# ПІДХІД ДО МОДЕРНІЗАЦІЇ СЕРВЕРНОГО ЗАБЕЗПЕЧЕННЯ ТА МЕРЕЖНОЇ ТОПОЛОГІЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

***О. О. Каплун***,

Україна, Київ, Інститут інформаційних технологій і засобів навчання НАПН України

"Якщо ви знайшли помилку, краще виправити її самостійно, ніж просити кого-небудь зробити це. Якщо ви роздумуєте про підвищення продуктивності або про додавання нових можливостей, простіше зробити це самостійно. Якщо виявляється дірка в системі безпеки, «залатайте» її самостійно і не чекайте, поки цим займеться постачальник ОС."

*Скотт Максвелл*

## Поточна структура мережі. Аналіз топології та маршрутизації.

Головною функцією мережі відділу інформаційних ресурсів і мережних технологій Інституту інформаційних технологій і засобів навчання НАПН України є забезпечення безперебійним доступом до мережі Internet співробітників відділу та забезпечення цілодобової роботи Internet-ресурсів відділу. Розглянемо схематичне відображення топології мережі до модернізації (рис. 1).

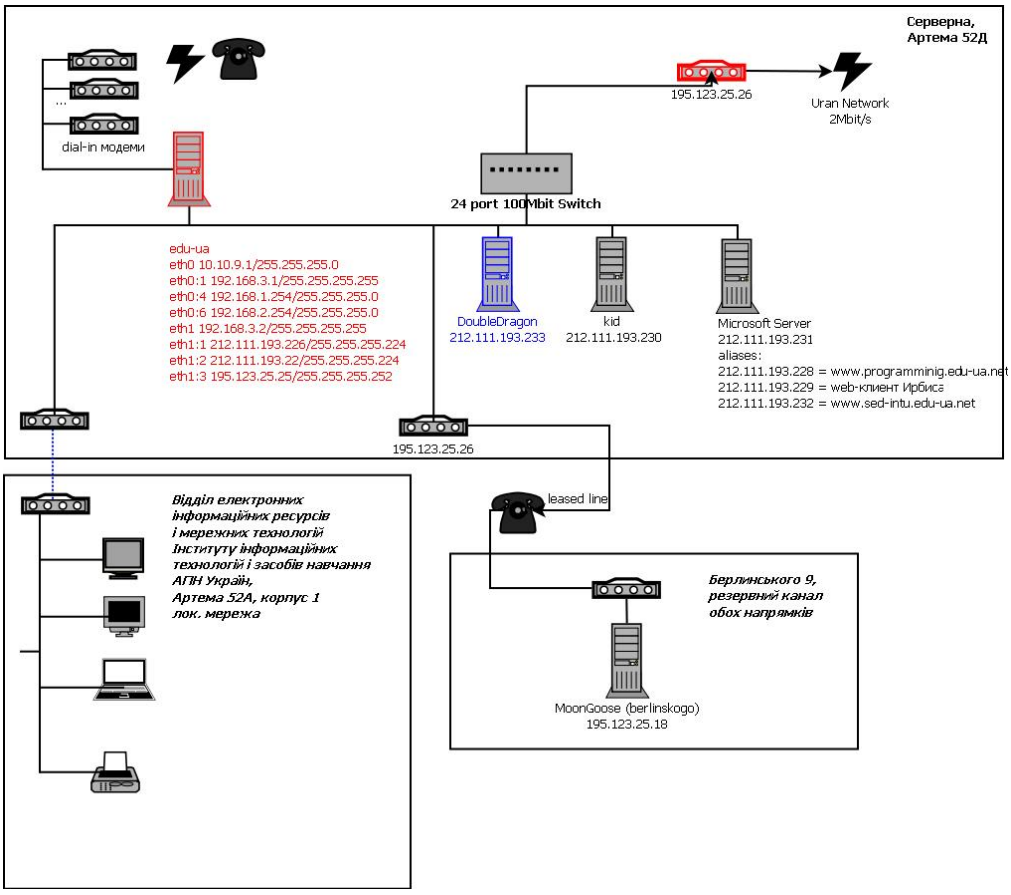


*Рис. 1. Мережа відділу до початку модернізацій*

На схематичному зображені відображено три основних сегмента мережі – це основний серверний сегмент, мережа відділу та серверний сегмент, який розташовано в окремому приміщенні, і який являє собою резервний канал на випадок падіння основного каналу.

Головний серверний сегмент являє собою чотири сервера та модеми-роутери, які фізично пов'язані між собою через один 100 Мбіт хаб. Головний канал, який забезпечує доступ всього сегменту до глобальної мережі Internet, організовано за допомогою модему (на рис.1 позначено червоним кольором), підключеного через виділену лінію до Інтернет-провайдера УРАН. Максимальна пропускна спроможність каналу на даний час являє 2 Мбіт/с.

До головного серверного сегменту належать чотири сервери, це :

1. **"Microsoft server"**, який працює під управлінням ОС Microsoft Windows Server та забезпечує роботу веб-ресурсів Інституту, забезпечує роботу пошти домену edu-ua.net;
2. **"Kid"**, сервер який працює під управлінням ОС Linux, забезпечує роботу електронної пошти, працює як веб-сервер, dns-сервер та проксі;
3. "**EDU-UA**" – сервер який працює під управлінням ОС Linux, працює центральним маршрутизатором мережі, проксі-сервером та сервером який забезпечує "dial-up" доступ до мережі Інтернет через модемний пул.

Розглянувши топологію мережі ми побачимо, що одне фізичне середовище використовується одразу кількома логічними сегментами, центральним маршрутизатором яких є сервер EDU-UA, та центральним фізичним реплікатором – 100Мбіт хаб. Така організація не є оптимальною і призводить до частої появи колізій, що в свою чергу позначається на загальній швидкості та надійності мережі.

Проаналізувавши фізичну та логічну організацію та структуру мережі, можна зробити висновки, що головні елементи, від яких залежить швидкість та надійність мережі – це центральний хаб та сервер – маршрутизатор EDU-UA. Таким чином модернізацію та вдосконалення мережі потрібно починати з них.

**Модернізація апаратної складової мережі, перехід на новий сервер**

Як бачимо з схеми топології мережі, основним сервером, виконуючим роботу маршрутизації та обслуговуючим мережу, є сервер "edu-ua". Тому ефективність та надійність роботи мережі цілком та в основному залежіть від спроможності та потужності цього серверу. Розглянемо його апаратну конфігурацію та проаналізуємо її.

Сервер "edu-ua" має таку апаратну конфігурацію:

*Процесор: AMD-K6 233Mhz*
*Пам'ять: 62Mb 72pin simm*
*Материнська плата: ISA та PCI слоти, AT блок живлення*
*Жорсткий диск: 3.6 Gb інтерфейс IDE*
*Плата Digiboard PC/16e ISA для підтримки модемного пулу*

Сервер працює під управлінням ОС Red Hat Linux 3.4.2-6.fc3 (Fedora Core 3) з ядром версії 2.6.9. Станом на поточний час це дуже застаріла ОС та версія ядра. Застаріла ОС та невеликі апаратні ресурси ведуть до неспроможності оновлювати та модернізувати програмне забезпечення на ньому, а це може призвести до потенційної небезпеки, злому зловмисниками та ненадійності роботи самого забезпечення у постійно зростаючих на сервер навантаженнях.

Як ми бачимо, така конфігурація, як апаратна, так і програмна, вже застаріла і потребує модернізації. Модернізація програмного забезпечення цього серверу не можлива, поки не буде проведено апаратну модернізацію, бо нове програмне забезпечення потребує більше пам'яті та потужності процесора. Тому спочатку розглянемо, яким чином можливо провести апаратну модернізацію сервера.

У нашому випадку сервер виступає також як dial-in сервер та має у своєму складі 16-ти портову плату DigiBoard, до якої підключені модеми для телефонного Інтернет-доступу. Ця плата має ISA інтерфейс, її під'єднано до материнської плати серверу через ISA роз'єм. Це є великою проблемою, тому що такий інтерфейс дуже давно не використовується та є морально застарілим. Нові портові плати DigiBoard представлено на ринку обмеженою кількістю, та коштують вони дуже дорого у зв'язку з тим, що модемний Інтернет використовується у наш час дуже рідко. Тому було прийнято рішення залишити цю плату та підібрати платформу з підтримкою ISA інтерфейсу.

Така платформа була знайдена. Це intel-based сервер, побудований на материнський платі чипсету i440bx. Такий сервер має потужний tower-корпус з потужним блоком живлення, пристрій для гарячої заміни SCSI жорстких дисків з активним охолодженням. Сервер підтримує роботу з двома процесорами. Так, цей сервер не є цілком сучасним, але він підтримує ISA інтерфейс, що дає можливість використання в ньому плати DigiBoard ISA, модернізації, та має підтримку таких технологій, як PCI, USB, SCSI. Материнська плата серверу має також в своєму складі один мережний порт швидкістю 1Gbit, а також можливості моніторингу та віддаленого управління, що є дуже суттєвим для адміністрування серверною платформою.

**Конфігурація "нової" платформи сервера:**

*Процесор 2x350Mhz Pentium 2*
*Пам'ять 528Mb DIMM*
*Плата Intel 440BX*
*Жорстки диски 3x SCSI 3.2 Gb + 1xIDE 40Gb*
*Інши комплектуючі DigiBoard PC/16e, 2x 100Mbit Network Card*

Якщо порівнювати цю платформу з конфігурацією серверу edu-ua, то ми бачимо що потужність ЦПУ зросла понад 300%, розмір пам'яті - на 825%. Така конфігурація вже достатньо потужна, щоб обслуговувати нашу мережу під управлінням сучасної мережної ОС та нести додаткове навантаження, таке як робота проксі-серверу, ДНС-серверу та інші.

Слід зауважити, що метою апгрейда серверу “EDU-UA” не є його заміна на надсучасну та потужну платформу. Ціль модернізації є забезпечення при мінімальних витратах виконання усіх тих функцій, які виконував старий сервер, відновити програмне забезпечення та створити можливість збільшення навантаження на сервер. Таким чином ми отримали, можливо, не надсучасний, але потужніший, ніж попередній сервер, який зміг обслуговувати старе апаратне забезпечення (Digiboard PC/16e ISA) та має більше можливостей для апгрейду.

**Підбір та установка ОС на модернізований сервер**

Чому вибір пав саме на дистрибутив Ubuntu? Ubuntu – операційна система для робочих станцій, лептопів і серверів, є найпопулярнішим у світі дистрибутивом Linux. Серед основних цілей Ubuntu – надання сучасного і водночас стабільного програмного забезпечення для пересічного користувача із сильним акцентом на простоту встановлення і користування.

Для інсталяції на сервер було вибрано серверну версію Ubuntu 4.2.4 з ядром версії 2.6.24 як більш стабільної та гнучкої. На сервер спочатку встановлюється основна частина системи, після чого всі необхідні пакети (архівні файлі і метадані для інсталяції та видалення архівних файлів) можуть завантажуватися та встановлюватися з Internet репозиторія. Така схема дуже гнучка та забезпечує користувача найновішими версіями пакетів на момент їх встановлення.

Було встановлено ядро для підтримки двох процесорів у режимі “symmetric multiprocessing”.

**Проблема підтримки мультипортової плати DigiBoard у новій операційній системі**

Одним з ключових моментів апгрейда є підтримка старої мультипортової ISA плати DigiBoard PC/16e. Якщо з боку апаратної структури ця проблема була вирішена підтримкою шини ISA та формфактору плати новою платформою, то з боку програмного забезпечення виникли складності.

Один з основних недоліків плати DigiBoard PC/16e – це її моральна застарілість та відмова від її підтримки виробником. Тому офіційних драйверів/модулів для цієї плати під сучасні опреційниі системи просто не існує. Але є два модулі, написанні ентузіастами та розробниками ОС Linux для серії плат, до якої належить наша плата. Це модулі **pcxx.c** та **epca.c**:

**Pcxx.c** - Digiboard PC/X{i,e,eve} driver v1.6.3_e, драйвер перша версія якого з’явилась у 1995 році та була написана Христофором Ламетером. Драйвер працює тільки з ISA картами та має вбудовану можливість ініціювання bios-програми плати.

**Epca.c** - це офіційний драйвер Digiboard, який був змінений програмістами для підтримки нових ядер версій 2.6.X. Драйвер підтримує старі ISA та сучасніші PCI плати, але не має можливості ініціювання bios-програми ISA-плат. Для цього використовувалась окрема програма, яка у коди ядра не входить.

Ці модулі підтримувались ядром до версії 2.6.11. Після цього Лінусом Товальдсом (розробник ядра Linux та засновник операційної системи Linux) було прийнято рішення видалити модулі з початкових кодів ядра у зв'язку з відсутністю підтримки з боку програмістів в адаптуванні цих модулів під нові версії. Відсутність підтримки пояснюється тим, що самі плати, для яких були написані модулі, дуже застаріли та dial-in метод доступу до Internet, для яких вони використовувались, стає все менш популярним.

Встановлення старого ядра версії, в якому ці модулі були присутні, робить багато сервісів системи недієспроможними, оскільки вони використовують можливості сучасніших ядер. Тому було прийнято рішення вести доробку одного з цих модулів для роботи з поточною версією ядра. Модуль *epca*, не зважаючи на те, що він є сучаснішим, у зв'язку з відсутністю вбудованої можливості ініціювати *біос* карти, а без цього карта функціювати не буде, є менш прийнятним для доробки, тому було прийнято рішення доробити модуль *pcxx.c*.

## Доробка драйвера pcxx.c для роботи в симетричному мультипроцесорному режимі ядра версії 2.6.24

Для роботи над драйвером було встановлено початкові коди ядра та пакети, які необхідні для компіляції та інсталяції ядра з початкових кодів. Ядро було сконструйовано для роботи в симетричному мультипроцесорному режимі та максимально налаштовано під апаратну конфігурацію платформи.

Після модернізації ядра завантажено останню версію початкових кодів модуля pcxx з версії ядра, де цей модуль ще підтримувався (2.6.11). Файли модуля були зібрані в окремий каталог, та за допомогою файла Makefile створено середовище для компілювання драйверу в режимі модуля окремо, без повторного компілювання всього ядра системи, що дало змогу заощадити час та максимально швидко проводити тестування роботи модуля:

```
obj-m   := pcxx.o
KDIR    := /../../linux-2.6.24/debian/build/build-generic/
PWD     := $(shell pwd)
default:
	$(MAKE) -C $(KDIR) SUBDIRS=$(PWD) modules
```

Після створення цього файлу для компіляції драйвера достатньо було лише набрати команду make, та, якщо компіляція проходила без помилок, ми отримували готовий модуль-файл pcxx.ko. Але драйвер не був готовий для роботи з новим ядром та при спробі компіляції видав наступні помилки:

```
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c: In function 'pcxe_cleanup':
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:212: error: implicit declaration of function 'save_flags'
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:213: error: implicit declaration of function 'cli'
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:222: error: implicit declaration of function 'restore_flags'
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c: In function 'pcxx_waitcarrier':
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:331: error: implicit declaration of function 'sti'
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c: At top level:
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:971: warning: initialization from incompatible pointer type
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c: In function 'pcxe_init':
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:1093: error: 'struct tty_driver' has no member named 'devfs_name'
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:1387:41: error: macro "INIT_WORK" passed 3 arguments, but takes just 2
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:1387: error: 'INIT_WORK' undeclared (first use in this function)
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:1387: error: (Each undeclared identifier is reported only once
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:1387: error: for each function it appears in.)
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c: In function 'doevent':
```

```
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:1607: error: ‘struct tty_struct’ has no member named ‘flip’
...
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c: In function ‘pcxxparam’:
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:1769: warning: assignment from incompatible pointer type
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c: In function ‘receive_data’:
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:1856: warning: assignment from incompatible pointer type
/home/alexk/edu-ua.laboratory/pcxx/pcxx.c:1884: error: ‘struct tty_struct’ has no member named ‘flip’
...
make[4]: *** [/home/alexk/edu-ua.laboratory/pcxx/pcxx.o] Error 1
make[3]: *** [_module_/home/alexk/edu-ua.laboratory/pcxx] Error 2
make[2]: *** [sub-make] Error 2
make[1]: *** [all] Error 2
make[1]: Leaving directory `/home/alexk/linux-2.6.24/debian/build/build-generic'
make: *** [default] Error 2
```

За допомогою пошукової системи було знайдено, що ці помилки викликані переходом нового ядра на новий тип „блокування” (операцій процесора для захисту даних, які він обробляє у поточний період, від спроб роботи з ними іншим процесором) при роботі у мультипроцесорному режимі, та знайдено декілька документів стосовно інших схожих модулів (drivers/char/riscom8.c, drivers/char/.moxa.c тощо) про те, як цю помилку обійти та примусити старий модуль повноцінно функціонувати з новим ядром. Завдяки посиланням в цих документів в початкових кодах ядра було знайдено файл з описом, яким чином здійснити перехід від використання механізмів блокування старого ядра до використання механізмів нового ядра, написаного Ingo Molnar. Користуючись цією документацію та по аналогії з іншими модулями, драйвер *pcxx.c* для DigiBoard було перероблено з використанням механізмів блокування нового ядра.

Після модернізації драйвера залишились ще помилки, але помилки стосовно блокування cli/sti механізмом зникли.

Помилки, що залишились, були пов’язані зі структурою ‘struct tty_struct’ та використанням у драйвері її старої версії. Рішення цієї проблеми почалося з аналізу іншого сучаснішого модуля *epca.c*. Він вже розрахований на роботу з новою структурою ‘struct tty_struct’, і оскільки ці два модуля розроблено для одного обладнання, а також вони дуже схожі за структурою, то з *epca.c.* було запозичено деякі рядки коду роботи з ‘struct tty_struct’, а також повністю функція *receive_data*, яка ідентична до функції у pcxx за параметрами виклику та даними повернення. Компіляція з урахуванням вище наведених модифікацій успішно завершилася:

```
        …
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/alexk/edu-ua.laboratory/pcxx/pcxx.mod.o
LD [M]  /home/alexk/edu-ua.laboratory/pcxx/pcxx.ko
        …
```

Таким чином, ми отримали зкомпільований під нове ядро модуль *pcxx.ko*. Після його запуску командою modprobe, в /var/log отримуємо такий запис:

```
[  134.097150] Digiboard PC/X{i,e,eve} driver v1.6.3_e
[  134.746852] PC/Xx: PC/Xe (64k) I/O=0x200 Mem=0xd0000 Ports=16
```

Цей запис показує, що драйвер вірно ідентифікував плату DigiBoard, розмір її пам’яті та кількість портів. Для того, щоб остаточно переконатись в працездатності драйвера підключаємо до портової плати DigiBoard декілька модемів, наприклад на порти позначені 1,3,5. В системі їм

будуть відповідати системні файли /dev/ttyD0, /dev/ttyD2, /dev/ttyD4 (нумерація портів в системі починається з нуля). Після підключення модемів спробуємо провести з ними обмін інформацією (на прикладі з /dev/ttyD0) за допомогою програми – терміналу “cu”:

```
alexk@doubledragon:/var/log$ cu -l ttyD0
Connected.
>ATZ
OK
>ATi9
($ZYX0304\0000DCAF\MODEM\\  U336E  V 1.18     8B)
OK
```

Бачимо, що модем нам відповів – отже обмін інформацією відбувається, а це свідчить про те, що драйвер працює.

**Перенос конфігурації та налаштування сервісу “dial-in” на сервері “DoubleDragon”**

Після того, як драйвер працює, можна починати організацію dial-in сервісу, який являє собою симбіоз двох служб: це *mgetty* – служба, яка відповідає за початкове налаштування та запуск *pppd*, і сам *pppd*, який відповідає за авторизацію користувачів та забезпечує обмін інформацією між користувачем та сервером за протоколом PPP.

Пакет *mgetty* встановлюється стандартний, який завантажується з репозиторія Ubuntu та встановлюється за допомогою пакет-менеджера apt-get.

З пакетом *pppd* виникає проблема, оскільки у стандартній версії пакету не реалізовано функцію call-back (зворотного дзвінка), у зв’язку з цим початкові коди пакету *pppd* були завантажені з репозиторія, налаштовані і встановлені самостійно. Базова версія *pppd* підтримувала тільки клієнтську версію протоколу *cbcp*, який був необхідним для забезпечення зворотнього дзвінка. Для підтримки серверної версії протоколу до початкових кодів *pppd* було застосовано „патчи” (у буквальному перекладі - "латочка", заміна у вказаному(их) файлі(ах) деякої послідовності байт) з системи ASPLinux

Після встановлення сервісів налаштовуємо їх. Пересвідчуємось, що в файлі конфігурації *mgetty* (/etc/mgetty/login.conf) є рядок:

```
/AutoPPP/ -     a_ppp   /usr/sbin/pppd auth -chap +pap login debug
```

Він автоматично запускає сервіс *pppd* при додзвоні до модему та використанні протоколу ppp на клієнтській стороні. У файл /etc/inittab прописуємо команди запуску сервісів *mgetty*, які будуть очікувати на терміналах додзвону клієнтів:

```
T0:23:respawn:/sbin/mgetty -x0 -s 57600 ttyD0
T2:23:respawn:/sbin/mgetty -x0 -s 57600 ttyD2
T6:23:respawn:/sbin/mgetty -x0 -s 57600 ttyD6
```

Виконаємо команду init –Q, щоб зміни вступили в дію та перевіримо, чи запустились необхідні процеси:

```
root@doubledragon:/etc# ps ax|grep mgetty
 4923 ?        Ss     1:08 /sbin/mgetty -x0 -s 57600 ttyD2
 5098 ?        Ss     0:59 /sbin/mgetty -x0 -s 57600 ttyD0
 7409 ?        Ss     0:00 /sbin/mgetty -x0 -s 57600 ttyD6
```

Бачимо, що на першому (ttyD0), третьому (ttyD2) та сьомому порту (ttyD6) запущено процеси mgetty, які контролюють модеми на цих портах та чекають додзвону користувачів.

Для того щоб користувач після додзвону до сервера отримав підключення до Internet, необхідно налаштувати ppp. Для цього файли options (налаштування pppd), callback-server (скріпт додзвону), pap-secrets(паролі користувачів), callback-users (користувачі, яким дозволено call-back) та denylist(список телефонів, на які додзвін заборонено) було перенесено з серверу edu-ua без змін, що дозволило зберегти усі налаштування та зробити оновлення серверу непомітним для користувачів.

Перевіряємо додзвін до серверу на один з модемів. Пересвідчуємось, що підключення відбувається, перевірка користувача (авторизація) проходить успішно та після підключення на клієнтській машині з'являється доступ до мережі Internet.

**Перенесення сервісів, служб та функціонала зі старого сервера на новий, фізична заміна сервера, зміни в топології мережі**

На цьому етапі сервер DoubleDragon вже виконує функції "Edu-Ua" у якості dial-in сервера. Усі користувачі цієї послуги перенесені на новий сервер без зміни паролів. З сервісів, які залишились на старому сервері – це функція проксі-сервису та маршрутизація. Проксі-сервіс встановлюється стандартно за допомогою команди apt-get з репозіторію Ubuntu. Маршрутизація на старому сервері велась через один фізичний інтерфейс, який відповідає декільком логічним. Це не дуже вдала конфігурація, тому що це a) велике навантаження на інтерфейс, повільність роботи; b) пакети для різних логічних сегментів посилаються в одну фізичну мережу, що приводить до навантажені мережі та появі колізій; c) всі логічні сегменти на фізичному рівні залежать від надійності одного порту комутатора, якій при виході з ладу призведе до непрацездатності усіх сегментів одночасно.

Тому логічні сегменти мережі були розділені на різні фізичні середовища (див рис.2) за допомогою підключення до різних мережних інтерфейсів на новому сервері. Інтерфейс eth0 залишився основним та отримав адресу основного інтерфейсу старого серверу. Інтерфейси eth1 та eth2 отримали адреси логічних (віртуальних) інтерфейсів зі старого серверу.

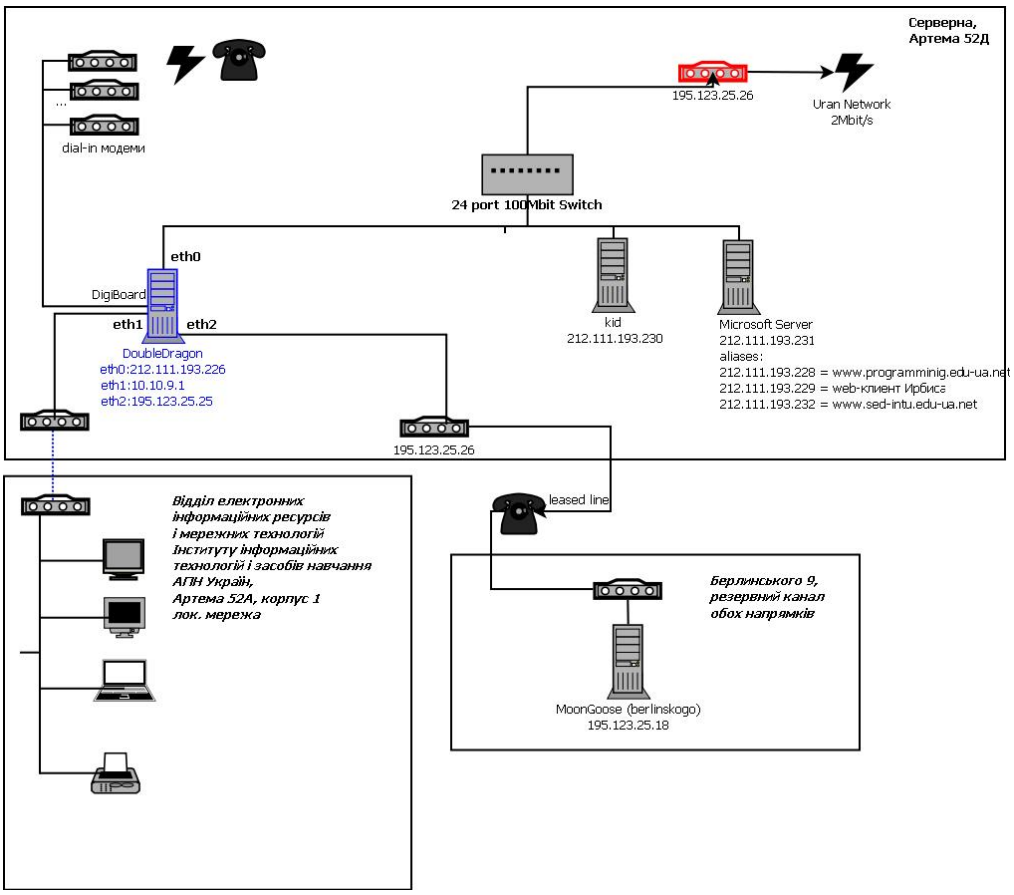


*Рис. 2. Мережа після модернізації*

## Висновки

Проведена модернізація торкнулась як фізичної складової мережі, так і логічної її структури. Фізична модернізація або апаратний апгрейд серверу дає можливість швидше та якісніше виконувати вже покладені на нього завдання, та виконувати набагато потужніші завдання. Логічна зміна в структурі мережі призвела до більш логічної її організації, що як наслідок проведеної модернізації підвищить її надійність і стійкість до аварій.

Описані у статті підхід і методика модернізації серверного забезпечення та мережної топології можуть використовуватися для модернізації та вдосконалення мереж у навчальних та наукових закладах, а також в якості навчальних матеріалів і прикладів реальних розробок при вивченні теорії і проведенні практичних занять з таких дисциплін, як інформаційно-комп'ютерні мережі, архітектура обчислювальних систем, операційні системи.

## Список використаних джерел

1. Scott Maxwell, Linux Core Kernel Commentary.– Coriolis Press, 1999
2. Ubuntu (Матеріал з Вікіпедії — вільної енциклопедії.) [Електронний ресурс]. – Режим доступу: http://uk.wikipedia.org/wiki/Ubuntu
3. HOWTO compile kernel modules for the kernel 2.6 [Електронний ресурс]. – Режим доступу: http://www.captain.at/programming/kernel-2.6/