل 26– مخدم TFTP

يمكنك هنا تحديد مجلد المشاركة: Current Directory والعنوان الذي ستستقبل عليه الطلبات

اخترت هنا عنوان 192.168.137.1 لبطاقة الشبكة اللاسلكية:

```
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Wireless Network Connection 2:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Virtual WiFi Miniport Adapter
   Physical Address. . . . . . . . . : 68-94-23-BE-CF-CA
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::b890:4edc:a097:30e1%22(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.137.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 661165091
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-18-90-17-A4-84-34-97-76-EF-74
```

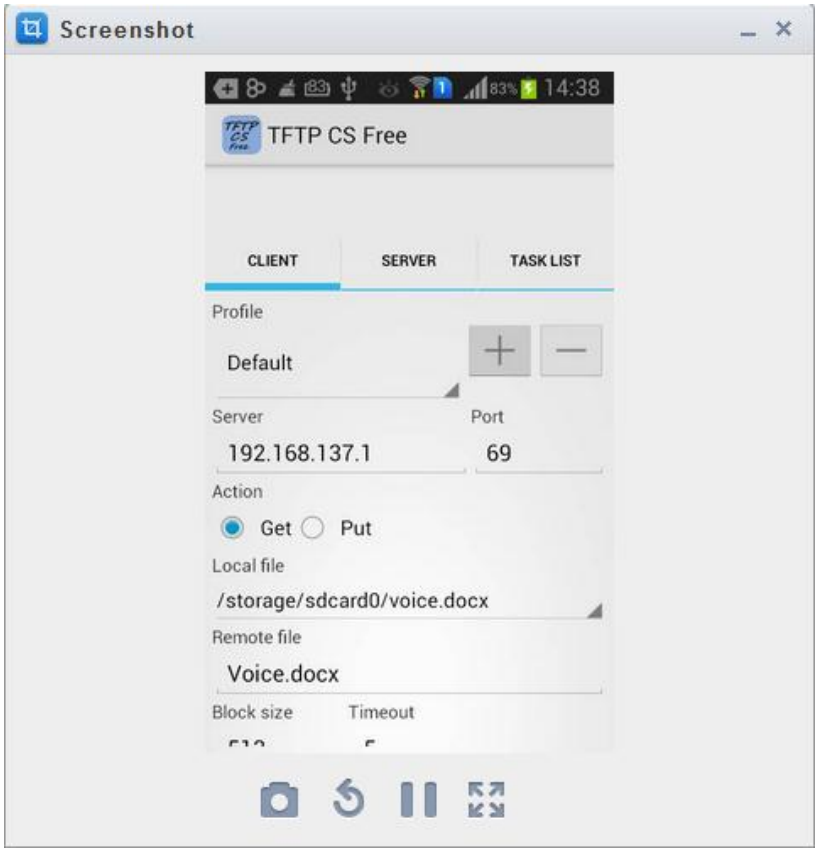
الشكل 27– تعريف البطاقة اللاسلكية

يجب الآن تحميل برنامج TFTP Client إما على حاسب آخر أو على نظام Android. اخترت برنامج TFTPCS الموجود ضمن أسواق الآندرويد.

أنشئت بعد ذلك مجلد على سطح المكتب تحت اسم TFTP ووضعت فيه ملفين doc1.txt وVoice.docx.

قمت بتشغيل المخدم والزبون wireshark.

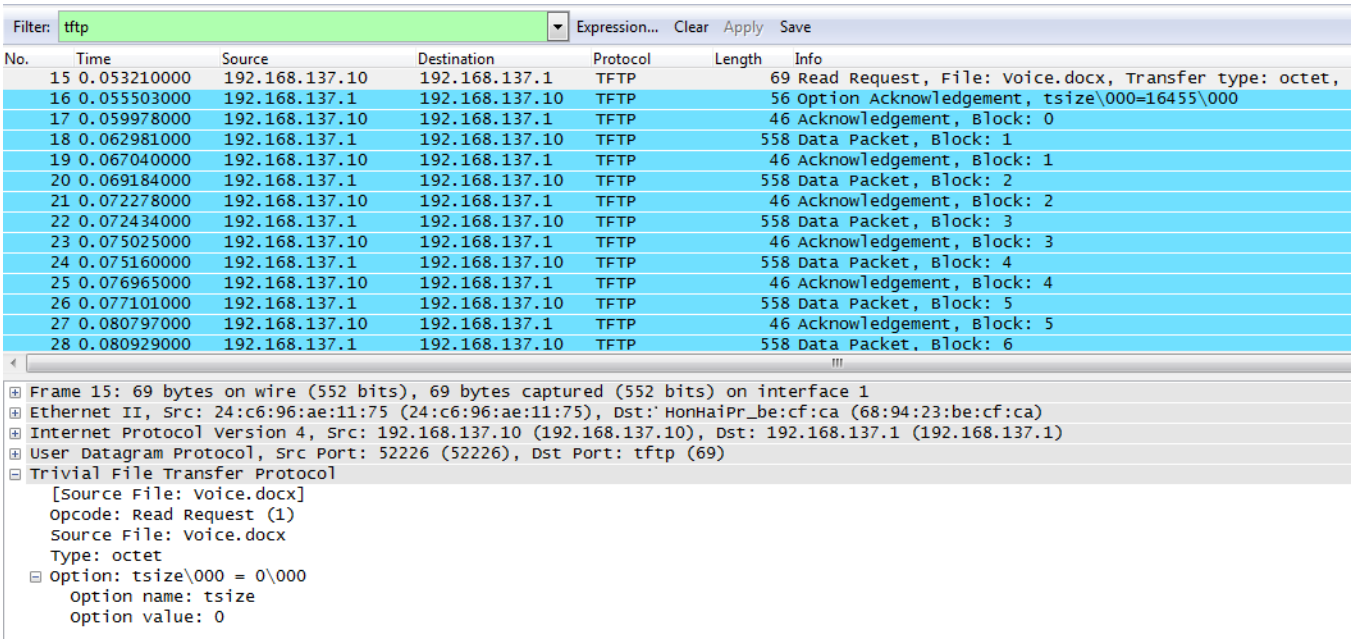
بعد تشغيل زبون TFTP على جهاز الموبايل حصلت على النافذة التالية:




الشكل 28– واجهة TFTP Client on Android

لاحظ أنني أدخلت عنوان مخدم TFTP وأسم الملف البعيد والمحلي ثم بدأت عملية التحميل "Get".

بالنسبة لمحلل الطرود فحصلت على الواجهة التالية:




الشكل 29– واجهة تحليل طرود TFTP

المطلوب من الطالب:

1. إعداد مخدم TFTP على جهاز الحاسب
2. إعداد زبون TFTP على جهاز حاسب لآخر أو على جهاز Android
3. تشغيل Wireshark على الجهاز المخدم

### 1.4. حالة تنزيل ملف (قراءة) من المخدم:

أجب عن الأسئلة التالية:

1. ما هي عناوين IP لكل من المخدم والزبون؟
2. ما هو البروتوكول الذي يستخدمه TFTP؟ TCP or UDP؟
3. ما هو رقم بوابة الوجهة لأمر Read Request الموجه من الزبون إلى المخدم؟
4. ما هو رقم بوابة المصدر لأول كتلة معطيات (أي Block 1) من المخدم إلى الزبون؟
5. ما عدد البايتات التي تحملها الكتلة رقم 1 Payload only؟
6. ما عدد البابتات التي تحملها الكتلة الأخيرة؟

# تطبيقات الند للند: بت تورنت وسكايب
# P2P Applications: BitTorrent & Skype

ISSN: 2617-989X

تعتمد التطبيقات الشبكية التي درسناها سابقاً مثل الوب والبريد الالكتروني ومخدمات الأسماء ومخدمات نقل الملفات على بنية من نوع زبون–مخدم أو بشكل آخر تعتمد اعتماداً كبيراً على مخدم ذو إمكانيات كبيرة في البنية التحتية للتطبيقات. أما في حالة استخدام بنية من نوع P2P فإن الاعتماد على مخدمات مخصصة ومراكز معلومات لاستضافتها يصبح ضئيلاً جداً إذا لم يكن معدوماً. تعتمد هنا التطبيقات على الاتصال المباشر بين زوج من المضيفين، يدعى كل منهما "ند"، دون أن تحتاج هذه العقدة للعمل بشكل دائم Always-on مثلها مثل المخدمات. تستضيف عادةً حواسيب مكتبية أو محمولة موجودة في المنازل أو الجامعات هذه التطبيقات.

إن أهم التطبيقات الشائعة وكثيرة الاستخدام والتي تولد حركات مرور هائلة على الإنترنت هي من نوع P2P. نذكر، من أهم هذه التطبيقات، تطبيق "بت تورنت BitTorrent لمشاركة الملفات، وتطبيق Xunlei لتسريع تحميل الملفات، وتطبيق سكايب للهتف عن طريق الإنترنت إضافة إلى تطبيقات IPTV مثل Kankan وPPstream.

تجدر الإشارة هنا إلى أن بعض تطبيقات P2P تكون من النوع الهجين، أي تعتمد على البنية زبون–مخدم لتحقيق وظيفة ما مثل متابعة تواجد المستثمرين وتسجيل عناوين IP المخصصة لهم بينما يكون التواصل بين المستثمرين من نوع P2P مثل بعض تطبيقات الرسائل الفورية Instant Messaging.

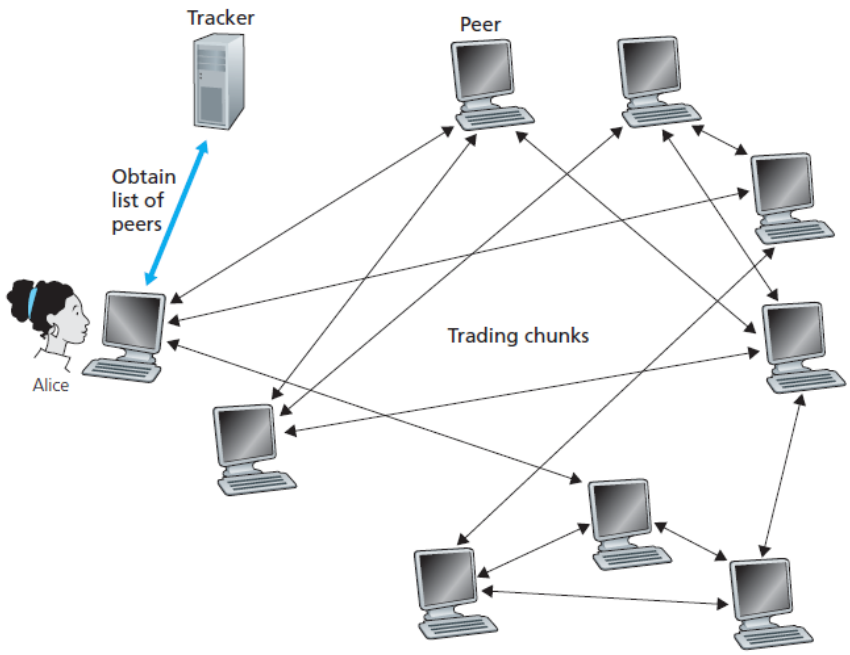
سندرس ضمن هذا الفصل تطبيقين شائعي الاستخدام وهما بت تورنت للمشاركة في الملفات وسكايب للدردشة والهاتف عن طريق الإنترنت.

## 1. تطبيق BitTorrent

حسب مصطلحات بت تورنت فإنه يطلق على مجموعة العقد التي تتشارك في توزيع ملف ما اسم "تورنت". تحمل كل عقدة (أو ند)، مشاركة ضمن تورنت، أجزاء Chunks متساوية الطول من الملف. يكون عادة طول الجزء 256 Kbytes. لا تملك العقدة عندما تنضم إلى التورنت أي جزء من الملف لكن مع مرور الوقت تبدأ بمراكمة الأجزاء لديها وبمشاركة هذه الأجزاء مع غيرها. يمكن أيضاً لكل عقدة أن تغادر التورنت في أي لحظة حتى قبل انتهاء تنزيل الملف المطلوب والعودة لاحقاً لمتابعة التحميل.

يمتلك كل تورنت عقدة ثابتة تدعى المتعقب Tracker. عندما تنضم عقدة إلى التورنت فإنها تسجل نفسها عند المتعقب وتعلم المتعقب بشكل دوري عن بقائها ضمن التورنت. يمكن أن يتراوح عدد العقد المشاركة ضمن التورنت الواحد بين العشرة والعدة آلاف حسب نوع الملف وانتشاره.

عندما تريد عقدة ما، لتكن Alice، الانضمام إلى التورنت فإن المتعقب يختار مجموعة عقد عشوائية (عادةً 50 عقدة) من العقد المشاركة في التورنت في اللحظة نفسها ويرسل عناوين IP إلى Alice كما هو موضح في الشكل التالي:




**الشكل 1 – توزيع الملفات باستخدام BitTorrent**

تحاول Alice الآن فتح ارتباطات TCP مع جميع هذه العقد. لندعو مجموعة العقد التي نجحت Alice في فتح ارتباط معها بالجيران Neighboring peers. يبين الشكل السابق 3 جيران للعقدة Alice. مع مرور الزمن، تغادر بعض العقد ولا تعود جارة للعقدة Alice كما يمكن أن تنضم عقد جديدة من خارج العقد الخمسين إلى جيران Alice عن طريق محاولة فتح ارتباط معها.

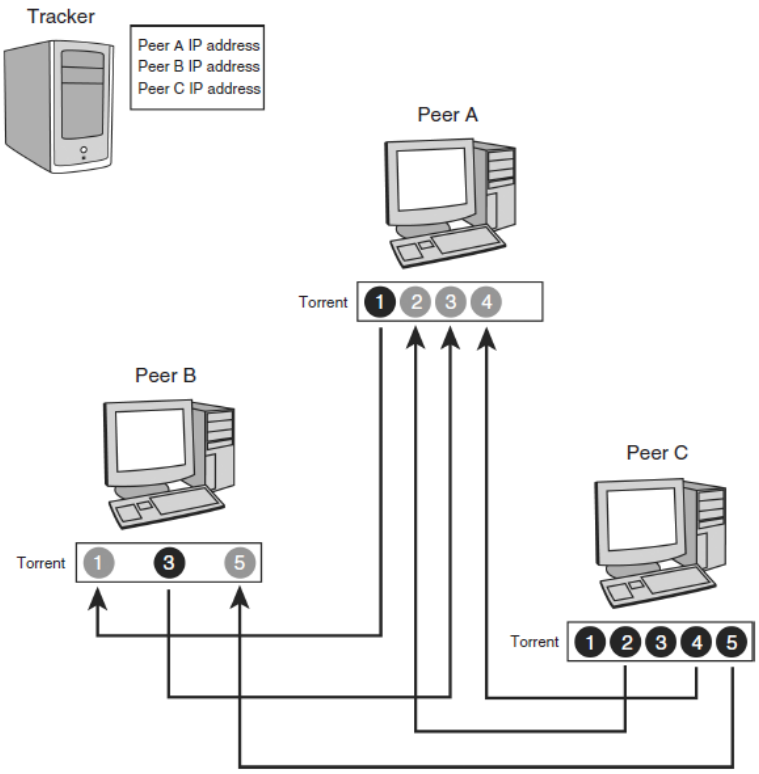
تمتلك كل عقدة في لحظة ما مجموعة جزئية من الأجزاء Chunks المكونة للملف، وكل عقدة تمتلك مجموعة مختلفة عن المجموعة التي تمتلكها غيرها. تسأل Alice جيرانها دورياً عن الأجزاء التي تمتلكها. بهذه الطريقة، ستقوم Alice بطلب الأجزاء التي لا تمتلكها من الملف من جيرانها. ينبغي هنا على Alice اتخاذ قرارين أساسين للحصول على ما ينقصها من الأجزاء: (1) ما هي الأجزاء التي يجب طلبها أولاً؟ و(2) ممن يجب طلب هذه الأجزاء؟. تعتمد Alice هنا على طريقة الأندر أولاً rarest first. أي تطلب Alice الأجزاء النادرة أو الموجودة عند أقل عدد من الجيران أولاً الأمر الذي يسمح بزيادة عدد النسخ من الأجزاء النادرة وتحقيق نوع من المساواة بعدد النسخ من كل جزء من الملف بين العقد.

أما فيما يتعلق بالطلبات التي تستجيب لها Alice فإن بت تورنت يستخدم الخوارزمية التالية: تعطي Alice الأولوية للعقد التي تزودها بالأجزاء بأسرع معدل لنقل المعطيات. أي أن Alice تقيس باستمرار معدل نقل المعطيات التي تستقبلها من كل جار من جيرانها وتحدد العقد الأربع الأسرع. تقوم Alice عندئذ بالمعاملة بالمثل، أي تجاوب فقط هذه العقد الأربعة. تعيد Alice حساب معدلات النقل كل 10 ثوانٍ وتعدل على أساس النتيجة ترتيب العقد الأربعة الأسرع. يطلق بت تورنت اسم Unchoked peers على العقد الأربعة هذه. أضف إلى ذلك أن Alice تختار كل 30 ثانية عقدة عشوائية من الجيران وترسل لها الأجزاء. لندعو هذه العقدة Bob وهي تعرف باسم Optimistically unchoked peer. بما أن Alice ترسل لـ Bob الأجزاء فإنه من الممكن أن تصبح Alice من بين أفضل أربع مزودين لـ Bob أي بما معناه أن Bob سيرسل الأجزاء إلى Alice. إذا كان معدل نقل معطيات Bob مرتفع فيمكن أن يصبح Bob من بين أفضل أربع عقد مزودة لـ Alice. يكمن الهدف من هذه الطريقة في أن لا تبقى العقد الأربعة مغلقة إلى اللانهاية وأنه يجب تجريب بقية العقد، عقدة في كل جولة، لعل إحدى هذه العقد تصبح أسرع في تزويد الأجزاء. أما بالنسبة لبقية جيران Alice خارج مجموعة 1+4 فهم Chocked أي أنهم لا يتلقون أي جزء من Alice.

يوجد حالياً الملايين من العقد التي تستخدم بت تورنت وتتشارك على مئات الآلاف من الملفات الأمر الذي أدى إلى نجاح بت تورنت وانتشاره الكبير. لعل خوارزمية تحفيز المشاركين ضمن التورنت على تزويد الأجزاء الموجودة لديهم بأعلى سرعة ممكنة حتى يصبحون قادرين على استقبال بقية الأجزاء (تعرف هذه الخوارزمية باسم Tit-for-tat) هي سبب انتشار بت تورنت وعدم تحول المستثمرين إلى "ركاب بالمجان Freeriders".

### 1.1. مثال

يبين الشكل التالي آلية تحميل ورفع ملف بين مجموعة من العقد. أسماء العقد هي A, B, and C. يدل المستطيل تحت كل عقدة على عدد الأجزاء المكونة للملف (وهي 5) إضافة إلى الأجزاء الموجودة باللون الأسود ضمن كل عقدة والأجزاء الناقصة باللون الرمادي. هنا A تملك الجزء الأول فقط بينما C تملك جميع الأجزاء أي أنها بذرة. نجد أيضاً أن العقدة A قيد إرسال الجزء رقم 1 إلى العقدة B وقيد طلب الجزأين 2 و 4 من C والجزء 3 من B.




**الشكل 2– نقل ملف بين العقد**

### 2.1. واجهة بت تورنت

إن طريقة تحميل الملفات وفق بت تورنت بسيطة جداً. يضغط المستثمر على رابطة Hyperlink للملف الذي يرغب بتحميله ومن ثم يظهر له صندوق حوار Save-as المعروف ثم تبدأ عملية التحميل ويظهر معها سرعة التحميل وسرعة الرفع Upload.

حتى نستطيع تحميل ملف وفق تطبيق بت تورنت، يجب أولاً إعداد برنامج زبون BitTorrent. لاحظ أن شبكة بت تورنت لا تزود إمكانيات البحث عن الملفات حيث يجب العودة إلى محركات البحث التقليدية مثل Google للبحث عن ملفات باللاحقة torrent.

### 3.1. نشر المحتوى

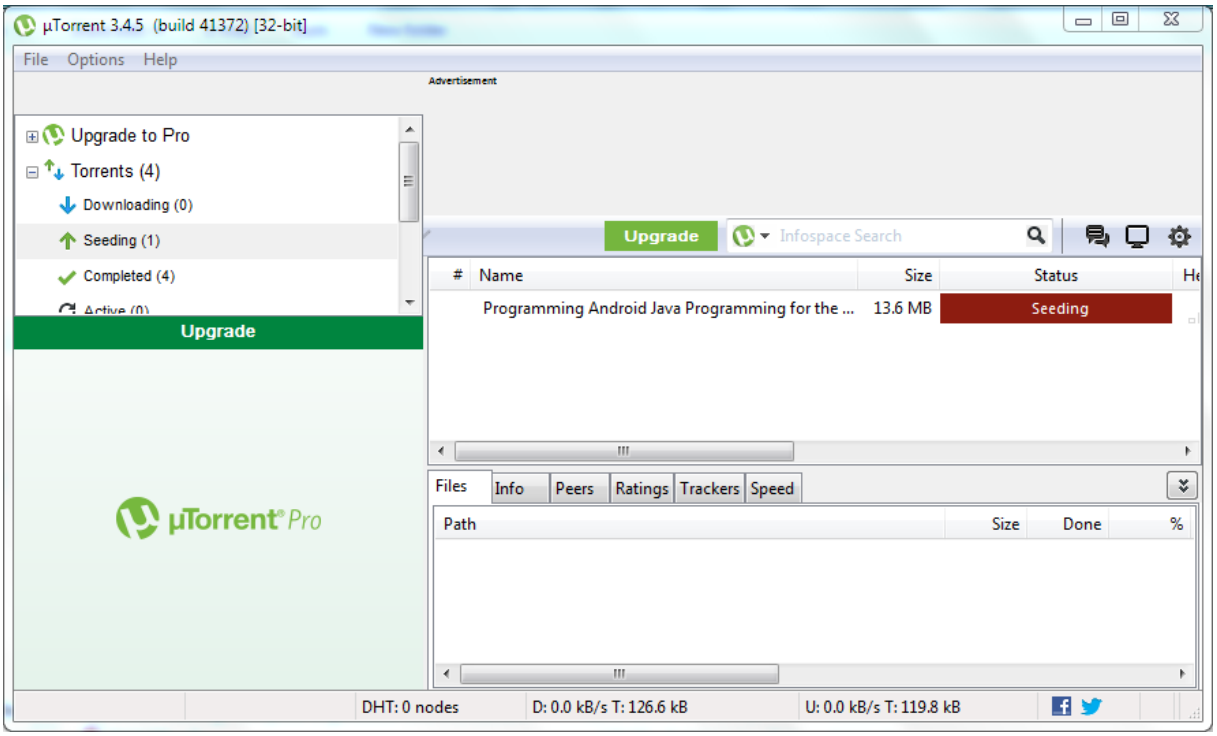
حتى نستطيع نشر ملف، يجب أولاً توليد ملف ساكن بنهاية من نوع torrent. ووضعه على مخدم وب. يحوي ملف torrent. معلومات عن الملف وطوله واسمه ومعلومات خاصة بالتهشير SHA-1 لجميع الأجزاء ورابط إلى المتعقب URL of the tracker. يساعد المتعقب محملي الملفات downloaders على إيجاد بعضهم البعض. تجري هذه المساعدة باستخدام تطبيق بسيط يعمل فوق HTTP، حيث يرسل محمل الملفات معلومات مثل اسم الملف الذي يحمله ورقم البوابة التي يتنصت عليها وغيرها من المعلومات المفيدة

بينما يجيب المتعقب بعناوين العقد التي تحمل الملف نفسه. يستطيع الآن محملي الملفات الاتصال ببعضهم. حتى يصبح الملف متاحاً، يجب على أحد العقد المحملة التي تمتلك كامل الملف (تعرف بالبذرة Seed) أن تبدأ برفع نسخة كاملة على الأقل من الملف.

ملاحظة: يمكن المشاركة على مجلد يحوي مجموعة ملفات لكن لا يمكن تغيير محتوى المجلد لاحقاً لأن الهاش الذي يولد للمجلد لا يمكن تغييره.

## 4.1. تمارين

1. تحميل وتنصيب برنامج الزبون (سنعمل مع البرنامج uTorrent 3.4.5). بعد التنصيب ستشاهد نافذة شبيهة بالشكل التالي:




**الشكل 3 – نافذة برنامج الزبون utorrent**

2. البحث عن ملف torrent. الخاص بملف ما. أدخل على google وابحث عن العبارة التالية:

   programming android ebook filetype:torrent

3. اختر الموقع torrentcd.pw

4. أضغط على download torrent ثم open with µTorrent

5. أضغط OK عند طلب موقع الحفظ

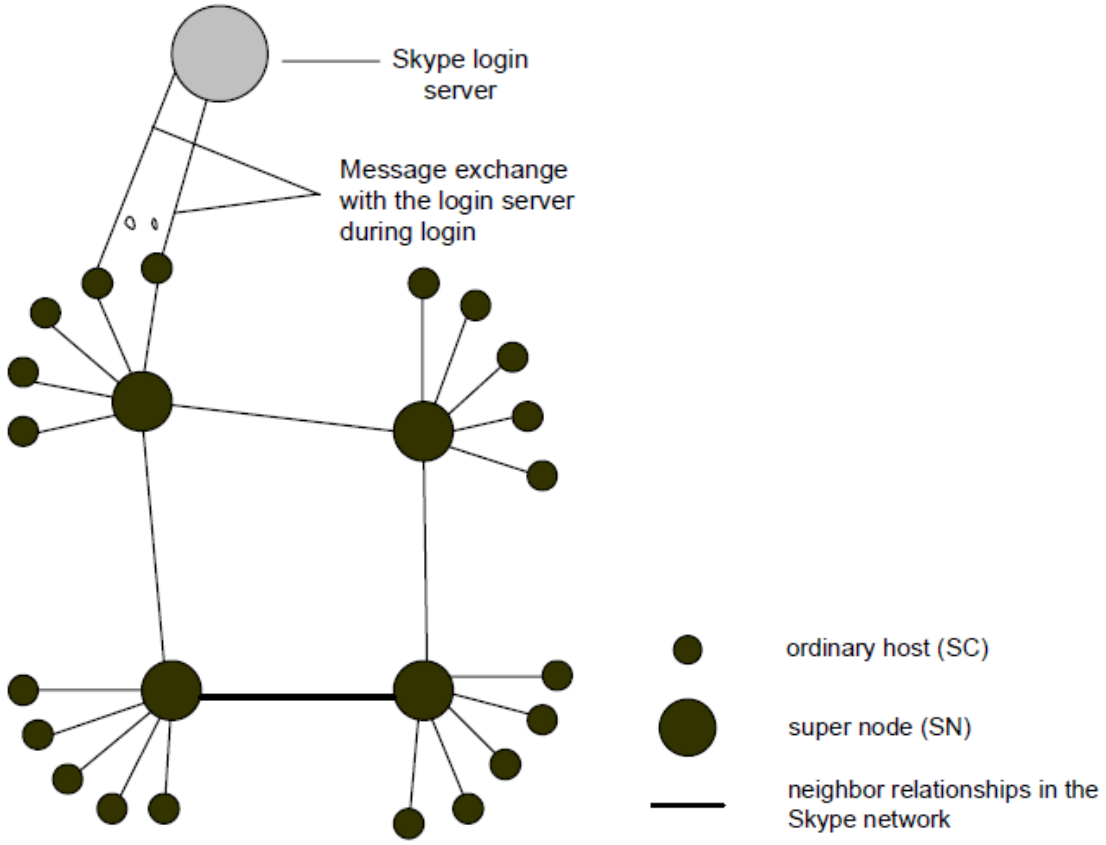
المطلوب:

البحث عن الكتاب التالي: Home Networking for Dummies 3rd 4th Ed E-Book ومن ثم البدء بتحميله والإجابة على الأسئلة التالية:

1. ماعدد الملفات التي يحويها هذا التورنت؟
2. ما هو طول الملف الكلي؟
3. ما عدد الأجزاء المكونة للملف الكلي وما طول كل جزء؟
4. ما نوع البروتوكول المستخدم TCP or UDP؟ وما هو رقم Destination port التي يطلبها الزبون؟

## 2. تطبيق سكايب

سكايب هو تطبيق لنقل الصوت عبر الإنترنت VOIP من نوع P2P. يسمح سكايب بالاتصال عبر الإنترنت وبالدردشة الفورية إضافةً إلى المكالمات المرئية Video Conferencing. يستخدم سكايب نوعين من العقد: العادية والمتقدمة Super node (SN). العقدة العادية هي العقدة التي تستخدم سكايب للاتصال الصوتي أو الدردشة. العقدة المتقدمة هي حاسب مضيف موجود على شبكة سكايب. يمكن لأي عقدة مزودة بعنوان IP عام Public IP address وموارد كافية من المعالجة CPU والذاكرة ومعدل نقل المعطيات أن تصبح عقدة متقدمة. يجب أن تتصل العقدة العادية بعقدة متقدمة كما يجب أن توفر معلومات الاستيقان Authentication مع مخدم النفاذ Skype Login Server الذي يخزن أسماء المستثمرين وكلمات المرور. يجري أيضاً تخزين لائحة الأصدقاء Buddy List ضمن هذا المخدم. يوضح الشكل التالي العلاقات بين العقدة العادية والعقدة المتقدمة ومخدم النفاذ.




**الشكل 4 – العلاقات بين العقد ومخدم النفاذ في سكايب**

يجري ربط شبكة سكايب مع شبكة الهاتف العادية عن طريق مخدمات SkypeIn and SkypeOut اللذين لا يعتبران من مكونات شبكة P2P الخاصة بسكايب. نلاحظ هنا أن المخدم المركزي الوحيد هو مخدم النفاذ. أما بالنسبة للمعلومات عن كون المستثمرين على الخط Online أو خارج الخط Offline فيجري تخزينها و تبادلها بطريقة غير مركزية.

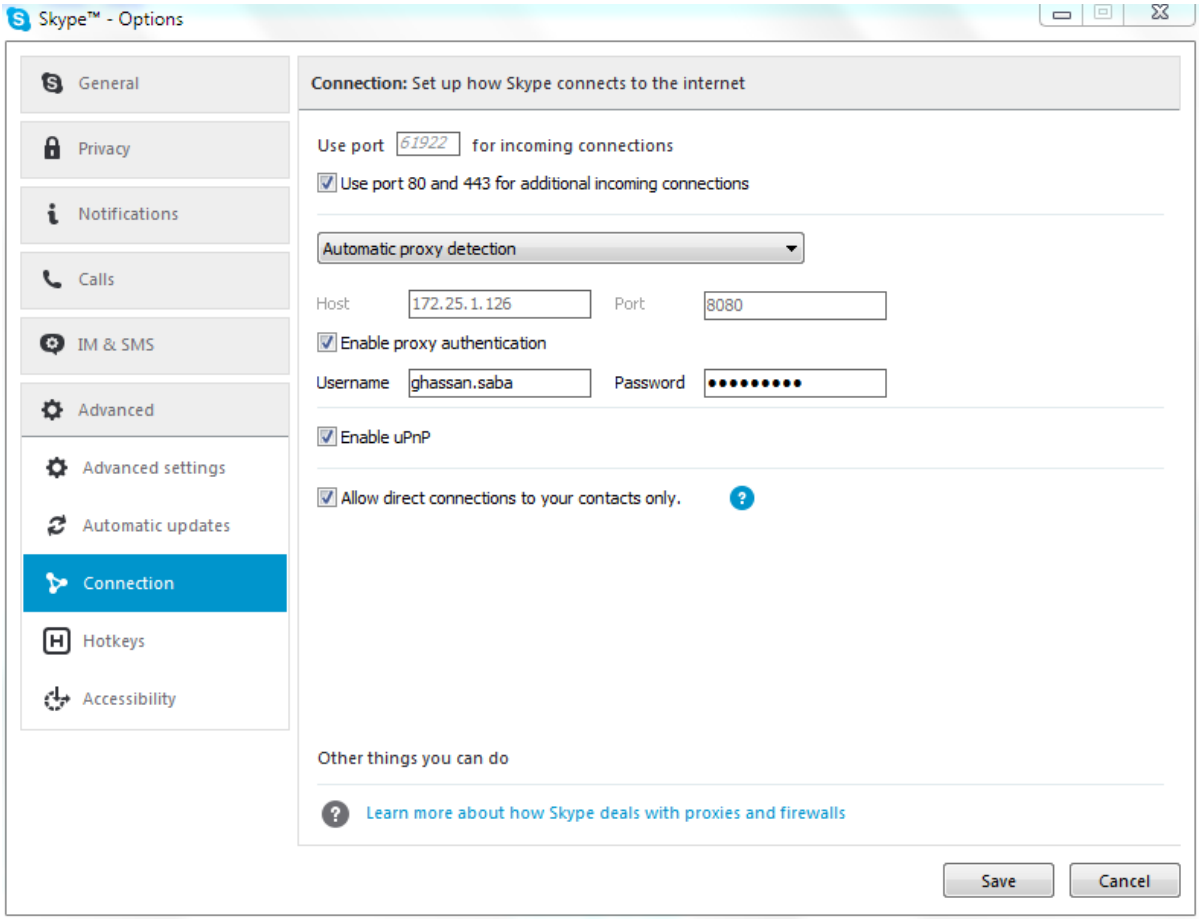
تستخدم عقد سكايب أحد أشكال بروتوكول Session Traversal Utilities for NAT (STUN) لتحديد نوع تحويل العناوين وجدار النار المستخدمين للوصول إلى الإنترنت. تعتبر شبكة سكايب من الشبكات الغطائية Overlay network حيث تحتاج كل عقدة (SC) Skype Client لبناء وتحديث جدول بالعقد التي تستطيع الوصول إليهم Reachable nodes. يدعى هذا الجدول بخابية المضيف Host Cache وهو يحوي عناوين IP وأرقام بوابات العقد المتقدمة. عادةً يجري تخزين HC ضمن ملف XML. يستخدم سكايب خوارزميات ضغط Codecs متقدمة قادرة على المحافظة على جودة عالية للمكالمات بمعدل معطيات 32 Kbps. كما يستخدم TCP للتأشير Signaling بينما يستخدم UDP و TCP لنقل حركات مرور المكالمات وغيرها.

### 1.2. المكونات الأساسية لبرمجيات سكايب

يتنصت زبون سكايب على بوابة خاصة للمكالمات الواردة ويدير على جدول ببقية عقد سكايب يعرف بخابية المضيف ويستخدم خوارزميات ضغط متقدمة ويتابع لائحة الأصدقاء ويشفر الرسائل من نهاية لنهاية كما يحدد فيما إذا كان خلف جدار نار أو NAT.

#### 1.1.2. البوابات Ports

يتنصت زبون سكايب على بوابة TCP وأخرى UDP حسب ما هو معرف ضمن إعدادات الزبون. يجري اختيار رقم البوابة عشوائياً عند التنصيب. بالإضافة إلى هاتين البوابتين، يتنصت الزبون على البوابتين HTTP port 80 و HTTPS port 443. يوضح الشكل التالي التنصت على البوابات ضمن سكايب.




**الشكل 5– البوابات التي يتنصت عليها سكايب**

### 2.1.2. خابية المضيف Host Cache (HC)

تعتبر خابية المضيف التي تحوي لائحة بعناوين وأرقام بوابات العقد المتقدمة التي تتجدد باستمرار هامة لعمل سكايب. يجب أن تحوي هذه الخابية، في النسخة v0.97 مدخل واحد على الأقل يحوي عنوان IP ورقم بوابة عقدة متقدمة على الخط. أثناء عملية النفاذ login، يحاول الزبون تأسيس ارتباط TCP مع أحد العقد الموجودة ضمن الخابية ومن ثم يتبادل معها المعلومات. إذا لم ينجح في تأسيس الارتباط فإنه يرسل تقرير فشل نفاذ. أما في النسخ الحديثة من زبون سكايب، أي v1.2 فما فوق، ففي حال عدم النجاح في تأسيس ارتباط مع جميع العقد الموجودة ضمن الخابية فإنه يحاول تأسيس ارتباط وتبادل المعلومات مع أحد عقد الإقلاع السبعة Bootstrap IP and ports المرمزة ضمن برنامج سكايب التنفيذي Hardcoded in the Skype executable. توجد الخابية في Windows 7 ضمن لمسار:

C:/Users/<username>/ AppData/Roaming/Skype/shared.xml)

بعد تشغيل سكايب ليومين متواصلين، يمكن أن يصل عدد المداخل ضمنه إلى 200.

```
<HostCache>
41C80105004105020059A9FF2F9DC20001040002DCCDF0DE
040003DCCDF0DE04000400050041050200
...
04000400050041050200
D5F0C7F6DE02
0001010002
A2A4E5DE
040003
D6E2E5DE
04000400050041050200
...
0400040005004105020 05D7B32BFF5930001040002 80CCF0DE
04000380CCF0DE04000400
</HostCache>
```

لاحظ أن القيمة باللون الأحمر إذا حولت إلى عنوان IP ورقم بوابة تصبح 213:240:199:246 والبوابة 5683.

### 3.1.2. لائحة الأصدقاء Buddy list

توجد لائحة الأصدقاء في Windwos 7 غير مشفرة ضمن المسار
C:\Users\<username>\AppData\Roaming\Skype\Skype_User_ID/config.xml

مثال: C:\Users\GS\AppData\Roaming\Skype\ghassansaba1/config.xml

كما يوجد نسخة مركزية مخزنة على أحد مخدمات سكايب. يبين الشكل التالي جزء منها:

```
<CentralStorage>
<LastBackoff>0</LastBackoff>
<LastFailure>0</LastFailure>
<LastSync>1135714076</LastSync>
<NeedSync>0</NeedSync>
<SyncSet>
<u>
<skypebuddy1>2f1b8360:2</skypebuddy1>
<skypebuddy2>d0450f12:2</skypebuddy2
```

### 4.1.2. التشفير Encryption

يستخدم سكايب خوارزمية التشفير Advanced Encryption Standard (AES) مع مفتاح تناظري بطول 256 bits لتشفير المحادثات والرسائل الفورية. كما يستخدم خوارزمية RSA مع مفتاح غير تناظري بطول 1024 bits لتبادل المفاتيح التناظرية. يجري اعتماد التوثق من المفتاح العام Public key للمستثمر من قبل مخدم نفاذ سكايب باستخدام شهادات رقمية RSA بطول 1536 bite أو 2048 bits.

## 2.2. آلية عمل سكايب

### 1.2.2. الإقلاع Startup

عند تشغيل زبون سكايب بعد التنصيب، فإنه يرسل رسالة HTTP Get للمخدم skype.com حيث يحوي السطر الأول من الرسالة الكلمة "installed".

### 2.2.2. النفاذ Login

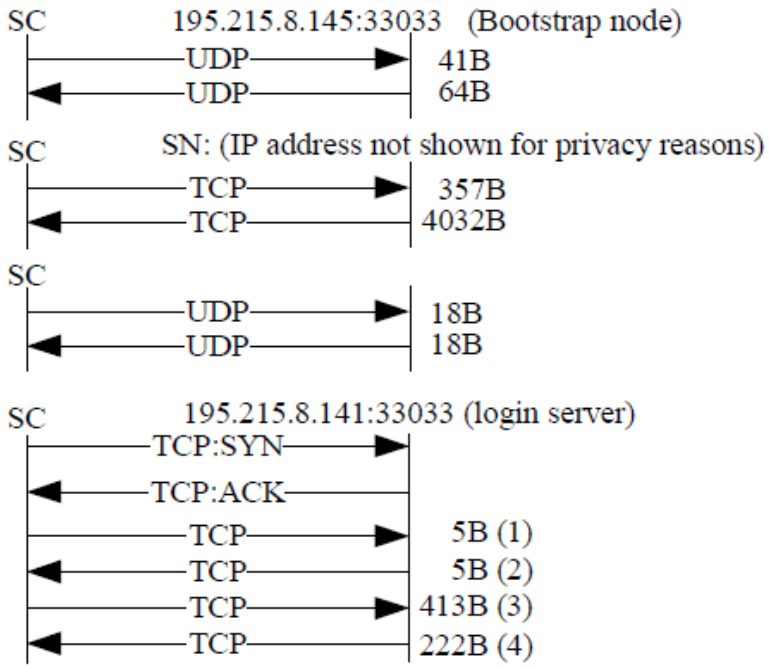
يعد النفاذ من أهم أعمال سكايب لأنه يجري خلاله الاستيقان من المستثمر، والإعلان عن تواجد المستثمر على الخط لبقية العقد والأصدقاء، وتحديد نوع NAT وجدار النار الذي يقف خلفها المستثمر، ويكتشف الأصدقاء الذين على الخط مع عناوينهم العامة ويتأكد من أن نسخة سكايب هي آخر تحديث موجود.

يبين الشكل التالي تسلسل طلبات دخول الزبون لأول مرة (أي دون استخدام الخابية).

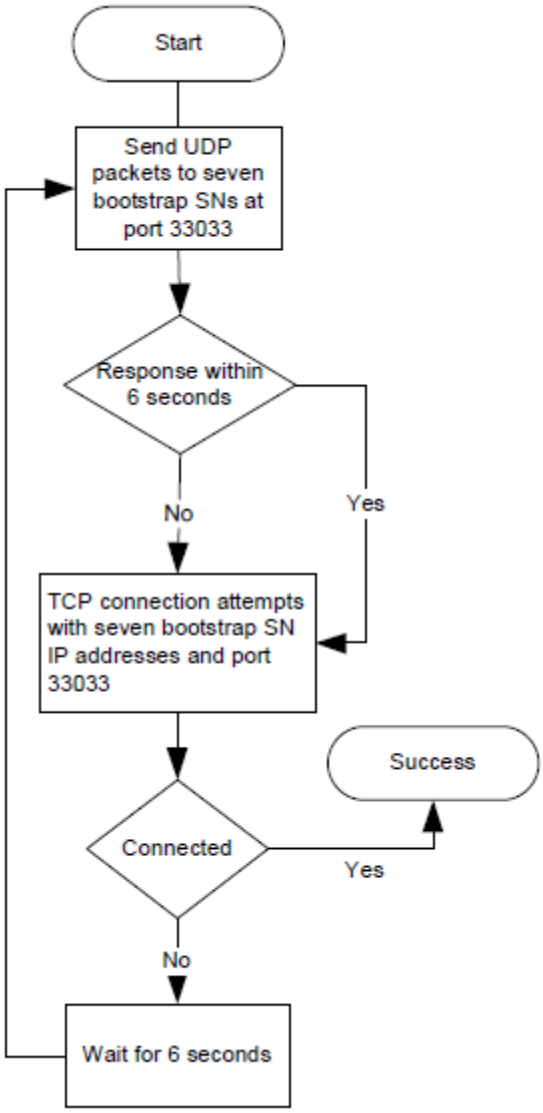
يحاول الزبون أيضاً الاتصال بمخدم سكايب عبر البوابتين 80 و 443 المخصصتين لبروتوكولي HTTP و HTTPS اللتين يفتحهما جدار النار عادةً.

يحاول الزبون أيضاً تأسيس اتصال TCP مع أحد العقد المتقدمة حتى يصبح متواجداً على شبكة سكايب وفي حالة عدم النجاح في تأسيس أي اتصال فإن سكايب يبلغ عن فشل نفاذ Login failure.

يبين الشكل التالي تسلسل عمليات النفاذ عندما تكون جميع بوابات TCP و UDP مفتوحة.

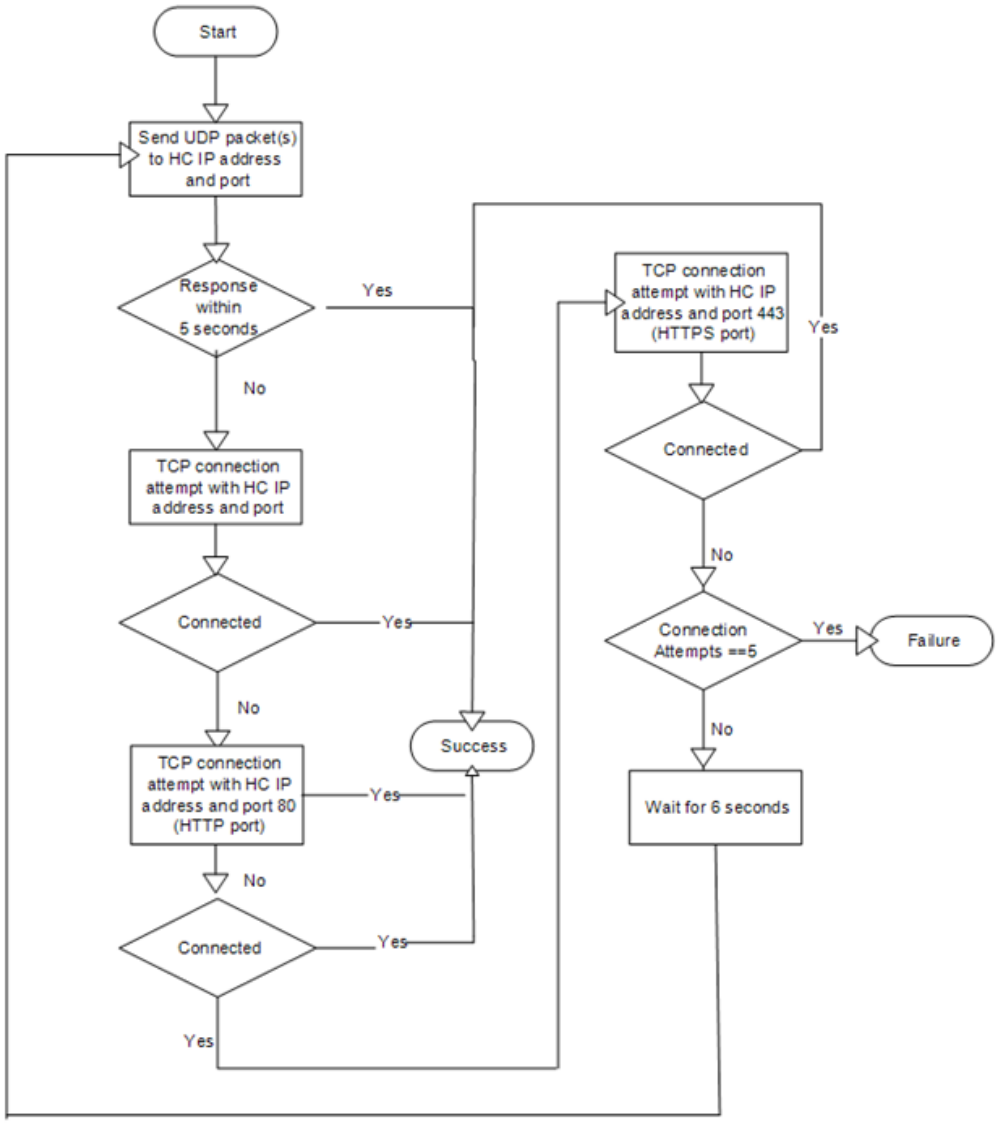



**الشكل 6 – الرسائل المتبادلة بين الزبون والعقد المتقدمة ومخدم النفاذ**




**الشكل 7 – خطوات النفاذ إلى سكايب**

أما في حال وجود معلومات ضمن الخابية وجرى تعريف الدخول عن طريق البوابات 80 و443 فيصبح النفاذ إلى سكايب كما هو موضح في الشكل التالي:




**الشكل 8– خطوات النفاذ إلى سكايب**

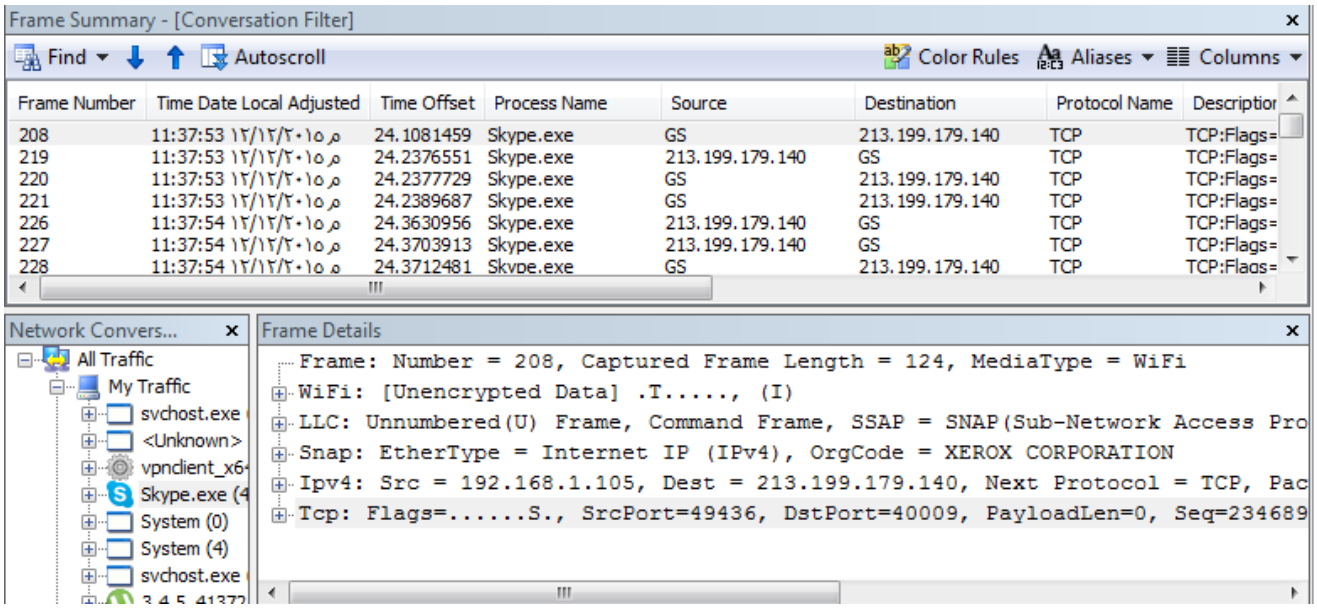
### 3.2.2. مخدم النفاذ Login server

بعد أن يتم الاتصال بين الزبون SC وعقدة متقدمة SN يقوم الزبون بالاتصال مع مخدم النفاذ لإرسال الاسم وكلمة المرور. يجري عادة الاستيقان من المستثمر باستخدام بروتوكول TCP مع عناوين IP لمخدمات سكايب.

### 3.2. مثال

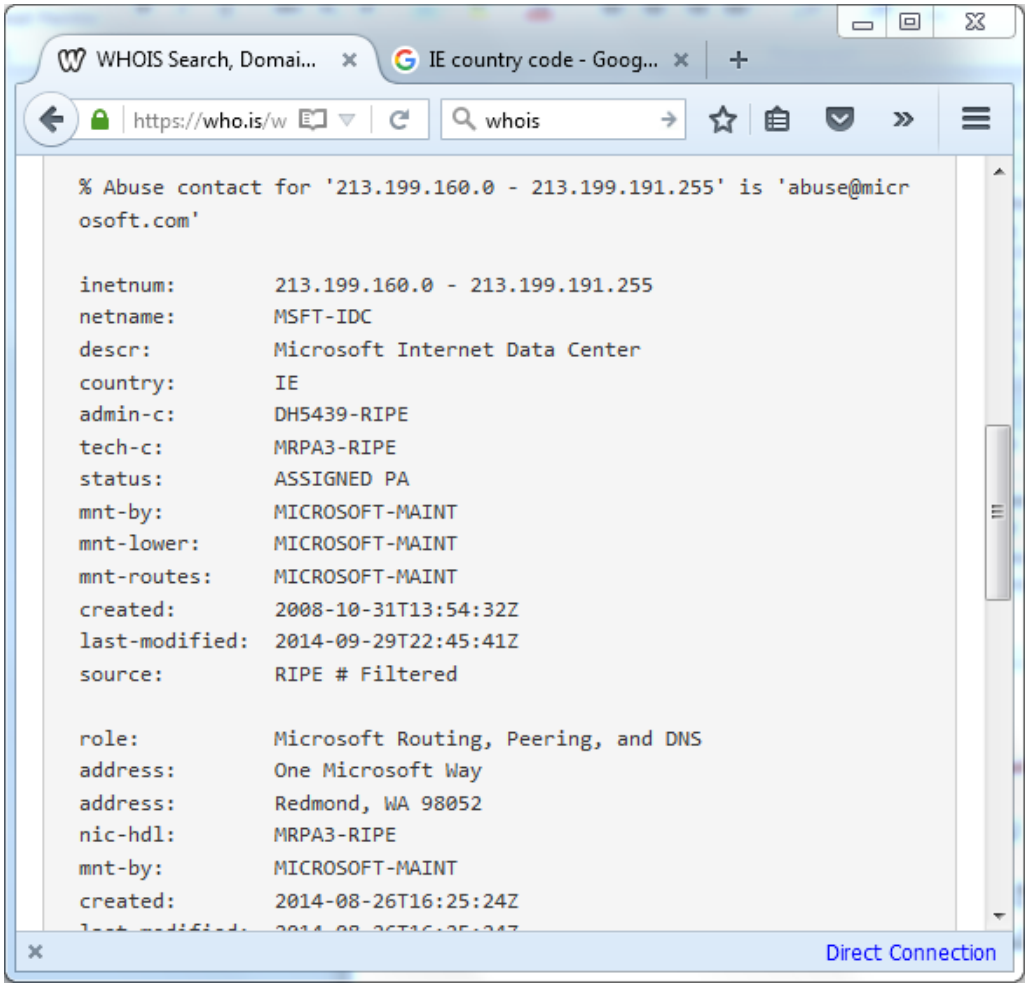
سنقوم في هذا المثال بتغيير اسم `Shared.xml` إلى `Shared_old.xml` للتعامل بدون الذاكرة الخابية ومن ثم نقوم بالخطوات التالية:

1. نخرج من سكايب `Skype logout`
2. نشغل البرنامج `Microsoft Network Monitor 3.4` لأنه قادر على تصفية الطرود حسب التطبيقات
3. نبدأ تحليل الطرود
4. ندخل إلى سكايب وفق الاسم وكلمة المرور
5. تظهر عندنا النتائج التالية:




الشكل 9– تحليل طرود سكايب باستخدام Network Monitor

6. لاحظ أن أول طرد مرسل من حاسبي `GS` إلى العنوان 213.199.179.140 هو طرد `TCP` إلى البوابة 40009 للتأكد من القدرة على التواصل مع هذه العقدة المتقدمة
7. أدخل على موقع `www.who.is` وأبحث عن مالك هذا العنوان فأجد أنه تابع لشركة مايكروسوفت في إيرلندة.




**الشكل 10– اسم وموقع مالك عنوان العقدة المتقدمة**

المطلوب:

أعد التجربة نفسها وأجب عن الأسئلة التالية:

1. ما هي البروتوكولات التي يستخدمها سكايب؟
2. ما هو أول عنوان يطلبه زبون سكايب؟
3. ما هو رقم البوابة التي يتصل إليها سكايب في أول اتصال له؟
4. أذكر أول ثلاث مواقع يتصل فيها سكايب باستخدام HTTP
5. ما هو عدد الطرود التي جرى تبادلها لفتح اتصال سكايب؟
6. حول العنوان الذي وجدته في الطلب الثاني إلى الصيغة الست عشرية ثم ابحث عن القيمة الناتجة ضمن الملف shared.xml. هل هي موجودة؟
7. حلل الطرود المتبادلة أثناء إقامة اتصال مع أحد الأصدقاء. ما هو عنوان IP للعقدة التي تخدمك أثناء إقامة الاتصال؟ ومن هو المالك؟ ومن أي دولة؟

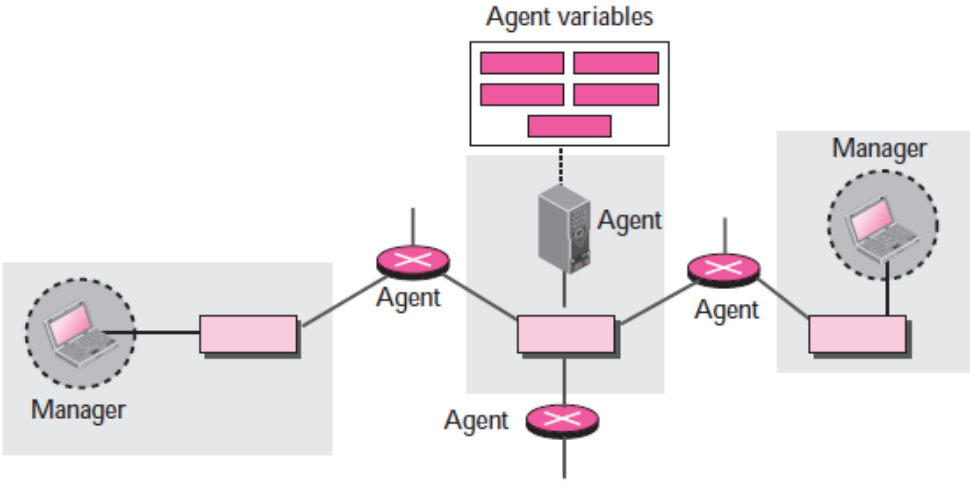
ملاحظة: تجدر الإشارة هنا إلى أن شركة مايكروسوفت اشترت شركة سكايب عام 2011 وأجرت عليه لعض التعديلات من أهمها استخدام الحوسبة السحابية Cloud Computing ضمن مراكز المعلومات Data Centers لاستضافة العقد المتقدمة SN التي أصبح عملها مركزياً وليس P2P غير أن الاتصال بين الزبائن بقي P2P.

# إدارة الشبكات وبروتوكول SNMP

بروتوكول إدارة الشبكات المبسط Simple Network Management Protocol(SNMP) هو عبارة هن منصة عمل لإدارة الأجهزة ضمن ترابط شبكات يستخدم TCP/IP. تزود هذه المنصة مجموعة من العمليات الأساسية لمراقبة وصيانة الشبكة أو مجموعة الشبكات المترابطة.

## 1. مبدأ العمل

يستخدم SNMP مبدأ المدير والعميل. يتحكم المدير، هو عادةً مضيف Host، ويراقب مجموعة من العملاء Agents، مسيرات Routers أو مبدلات Switches أو مخدمات.




**الشكل 1– مبدأ عمل SNMP**

يعمل بروتوكول SNMP على مستوى طبقة التطبيقات الامر الذي يسمح له بإدارة مجموعة متنوعة من الأجهزة مركبة على شبكات فيزيائية مختلفة.

### 1.1. المدير والعميل

تكون محطة الإدارة، أو المدير Manager، هو حاسب يشغل عليه برنامج زبون SNMP. بينما تكون المحطة المُدارة، أي العميل، مسير أو مخدم أو مبدلة يشغل عليها مخدم SNMP. يجري تحقيق الإدارة عن طريق تفاعل بسيط بين الزبون والمخدم.

يحافظ العميل على معلومات متعلقة بالأداء ضمن قاعدة معطيات. يستطيع المدير النفاذ إلى قيم هذه المعلومات الموجودة ضمن قاعدة المعطيات. فمثلاً، يخزن مسير ما عدد الطرود التي استقبلها وتلك التي أرسلها في متحولين. يقرأ المدير هذين المتحولين دورياً لمعرفة وجود اختناق Congestion عند المسير.

كما يمكن للمدير تعديل قيم المتحولات في بعض الحالات الأمر الذي يفيد في جعل المسير يعيد الإقلاع عندما يرى أن قيمة أحد المتحولات أصبحت مساوية للصفر مثلاً.
يمكن أيضاً للعميل أن يساهم في عملية المراقبة والإدارة عن طريق مراقبة قيم المتحولات لديه وفي حال تجاوزت إحدى القيم عتبة ما فإنه يرسل رسالة تنبيه للمدير (تدعى Trap).

## 2.1. مكونات نظام الإدارة

يحتاج بروتوكول SNMP لبروتوكولين آخرين من أجل عملية الإدارة: بنية معلومات الإدارة Structure of Management Information (SMI) وقاعدة معلومات الإدارة Management Information Base (MIB).

### دور SNMP

يعرف بروتوكول SNMP صيغة الطرود المتبادلة بين العميل والمخدم كما أنه يفسر النتائج. تحمل الطرود المتبادلة اسم الغرض المطلوب وقيمته أو حالته. يعتبر SNMP مسؤولاً عن قراءة هذه القيم أو تعديلها.

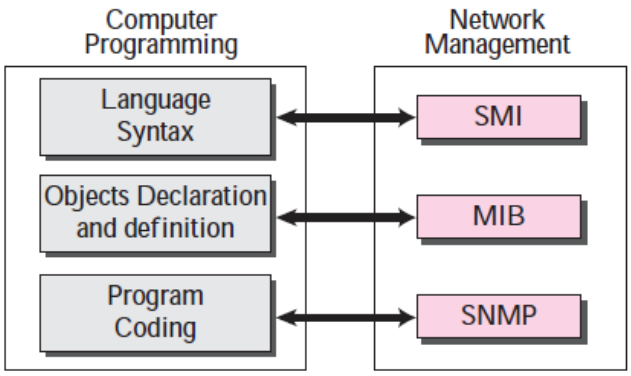
### دور SMI

حتى نستطيع استخدام SNMP نحتاج إلى تعريف قواعد لتسمية الأغراض. هذه القواعد هامة جداً لأن الأغراض تكون تراتبية Hierarchical أي لكل غرض أب وبعض الأبناء ويمكن الوراثة من الأب جزء من الاسم.
تعتبر مهمة SMI هي تعريف القواعد العامة لتسمية الأغراض وأنواعها وأطوالها إضافةً إلى إظهار طريقة ترميز الأغراض والقيم.

### دور MIB

تخلق قاعدة MIB مجموعة من الأغراض المسماة Name objects وتعرف أنماطها والعلاقات فيما بينها ضمن كينونة مدارة Entity.

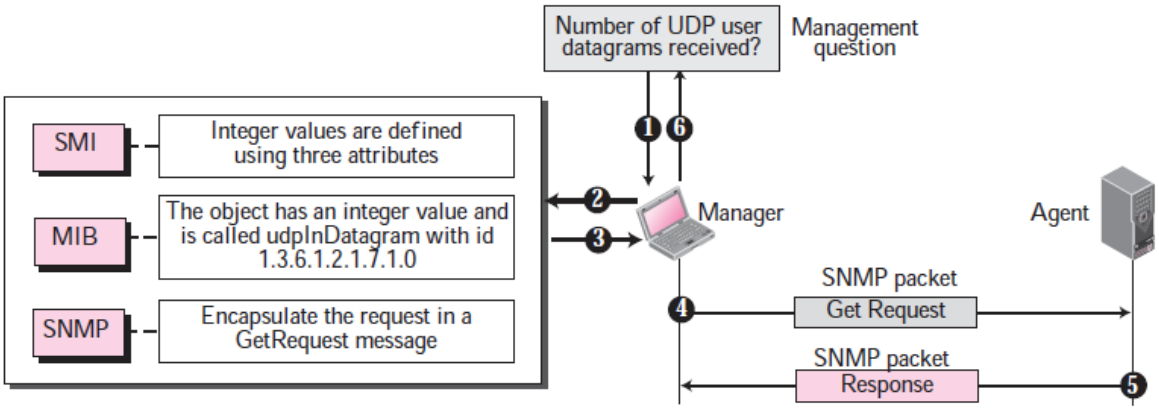
يمكننا مقارنة الأدوار المختلفة السابقة مع كتابة برنامج بلغة برمجة ما حسب ما هو موضح في الشكل التالي:




**الشكل 2– مقارنة البرمجة بإدارة الشبكات**

## لمحة عامة

لنعطي مثالً بسيطاً يوضح تفاعل هذه المكونات الثلاثة حسب السيناريو التالي. بفرض أن محطة المدير (زبون SNMP) يريد إرسال رسالة إلى محطة العميل (مخدم SNMP) لإيجاد عدد طرود UDP التي استقبلها العميل. يبين الشكل التالي تسلسل الخطوات المتبعة.




**الشكل 3– لمحة عن آلية عمل SNMP**

## SMI .3.1

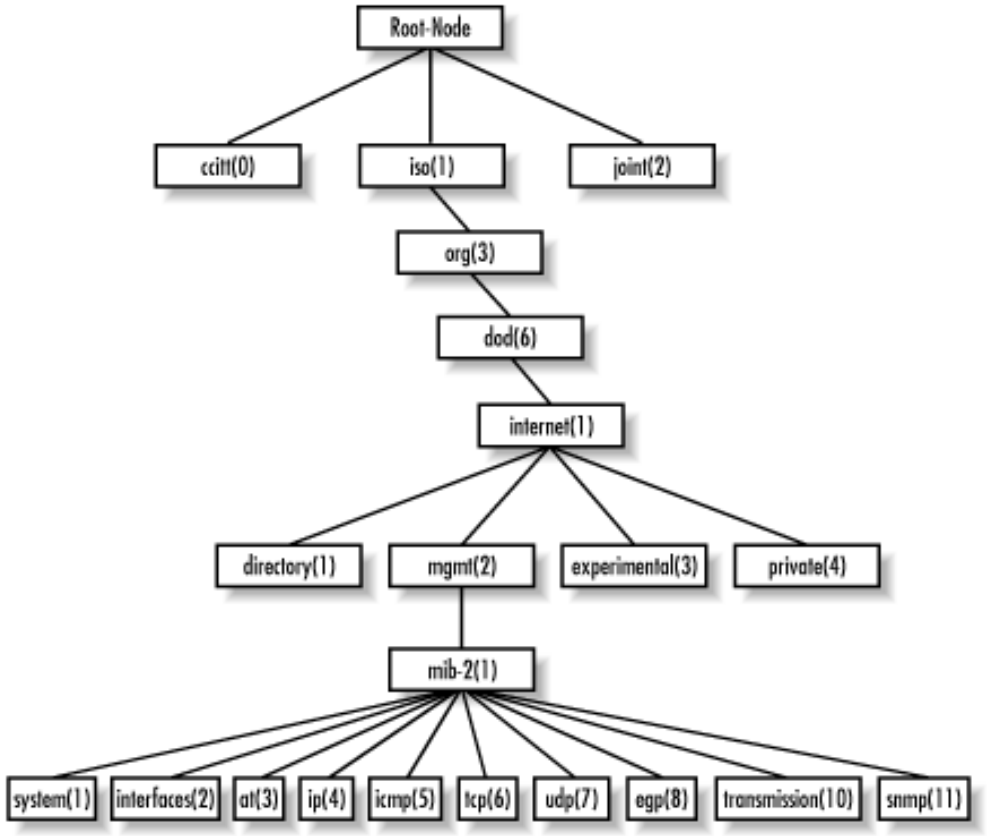
تعتبر SMI مسؤولة عن:

- تسمية الأغراض
- تعريف نمط المعطيات لتي يمكن تخزينها ضمن الأغراض
- طريقة ترميز المعطيات للنقل على الشبكة؟

فإذاً، يحدد SMI ثلاث سمات للأغراض ينبغي تعريفها: اسم Name ونمط معطيات Data type و طريقة ترميز Encoding method.

### Name الاسم .1.3.1

يجب على كل غرض قابل للإدارة (مسير أو مبدلة أو مخدم) أن يملك اسم وحيد. حتى نستطيع تسمية الأغراض تسميةً شاملة، يستخدم SMI مفهوم معرف الغرض Object identifier، الذي هو عبارة عن طريقة تراتبية للتعريف تعتمد على البنية الشجرية (شاهد الشكل التالي).




**الشكل 4 – البنية التراتبية لمعرف الغرض**

تبدأ الشجرة بالجذر الذي لا يملك اسم. يمكن تعريف كل غرض عن طريق تسلسل الأعداد الطبيعية integers المفصولة عن بعضها بالنقطة ".". كما يمكننا استخدام تسلسل الكلمات (الأسماء) المفصولة بالنقطة. عادةً، يستخدم SNMP الأرقام بينما يستخدم الأشخاص الأسماء. على سبيل المثال:

Iso.org.dod.internet.mgmt.mib-2.ip = 1.3.6.1.2.1.4

لاحظ أن الأغراض التي يتعامل معها بروتوكول SNMP موجودة تحت الغرض mib-2، فأرقامها دائماً تبدأ بالتسلسل 1.3.6.1.2.1.

### 2.3.1. النمط Type

يستخدم SMI هنا التعريفات الأساسية من Abstract Syntax Notation 1 (ASN.1) ويضيف إليها بعض التعريفات الجديدة.

يعرف SMI نوعين لأنماط المعطيات: البسيطة Simple والبنيوية Structured.

#### النمط البسيط

يبين الجدول التالي أهم هذه الأنماط مع العلم أن أول خمسة أنماط مأخوذة من ASN 1 وبقية الأنماط جديدة.

| *Type* | *Size* | *Description* |
|---|---|---|
| INTEGER | 4 bytes | An integer with a value between $-2^{31}$ and $2^{31}-1$ |
| Integer32 | 4 bytes | Same as INTEGER |
| Unsigned32 | 4 bytes | Unsigned with a value between 0 and $2^{32}-1$ |
| OCTET STRING | Variable | Byte-string up to 65,535 bytes long |
| OBJECT IDENTIFIER | Variable | An object identifier |
| IPAddress | 4 bytes | An IP address made of four integers |
| Counter32 | 4 bytes | An integer whose value can be incremented from zero to $2^{32}$; when it reaches its maximum value it wraps back to zero |
| Counter64 | 8 bytes | 64-bit counter |
| Gauge32 | 4 bytes | Same as Counter32, but when it reaches its maximum value, it does not wrap; it remains there until it is reset |
| TimeTicks | 4 bytes | A counting value that records time in 1/100ths of a second |
| BITS | | A string of bits |
| Opaque | Variable | Uninterpreted string |

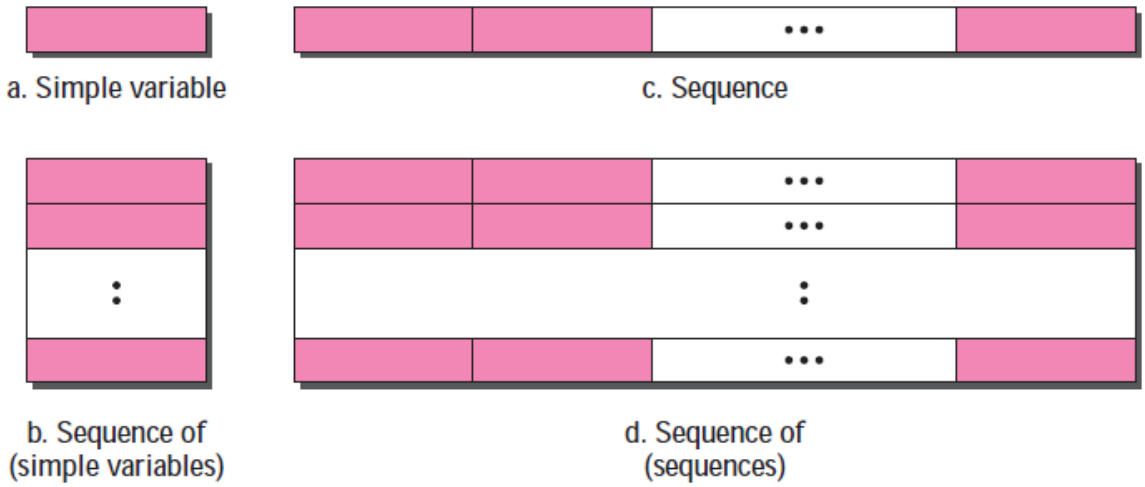
**الشكل 5 – أنماط المعطيات وفق SMI**

## الأنماط البنيوية

يعرف SMI نمطين بنيويين وهما "تسلسل وتسلسل من" Sequence and Sequence of واللذين يمكن استخدامهما مع الأنماط البسيطة لتوليد أنماط جديدة.

- التسلسل Sequence. التسلسل هو عبارة عن تركيب مجموعة من الأنماط البسيطة، ليس من الضروري من النمط نفسه. هو مفهوم قريب من نمط Struct or record المستخدمين في لغات البرمجة.
- التسلسل من Sequence of. نمط "التسلسل من" هو عبارة عن تركيب أنماط بسيطة كلها من النمط نفسه أو تركيب أنماط معطيات التسلسل كلها أيضاً من النمط نفسه. هو مفهوم قريب من نمط Array المستخدم في لغات البرمجة.

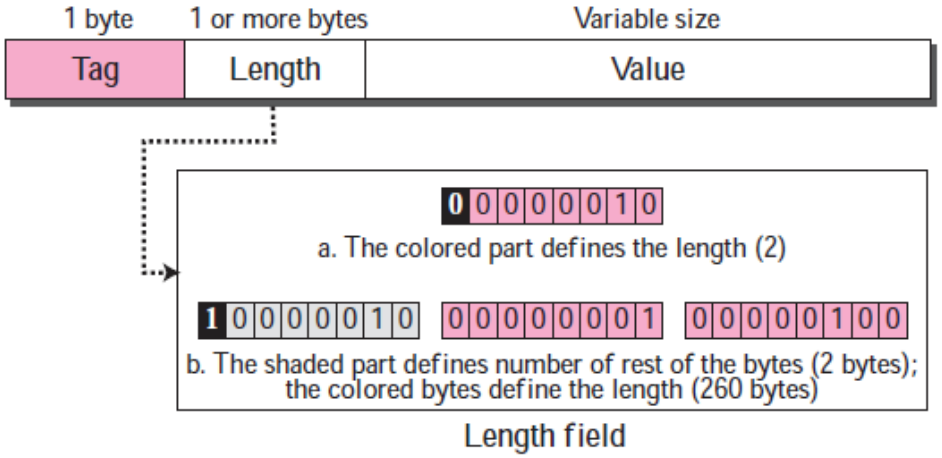
يبين الشكل التالي مفهوم الأنماط المختلفة.




**الشكل 6 – أنماط المعطيات وفق SMI**

### 3.3.1. طرائق الترميز Encoding methods

يستخدم SMI هنا معيار آخر هو Basic Encoding Rules (BER) لترميز المعطيات قبل نقلها على الشبكة. يستخدم BER أسلوب الترميز الثلاثي للمعطيات: سمة tag، وطول length، وقيمة Value.




**الشكل 7 – صيغة الترميز وفق SMI**

يعرف حقل السمة Tag نمط المعطيات وهو على طول بايت واحد. يرمز حقل الطول على بايت واحد أو أكثر فإذا كانت قيمة بت مؤشر الدلالة الأعلى هي صفراً فالترميز يكون على بايت واحد وتحدد بقية بتات هذا البايت طول المعطيات. أما إذا كانت قيمة بت مؤشر الدلالة الأعلى هي الواحد فتحدد بقية بتات البايت الأول عدد البايتات الباقية وتحدد بقية البايتات طول الحقل. الجدول التالي يعطي بعض الامثلة عن الأنماط المختلفة للمعطيات.

| Data Type | Tag (Hex) |
|---|---|
| INTERGER | 02 |
| OCTET STRING | 04 |
| OBJECT IDENTIFIER | 06 |
| NULL | 05 |
| Sequence, Sequence of | 10 |
| IPAddress | 40 |
| Counter | 41 |
| Gauge | 42 |
| TimeTicks | 43 |
| Opaque | 44 |

**الشكل 8 – أنماط المعطيات**

تمرين 1

يوضح الشكل التالي طريقة ترميز العدد الطبيعي INTEGER 14.

| 02 | 04 | 00 | 00 | 00 | 0E |
|---|---|---|---|---|---|
| 00000010 | 00000100 | 00000000 | 00000000 | 00000000 | 00001110 |
| Tag (integer) | Length (4 Bytes) | Value (14) | | | |

**الشكل 9– ترميز العدد الطبيعي 14**

تمرين 2

يوضح الشكل التالي طريقة ترميز OCTET STRING "HI"

| 04 | 02 | 48 | 49 |
|---|---|---|---|
| 00000100 | 00000010 | 01001000 | 01001001 |
| Tag (String) | Length (2 Bytes) | Value (H) | Value (I) |

تمرين 3

يوضح الشكل التالي طريقة ترميز Object Identifier 1.3.6.1 (iso.org.dod.internet).

| 06 | 04 | 01 | 03 | 06 | 01 |
|---|---|---|---|---|---|
| 00000110 | 00000100 | 00000001 | 00000011 | 00000110 | 00000001 |
| Tag (ObjectID) | Length (4 Bytes) | Value (1) | Value (3) | Value (6) | Value (1) |

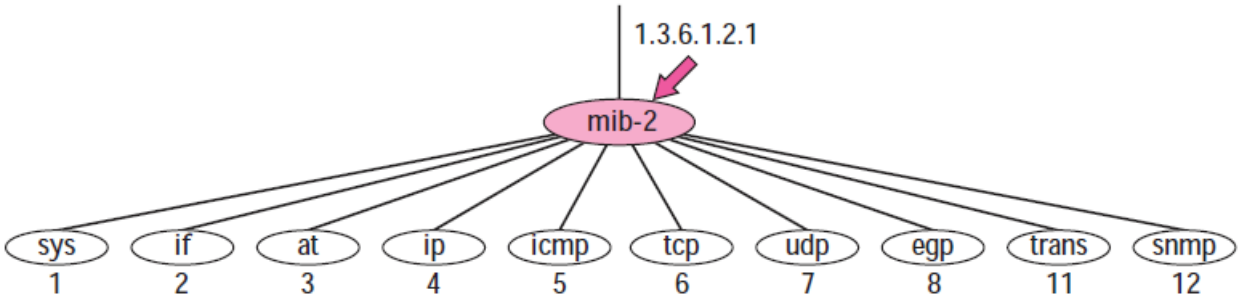
تمرين 4

يوضح الشكل التالي طريقة ترميز IPAddress 131.21.14.8.

| 40 | 04 | 83 | 15 | 0E | 08 |
|---|---|---|---|---|---|
| 01000000 | 00000100 | 10000011 | 00010101 | 00001110 | 00001000 |
| Tag (IPAddress) | Length (4 Bytes) | Value (131) | Value (21) | Value (14) | Value (8) |

### 4.1. قاعدة MIB

يملك كل عميل SNMP قاعدة معطيات MIB2 خاصة به، والتي هي مجموعة من الأغراض القابلة للإدارة. يجري تصنيف الأغراض ضمن MIB ضمن 10 مجموعات: النظام System، والواجهة Interface، وتحويل العناوين، وip، وtcp، وudp، وegp، والنقل transmission، وsnmp. تقع هذه المجموعات تحت الغرض mib-2 ضمن شجرة الأغراض. تعرف كل مجموعة عدة متحولات و/أو جداول.



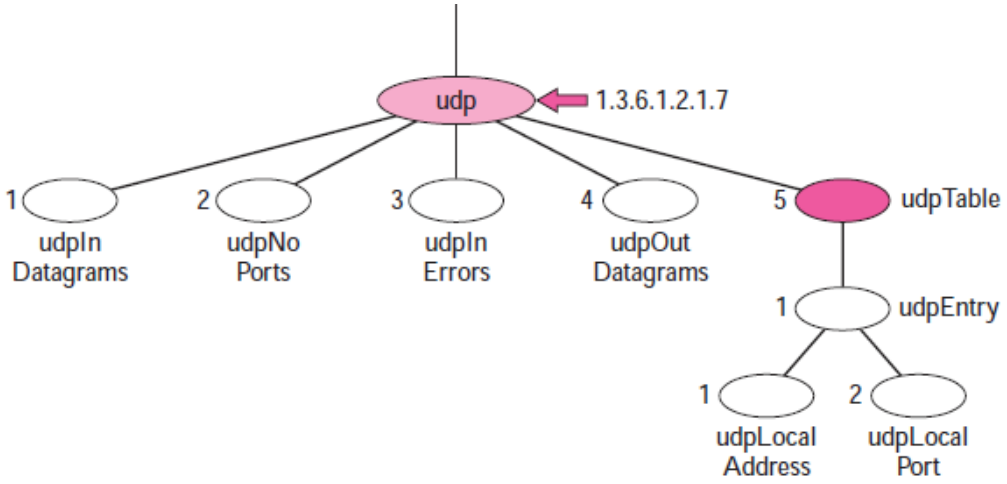

**الشكل 10– Mib-2**

نلخص فيما يلي وظيفة هذه الأغراض:

- Sys (system): يعرف معلومات عامة عن العقدة مثل اسمها وموقعها وعمرها.
- If (interface): يعطي معلومات عن جميع بطاقات الشبكة مثل رقم البطاقة والعنوان الفيزيائي وعنوان IP
- At (Address Translation): يعطي معلومات عن جدول ARP
- ip: يعطي معلومات عن Ip مثل جدول التسيير وعنوان IP
- Icmp: يعطي معلومات عن بروتوكول ICMP مثل عدد الطرود المرسلة أو المستقبلة وعدد الأخطاء الحاصلة
- Tcp: يعطي معلومات عامة عن TCP مثل جدول الارتباطات وقيم المؤقتات الزمنية وأرقام البوابات وعدد الطرود المرسلة أو المستقبلة
- Udp: يعطي معلومات عن UDP مثل عدد البوابات وعدد الطرود المرسلة أو المستقبلة
- Snmp: يعطي معلومات عن SNMP نفسه

### 1.4.1. النفاذ إلى متحولات MIB

حتى نستطيع النفاذ إلى متحولات UDP مثلاً، فإنه يوجد أربعة متحولات بسيطة ضمنه إضافةً إلى متحول من نوع "تسلسل من" السجلات Sequence of records. يوضح الشكل التالي المتحولات والجدول.




**الشكل 11– مجموعة udp**

سنناقش الآن طريقة النفاذ إلى كل متحول من المتحولات السابقة.

المتحولات البسيطة

نستخدم، للنفاذ إلى أي متحول بسيط، معرف المجموعة (1.3.6.1.2.1.7) متبوعة بمعرف المتحول المطلوب.

| Variable | ID |
|---|---|
| udpInDatagrams | 1.3.6.1.2.1.7.1 |
| udpNoPorts | 1.3.6.1.2.1.7.2 |
| udpInErrors | 1.3.6.1.2.1.7.3 |
| udpOutDatagrams | 1.3.6.1.2.1.7.4 |

غير أن معرفات الأغراض السابقة تعرف المتحولات وليس المحتوى. يجب علينا إضافة لاحقة خاصة بالمحتوى لإظهار محتوى المتحول. بالنسبة للمتحولات البسيطة فإن لاحقة المحتوى هي الصفر. أي لإظهار محتويات المتحولات السابقة نستخدم التالي.

| Variable | ID |
|---|---|
| udpInDatagrams | 1.3.6.1.2.1.7.1.0 |
| udpNoPorts | 1.3.6.1.2.1.7.2.0 |
| udpInErrors | 1.3.6.1.2.1.7.3.0 |
| udpOutDatagrams | 1.3.6.1.2.1.7.4.0 |

## الجداول

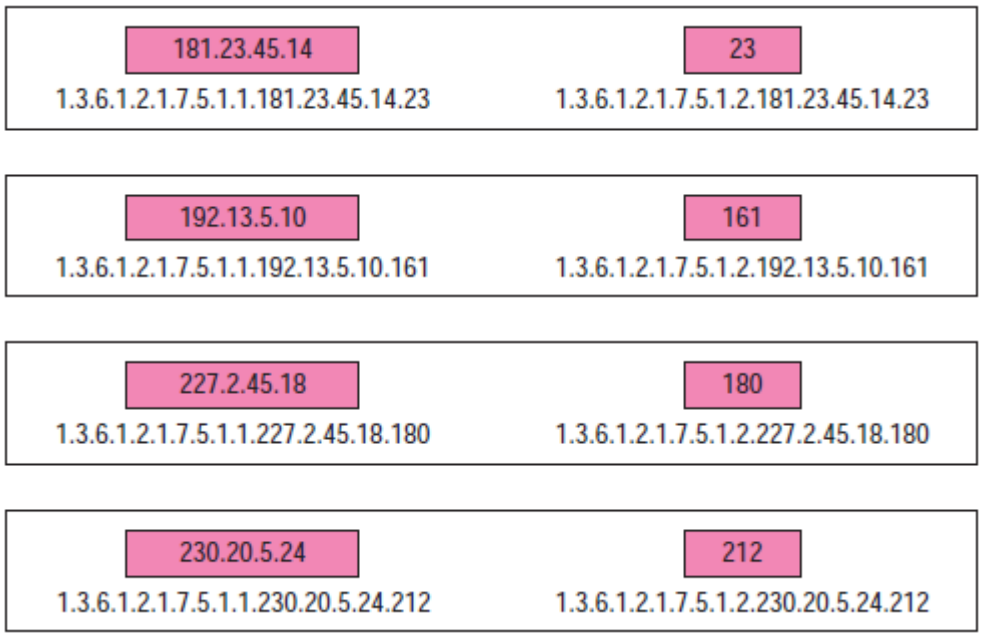
يجب أولاً معرفة معرف الجدول لنستطيع الوصول إلى جدول ما. كما لاحظنا سابقاً، تحوي مجموعة udp جدولاً واحداً (المعرف هو id=5). لذلك للنفاذ إلى هذا الجدول نستخدم المتحول 1.3.6.1.2.1.7.5. لكن الجدول ليس على مستوى الورقة النهائية ضمن التراتبية المستخدمة. فحتى نستطيع النفاذ إلى قيم الجدول، يجب علينا أولاً تعريف مدخل من نوع "تسلسل" ضمن الجدول (id=1) كما يلي:

udpEntry → 1.3.6.1.2.1.7.5.1

كما نحتاج إلى تعريف كل كينونة ضمن المدخل السابق.

udpLocalAddress → 1.3.6.1.2.1.7.5.1.1
udpLocalPort → 1.3.6.1.2.1.7.5.1.2

بما أنه في أي لحظة، يمكن أن يوجد عدة اتصالات udp مفتوحة، فحتى نستطيع النفاذ إلى اتصال محدد نضيف فهرس Index للمعرفات السابقة. يعتمد الفهرس هنا على قيمة أو بعض قيم المتحولات ضمن المدخل. في حالة udpTbale فإن الفهرس يعتمد على كل من الحقلين locaAddress وLocalPortNumber. يبين الشكل التالي جدولاً مكوناً من أربعة أسطر وقيم لكل حقل من حقوله. يكون الفهرس هو تراكب لقيمتين معاً.




**الشكل 12– UDP Table**

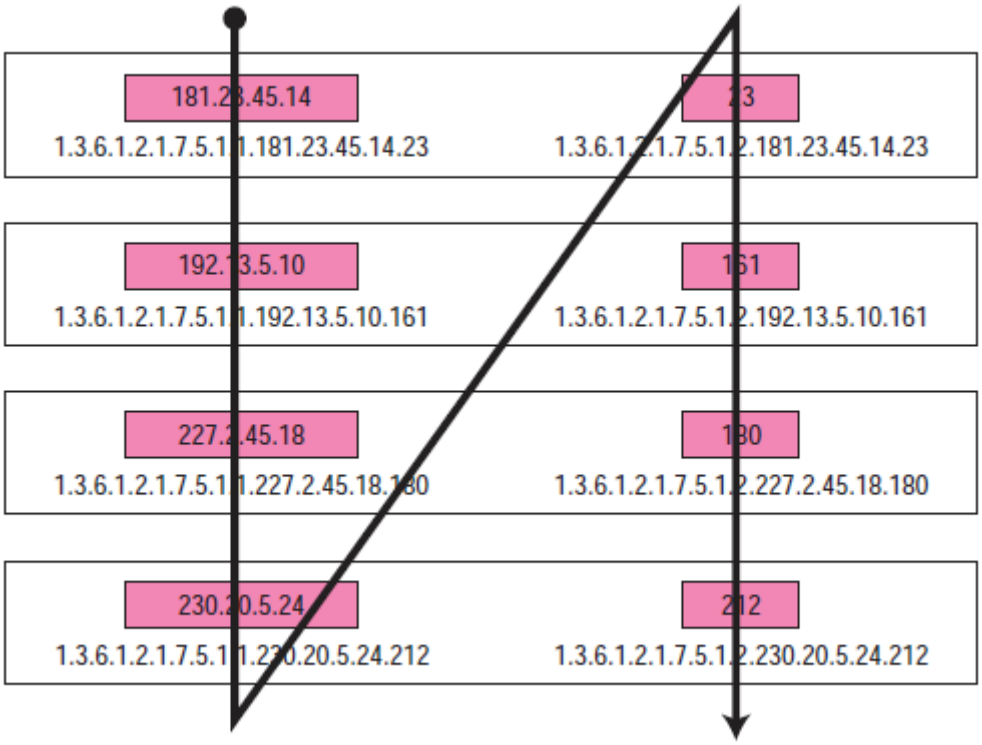
للنفاذ إلى قيمة العنوان المحلي للسطر الأول، نستخدم المعرف المضاف إليه فهرس القيمة:

udpLocalAddress.181.45.14.23 →1.3.6.1.2.7.5.1.1.181.23.45.14.2

لاحظ أنه لا تجري فهرسة جميع الجداول بالطريقة نفسها حيث يمكن استخدام قيمة حقل واحد فقط بدلاً عن أثنين.

### التسلسل المعجمي Lexicographical Ordering

تجري عملية تسلسل الجداول باستخدام قاعدة عمود–سطر، أي أنه بجب الذهاب عمود عمود. ضمن العمود، نذهب من الأعلى إلى الأدنى كما هو موضح في الشكل التالي:




**الشكل 13– التسلسل المعجمي**

## 5.1. بروتوكول SNMP

يستخدم بروتوكول SNMP كلٍ من معلومات SMI وقاعدة MIB لتحقيق إدارة الشبكات. هو تطبيق يعمل على مستوى طبقة التطبيقات يمكن:

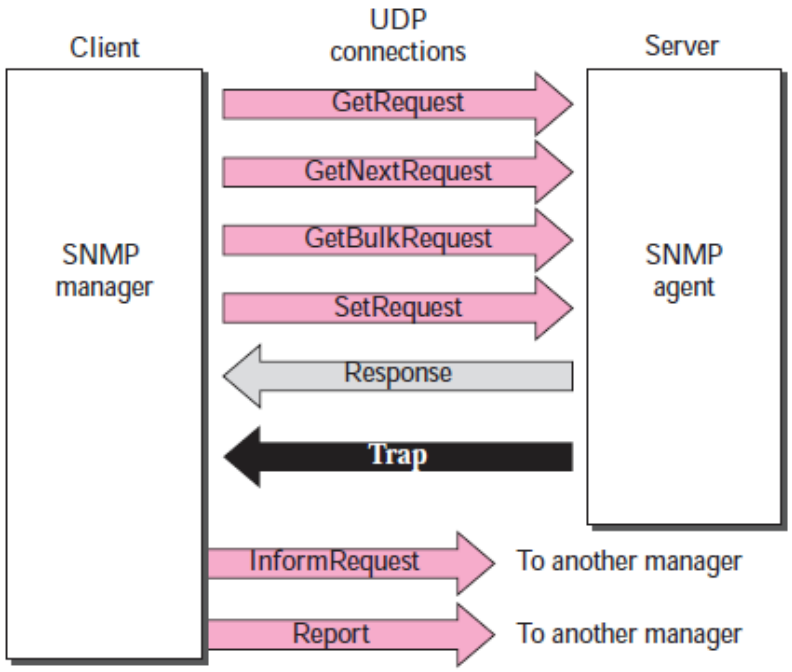
- المدير من استرجاع قيمة غرض ما معرفة ضمن العميل
- المدير من تخزين قيمة ضمن غرض ما معرف ضمن العميل
- العميل من إرسال رسالة إنذار حول وضع غير طبيعي للمدير.

### 1.5.1. وحدات معطيات البروتوكول Protocol Data Units (PDUs)

يعرف بروتوكول SNMPv3 ثمانية أنماط لوحدات معطيات البروتوكول:

1. GetRequest
2. GetNextRequest
3. GetBulkRequest
4. SetRequest
5. Response
6. Trap
7. InformationRequest
8. Report

كما هو موضح في الشكل التالي:




**الشكل 14– وحدات معطيات البروتوكول PDUs**

#### طلب GetRequest

يرسل من المدير للعميل لاسترجاع قيمة متحول ما أو مجموعة من المتحولات.

### طلب GetNextRequest

يرسل من المدير إلى العميل لاسترجاع قيمة متحول ما. تكون القيمة المسترجعة هي قيمة الغرض الذي يتبع معرف الغرض ObjectId ضمن PDU. تستخدم عادةً لاسترجاع قيم المداخل ضمن الجداول عندما لا يعرف المدير آلية فهرسة المداخل التالية. في هذه الحالة، يستخدم المدير الطلب GetNextRequest ويحدد معرف غرض الجدول الطلوب Object Id of the table ويكون الجواب هو قيمة أول مدخل ضمن الجدول. وهكذا، يستخدم المدير قيمة أول مدخل ضمن الجدول للحصول على قيمة المدخل الثاني من خلال طلب GetNextRequest ويتابع للوصول إلى آخر مدخل.

### طلب GetBulkRequest

يرسل من المدير إلى العميل لاسترجاع كمية كبيرة من المعلومات. يمكن استخدامها بدلاً عن عدة طلبات من نوع GetRequest أو GetNextRequest.

### طلب SetRequest

يرسل من المدير إلى العميل لتخزين(وضع) قيمة ضمن متحول.

### الجواب Response

يرسل من العميل إلى المدير كجواب على GetRequest أو GetNextRequest. يحوي قيمة أو قيم المتحول(ات) المطلوبة من المدير.

### الفخ Trap

يرسل من العميل إلى المدير للإعلان عن حدث ما. كإعادة إقلاع العميل مثلاً.

### طلب InformationRequest

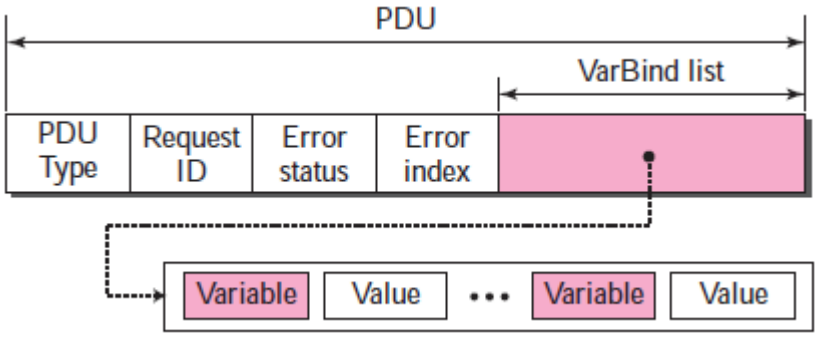
يرسل من أحد المديرين إلى مدير آخر بعيد للحصول على قيمة بعض المتحولات من عملاء تحت تصرف المدير البعيد. يجيب المدير البعيد بإرسال Response PDU.

### التقرير Report

يستخدم التقرير للإعلان عن بعض أنواع الأخطاء بين المديرين. لا يجري استخدامه حالياً.

### 2.5.1. صيغة وحدات معطيات البروتوكول

يوضح الشكل التالي صيغة PDUs الثمانية.




**الشكل 15– صيغة SNMP PDU**

نلاحظ هنا أن طرد (PDU) GetBulkRequest يختلف عن باقي الطرود في أمرين:

1. تكون قيمة حالة الخطأ Error Status وقيمة حقل فهرس الخطأ Error Index هي صفر في جميع الطرود ما عدا GetBulkRequest.
2. يستبدل حقل Error status بحقل None-repeater وحقل error index بحقل max-repetitions ضمن GetBulkRequest.

الحقول هي كالتالي:

- نمط PDU: يبين الجدول التالي أنماط PDU المختلفة:

| Type | Tag (Binary) | Tag (Hex) |
|---|---|---|
| GetRequest | 10100000 | A0 |
| GetNextRequest | 10100001 | A1 |
| Response | 10100010 | A2 |
| SetRequest | 10100011 | A3 |
| GetBulkRequest | 10100101 | A5 |
| InformRequest | 10100110 | A6 |
| Trap (SNMPv2) | 10100111 | A7 |
| Report | 10101000 | A8 |

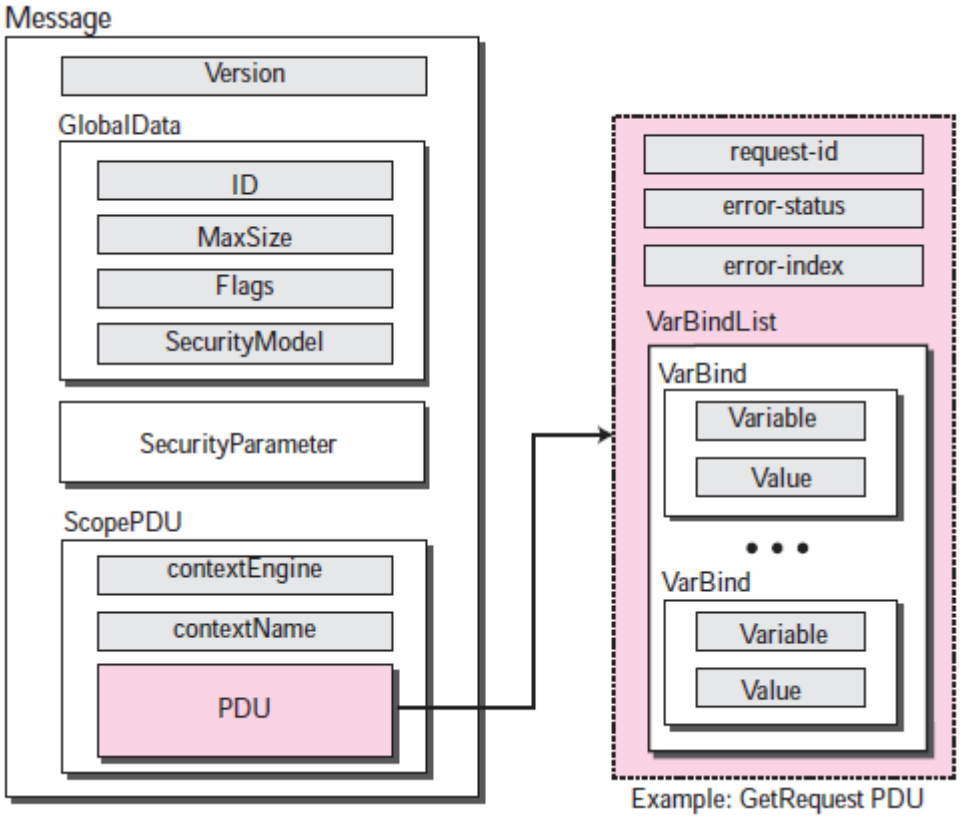
- معرف الطلب Request ID: هذا الحقل هو عبارة عن رقم تسلسل يستخدمه المدير ضمن طرد الطلب ويكرره العميل في الجواب. أي يستخدم لربط الطلب بالجواب.
- حالة الخطأ Error status: عدد طبيعي يستخدم في طرود (PDUs) الأجوبة لبيان نوع الخطأ الذي يصادفه العميل. تكون قيمته مساوية للصفر في طرد السؤال. يبين الشكل التالي أنواع الأخطاء المختلفة:

| الحالة | الاسم | المعنى |
|---|---|---|
| 0 | noError | لا يوجد أخطاء |
| 1 | tooBig | الجواب أكبر من أن يوضع في رسالة واحدة |
| 2 | noSuchName | المتحول غير موجود |
| 3 | badValue | القيمة المطلوب تخزينها غير صالحة |
| 4 | readOnly | لا يمكن تعديل القيمة |
| 5 | genErr | خطأ آخر |

- بدون مكررين None-repeaters: يستخدم مع GetBulkRequest بدلاً عن Error status.
- فهرس الخطأ Error index: انزياح يدل على أي متحول سبب الخطأ.
- التكرار الأعظمي Max-repetitions: يستخدم فقط مع GetBulkRequest بدلاً عن Error index.
- قائمة المتحولات VarBind: مجموعة من المتحولات مع القيم الموافقة لهم والتي يريد المدير استرجاعها أو وضع قيم لها. تكون القيم معدومة Null في طلبات GetRequest أو GetNextRequest. في طرد من نوع Trap، تزود المتحولات وقيمها المتعلقة بطرد محدد.

### 3.5.1. الرسائل Messages

يضع بروتوكول SNMP الطرد PDU ضمن رسالة. تتألف الرسالة في الإصدار SNMPv3 من أربعة عناصر: الإصدار Version، والمعطيات العامة GlobalData، ووسطاء الحماية SecurityParameters، وطرد المجال ScopePDU (الذي يحوي طرد PDU المرمز) كما هو موضح في الشكل التالي:




**الشكل 16– رسالة SNMPv3**

يكون العنصران الأول والثالث من النوع البسيط بينما يكون العنصران الثاني والرابع من النوع "تسلسل".

الإصدار Version

عدد طبيعي INTEGER لتحديد الإصدار. الإصدار الحالي هو 3.

المعطيات العامة GlobalData

هو عبارة عن تسلسل مكونة من أربعة عناصر من النوع البسيط: ID, Max-Size, Flags, and SecurityModel.

### وسطاء الحماية SecurityParameters

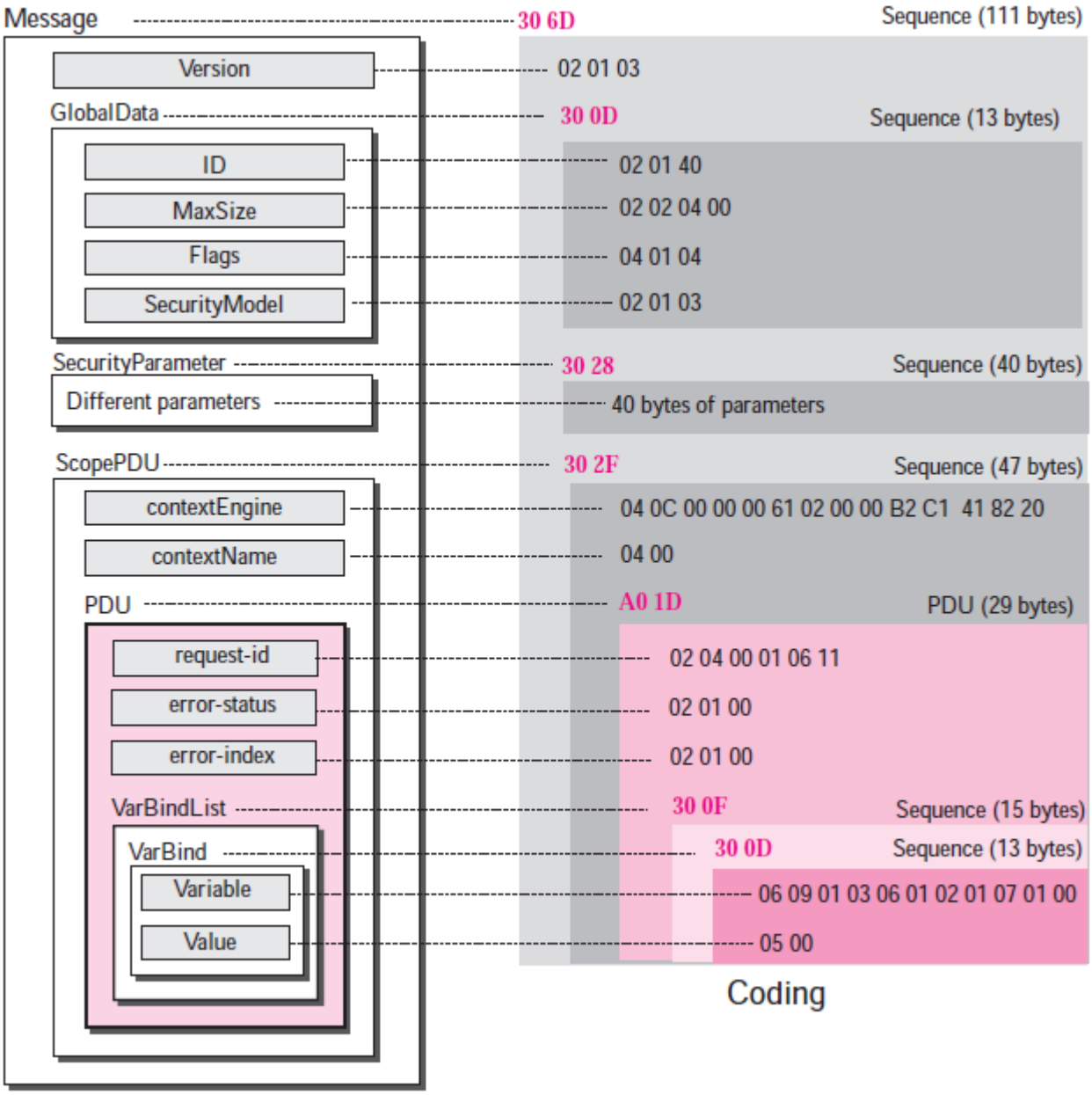
تتعلق درجة تعقيد هذا العنصر، وهو من نوع sequence، بنوع الحماية المطبق في الإصدار الثالث.

### المجال ScopePDU

يحوي ScopePDU عنصرين بسيطين والطرد نفسه. لاحظ هنا أن قائمة VarBind هي تسلسل مكونة من واحدة أو أكثر من التسلسلات التي تدعى VarBind. تتألف كل VarBind من نمطين بسيطين: متحول وقيمة Variable and Value.

### مثال

يحاول المدير في هذا المثال استخدام رسالة تحمل وحدة معطيات GetRequest PDU لاسترجاع عدد طرود UDP التي استقبلها مسير (الشكل التالي).




**الشكل 17– مثال عن رسالة SNMP**

يوجد ضمن الرسالة السابقة تسلسل واحد VarBind sequence.

متحول MIB المقابل لهذه المعلومة هو udpInDatagrams ذو المعرف 1.3.6.1.2.1.7.1.0.

بما أن المدير يريد استرجاع (قراءة) القيمة المخزنة ضمن المتحول فالقيمة تكون معدومة Null. مثلنا البايتات المرسلة بالتمثيل الست عشري.

يوجد متحول واحد ضمن قائمة VarBind. نمط المتحول هو 6 وطوله 09. القيمة هي نمط 05 والطول 00. قائمة VarBind هي تسلسل ذات طول 0D (أي 13). قائمة VarBind هي أيضاً تسلسل ذات طول 0F (أي 15). طول طرد GetRequest هو 1D (أي 29) جرى تغليف الطرد PDU ضمن تسلسل ScopePDU التي تشمل على 47 بايت. جرى أيضاً تخصيص 40 بايت للوسيط Security. بالنسبة لحقل GlobalData فهو تسلسل مكونة من 13 بايت. فيما يتعلق بتسلسل Message فهي مؤلفة من 3 سلاسل وعدد طبيعي (Version) الذي يشغل 111 بايت. الرسالة ككل هي 113بايت. يبين الشكل التالي الرسالة الفعلية المرسلة.

Message

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 30 | 6D | 02 | 01 | 03 | 30 | 0D | 02 | 01 | 04 | 02 | 02 | 04 | 00 | 04 | 01 |
| 04 | 02 | 01 | 03 | 30 | 28 | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | 30 | 2F |
| 04 | 0C | 00 | 00 | 00 | 61 | 02 | 00 | 00 | B2 | C1 | 41 | 82 | 20 | 04 | 00 |
| A0 | 1D | 02 | 04 | 00 | 01 | 06 | 11 | 02 | 01 | 00 | 02 | 01 | 00 | 30 | 0F |
| 30 | 0D | 06 | 09 | 01 | 03 | 06 | 01 | 02 | 01 | 07 | 01 | 00 | 05 | 00 | |

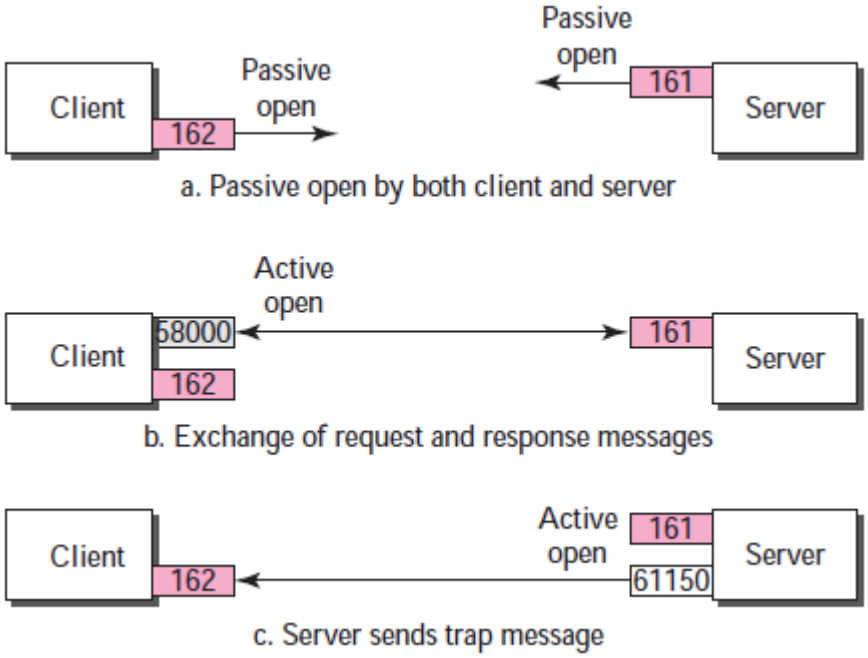
**الشكل 18– الرسالة الفعلية المرسلة**

### 6.1. بوابات SNMP

يستخدم بروتوكول SNMP خدمات UDP على بوابتين معروفتين Well-known ports، 161 و 162. تستخدم البوابة 161 من قبل المخدم (العميل) بينما تستخدم البوابة 162 من قبل الزبون (المدير).

يتنصت العميل على البوابة 161 عن طريق استدعاء Passive Open وينتظر اتصال من طرف المخدم. يفتح المدير اتصال مع العميل عن طريق استدعاء Active Open على بوابة ديناميكية. يجري هنا استخدام البوابتين السابقتين لتبادل الرسائل بين المدير والعميل.

بالنسبة للرسالة Trap فتكون بين بوابة ديناميكية من طرف العميل والبوابة 162 التي يتنصت عليها المدير ويكون الاتصال باتجاه واحد. يبين الشكل التالي آلية التعامل مع البوابات.




الشكل 19 – البوابات التي يستخدمها SNMP

## 2. تمارين

| Dec | Hx | Oct | Char | Dec | Hx | Oct | Html | Chr | Dec | Hx | Oct | Html | Chr | Dec | Hx | Oct | Html | Chr |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 000 | NUL (null) | 32 | 20 | 040 |   | Space | 64 | 40 | 100 | @ | @ | 96 | 60 | 140 | ` | ` |
| 1 | 1 | 001 | SOH (start of heading) | 33 | 21 | 041 | ! | ! | 65 | 41 | 101 | A | A | 97 | 61 | 141 | a | a |
| 2 | 2 | 002 | STX (start of text) | 34 | 22 | 042 | " | " | 66 | 42 | 102 | B | B | 98 | 62 | 142 | b | b |
| 3 | 3 | 003 | ETX (end of text) | 35 | 23 | 043 | # | # | 67 | 43 | 103 | C | C | 99 | 63 | 143 | c | c |
| 4 | 4 | 004 | EOT (end of transmission) | 36 | 24 | 044 | $ | $ | 68 | 44 | 104 | D | D | 100 | 64 | 144 | d | d |
| 5 | 5 | 005 | ENQ (enquiry) | 37 | 25 | 045 | % | % | 69 | 45 | 105 | E | E | 101 | 65 | 145 | e | e |
| 6 | 6 | 006 | ACK (acknowledge) | 38 | 26 | 046 | & | & | 70 | 46 | 106 | F | F | 102 | 66 | 146 | f | f |
| 7 | 7 | 007 | BEL (bell) | 39 | 27 | 047 | ' | ' | 71 | 47 | 107 | G | G | 103 | 67 | 147 | g | g |
| 8 | 8 | 010 | BS (backspace) | 40 | 28 | 050 | ( | ( | 72 | 48 | 110 | H | H | 104 | 68 | 150 | h | h |
| 9 | 9 | 011 | TAB (horizontal tab) | 41 | 29 | 051 | ) | ) | 73 | 49 | 111 | I | I | 105 | 69 | 151 | i | i |
| 10 | A | 012 | LF (NL line feed, new line) | 42 | 2A | 052 | * | * | 74 | 4A | 112 | J | J | 106 | 6A | 152 | j | j |
| 11 | B | 013 | VT (vertical tab) | 43 | 2B | 053 | + | + | 75 | 4B | 113 | K | K | 107 | 6B | 153 | k | k |
| 12 | C | 014 | FF (NP form feed, new page) | 44 | 2C | 054 | , | , | 76 | 4C | 114 | L | L | 108 | 6C | 154 | l | l |
| 13 | D | 015 | CR (carriage return) | 45 | 2D | 055 | - | - | 77 | 4D | 115 | M | M | 109 | 6D | 155 | m | m |
| 14 | E | 016 | SO (shift out) | 46 | 2E | 056 | . | . | 78 | 4E | 116 | N | N | 110 | 6E | 156 | n | n |
| 15 | F | 017 | SI (shift in) | 47 | 2F | 057 | / | / | 79 | 4F | 117 | O | O | 111 | 6F | 157 | o | o |
| 16 | 10 | 020 | DLE (data link escape) | 48 | 30 | 060 | 0 | 0 | 80 | 50 | 120 | P | P | 112 | 70 | 160 | p | p |
| 17 | 11 | 021 | DC1 (device control 1) | 49 | 31 | 061 | 1 | 1 | 81 | 51 | 121 | Q | Q | 113 | 71 | 161 | q | q |
| 18 | 12 | 022 | DC2 (device control 2) | 50 | 32 | 062 | 2 | 2 | 82 | 52 | 122 | R | R | 114 | 72 | 162 | r | r |
| 19 | 13 | 023 | DC3 (device control 3) | 51 | 33 | 063 | 3 | 3 | 83 | 53 | 123 | S | S | 115 | 73 | 163 | s | s |
| 20 | 14 | 024 | DC4 (device control 4) | 52 | 34 | 064 | 4 | 4 | 84 | 54 | 124 | T | T | 116 | 74 | 164 | t | t |
| 21 | 15 | 025 | NAK (negative acknowledge) | 53 | 35 | 065 | 5 | 5 | 85 | 55 | 125 | U | U | 117 | 75 | 165 | u | u |
| 22 | 16 | 026 | SYN (synchronous idle) | 54 | 36 | 066 | 6 | 6 | 86 | 56 | 126 | V | V | 118 | 76 | 166 | v | v |
| 23 | 17 | 027 | ETB (end of trans. block) | 55 | 37 | 067 | 7 | 7 | 87 | 57 | 127 | W | W | 119 | 77 | 167 | w | w |
| 24 | 18 | 030 | CAN (cancel) | 56 | 38 | 070 | 8 | 8 | 88 | 58 | 130 | X | X | 120 | 78 | 170 | x | x |
| 25 | 19 | 031 | EM (end of medium) | 57 | 39 | 071 | 9 | 9 | 89 | 59 | 131 | Y | Y | 121 | 79 | 171 | y | y |
| 26 | 1A | 032 | SUB (substitute) | 58 | 3A | 072 | : | : | 90 | 5A | 132 | Z | Z | 122 | 7A | 172 | z | z |
| 27 | 1B | 033 | ESC (escape) | 59 | 3B | 073 | ; | ; | 91 | 5B | 133 | [ | [ | 123 | 7B | 173 | { | { |
| 28 | 1C | 034 | FS (file separator) | 60 | 3C | 074 | < | < | 92 | 5C | 134 | \ | \ | 124 | 7C | 174 | | | \| |
| 29 | 1D | 035 | GS (group separator) | 61 | 3D | 075 | = | = | 93 | 5D | 135 | ] | ] | 125 | 7D | 175 | } | } |
| 30 | 1E | 036 | RS (record separator) | 62 | 3E | 076 | > | > | 94 | 5E | 136 | ^ | ^ | 126 | 7E | 176 | ~ | ~ |
| 31 | 1F | 037 | US (unit separator) | 63 | 3F | 077 | ? | ? | 95 | 5F | 137 | _ | _ | 127 | 7F | 177 | &#127; | DEL |

Source: www.LookupTables.com

**الشكل 20– جدول ASCII**

### 1.2. التمرين 1

أوجد ترميز العدد الطبيعي 1456 INTEGER

الجواب:

| | |
|---|---|
| **INTEGER tag** | 02 |
| **length** | 04 |
| **value** | 00 00 05 B0 |
| **Answer** | 02 04 00 00 05 B0 |

## 2.2. التمرين 2

أوجد طريقة ترميز السجل التالي:

```
TIME TICK INTEGER   Object Id
12000     14564     1.3.6.1.2.1.7
```

الجواب

| | |
|---|---|
| 30 15 | sequence, length |
| 43 04 00 00 2E E0 | TIME TICK, length, value (12000) |
| 02 04 00 00 38 E4 | INTEGER, length, value (14564) |
| 06 07 01 03 06 01 02 01 07 | Object ID, length, value (1.3.6.2.1.7) |

## 3.2. تمرين 3

أوجد طريقة ترميز السجل التالي:

```
INTEGER   OCTET STRING   Counter
2345 "COMPUTER"     345
1123 "DISK"    1430
3456 "MONITOR" 2313
```

الجواب:

| | |
|---|---|
| 30 43 | sequence, length |
| 30 41 | sequence, length |
| 02 04 00 00 09 29 | INTEGER, length, value (2345) |
| 04 08 43 4F 4D 50 55 54 45 52 | OCTET STRING, length, value (COMPUTER) |
| 41 04 00 00 01 59 | counter, length, value (345) |
| 30 29 | sequence, length |
| 02 04 00 00 04 63 | INTEGER, length, value (1123) |
| 04 04 44 49 53 4B | OCTET STRING, length, value (DISK) |
| 41 04 00 00 05 96 | counter, length, value (1430) |
| 30 15 | sequence, length |
| 02 04 00 00 0D 80 | INTEGER, length, value (3456) |
| 04 07 4D 4F 4E 49 54 4F 52 | OCTET STRING, length, value (MONITOR) |
| 41 04 00 00 09 09 | counter, length, value (2313) |

### 4.2.تمرين 4

فك ترميز التالي:

a. `02 04 01 02 14 32`
b. `30 06 02 01 11 02 01 14`
c. `30 09 04 03 41 43 42 02 02 14 14`
d. `30 0A 40 04 23 51 62 71 02 02 14 12`

الجواب:

a. `integer 16913458`
b. `sequence of 2 integers: 17 and 20`
c. `sequence of the string: "ACB" and the integer 5140`
d. `sequence of an IP address 35.81.98.113 and the integer 5138`

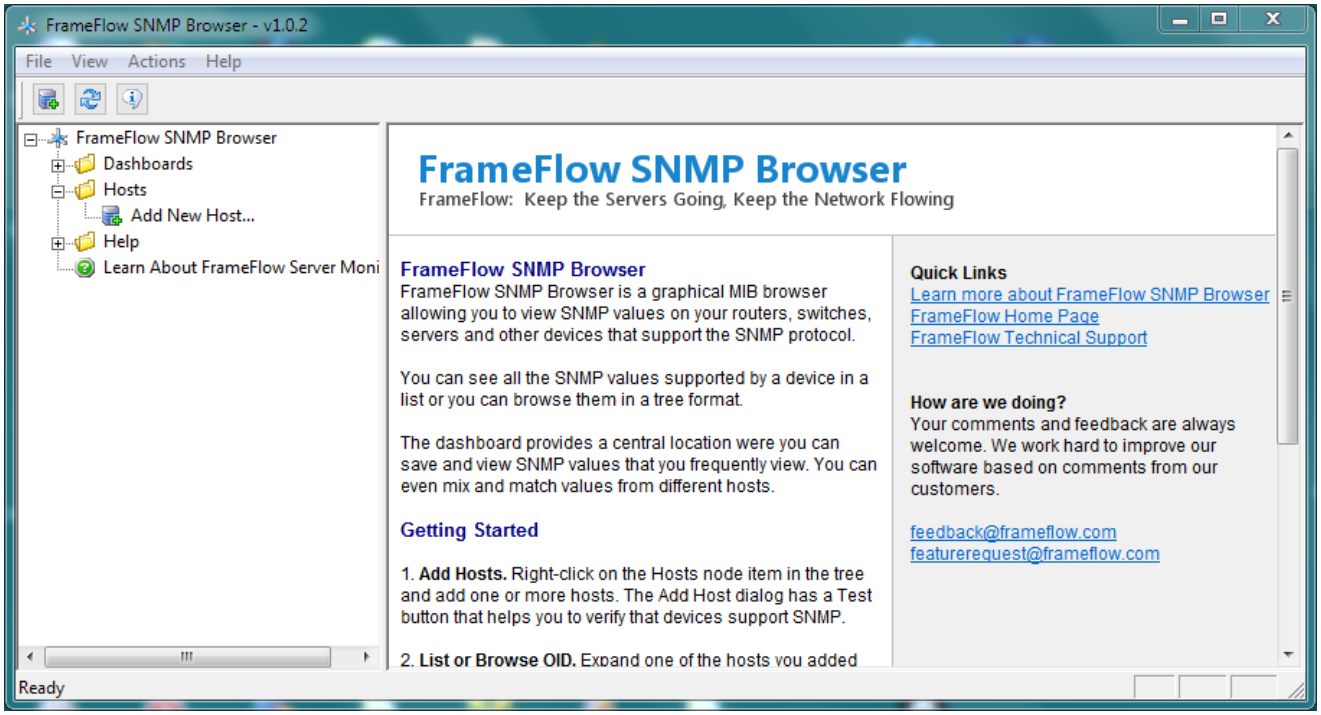
## 3. تجارب SNMP

### 1.3. استخلاص معلومات عن نقطة النفاذ Access Point

سنقوم في هذه التجربة بتنصيب برنامج FrameFlow الذي يعمل كمدير SNMP بغية استخلاص معلومات عن نقطة النفاذ أو أي جهاز شبكي آخر وحاسب.

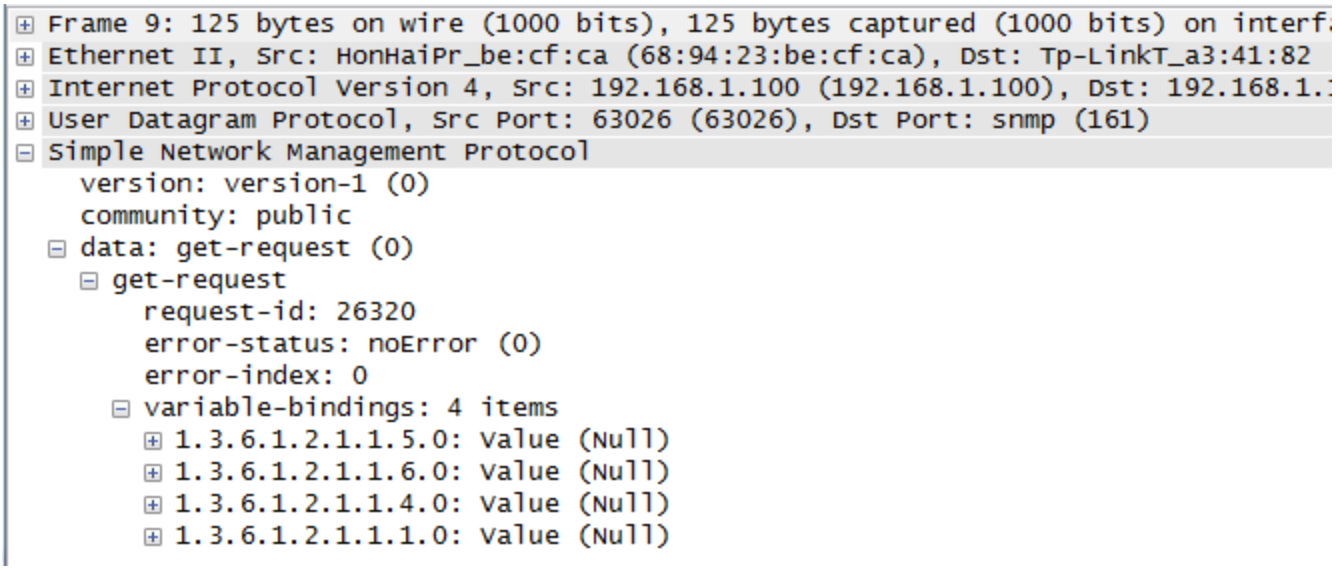
سنقوم بالخطوات التالية:

1. تنصيب برنامج FrameFlow ومن ثم تشغيله فتظهر نافذة مثل الشكل التالي:




**الشكل 21– نافذة Frame Flow**

2. نضغط على إضافة مضيف جديد Add new host وندخل عنوان IP لهذا المضيف. سيكون العنوان في حالتنا هو 192.168.1.1 لنقطة النفاذ اللاسلكية.
3. نشغل برنامج Wireshark ونبدأ التقاط الطرود
4. من برنامج FrameFlow نختار Host Summary تحت العنوان 192.168.1.1 فنلاحظ إرسال طرد SNMP إلى نقطة النفاذ واستقبال الجواب.

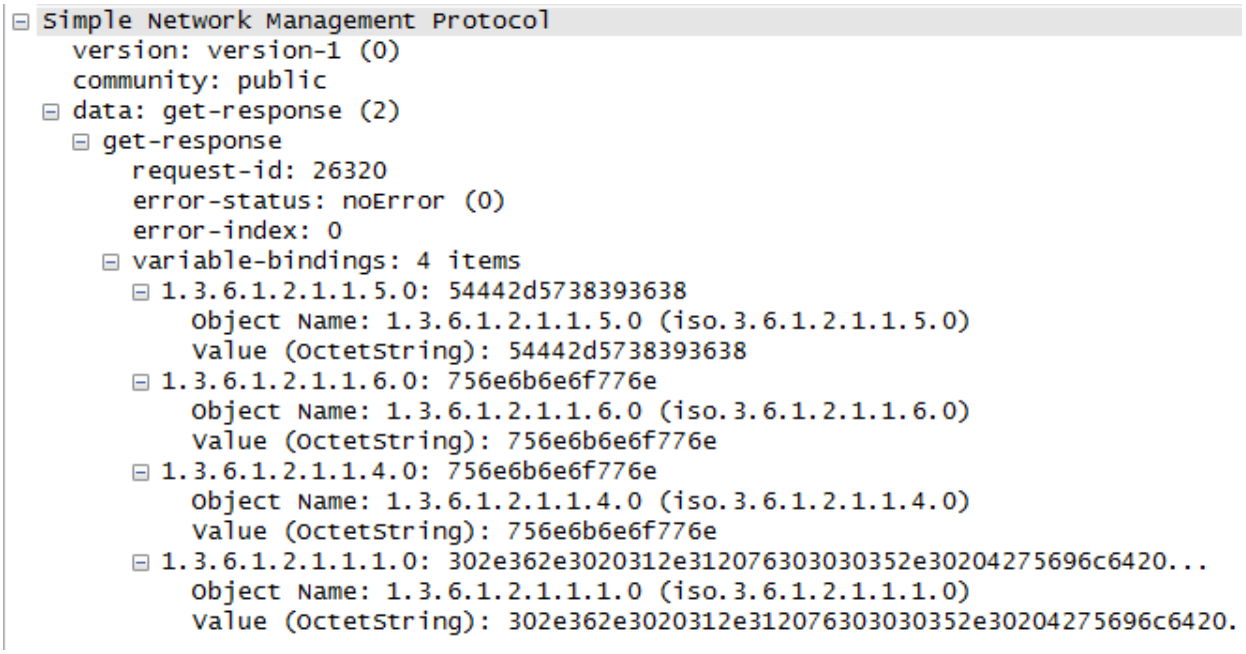



**الشكل 22– طرد طلب معلومات Host Summery**

نلاحظ من الشكل السابق:

- عناوين IP للمرسل وللوجهة
- البروتوكول الذي يستخدمه SNMP والبوابات المستخدمة
- نوع طلب SNMP
- المتحولات التي جرى طلبها: 1.3.6.1.2.1.1.1 – 1.3.6.1.2.1.1.6

يبين الشكل التالي جواب عميل SNMP:




**الشكل 23– جواب طلب Host summery**

نلاحظ أن الجواب يحوي المعلومات المطلوبة.

### 3.2. معرفة عدد البايتات المرسلة على كل بطاقة شبكة

المطلوب:

1. اختر أحد الأجهزة القابلة للإدارة: PC, Access Point, Switch, Router, Server
2. شغل كل من FrameFlow and Wireshark
3. أبحث على المتحول ifInOctets وأجب عن الأسئلة التالية:

أ. ما عدد بطاقات الشبكة الموجودة ضمن الجهاز الذي تديره؟

ب. ما عدد بطاقات الشبكة التي تستقبل معطيات؟

ت. أملئ الجدول التالي:

| Interface Description | IN (Octets) | Out (Octets) | In UnicastPackets | Out UnicastPackets |
|---|---|---|---|---|
| | | | | |
| | | | | |

بالنسبة للمتحول ifINOctets، شغل Wireshark وأجب عما يلي:

1. ما هي معرفات المتحولات OID التي استخدمتها ضمن طلب SNMP Get؟
2. ما هو ترميز أول متحول مطلوب؟
3. ما نوع المتحولات المستخدمة؟