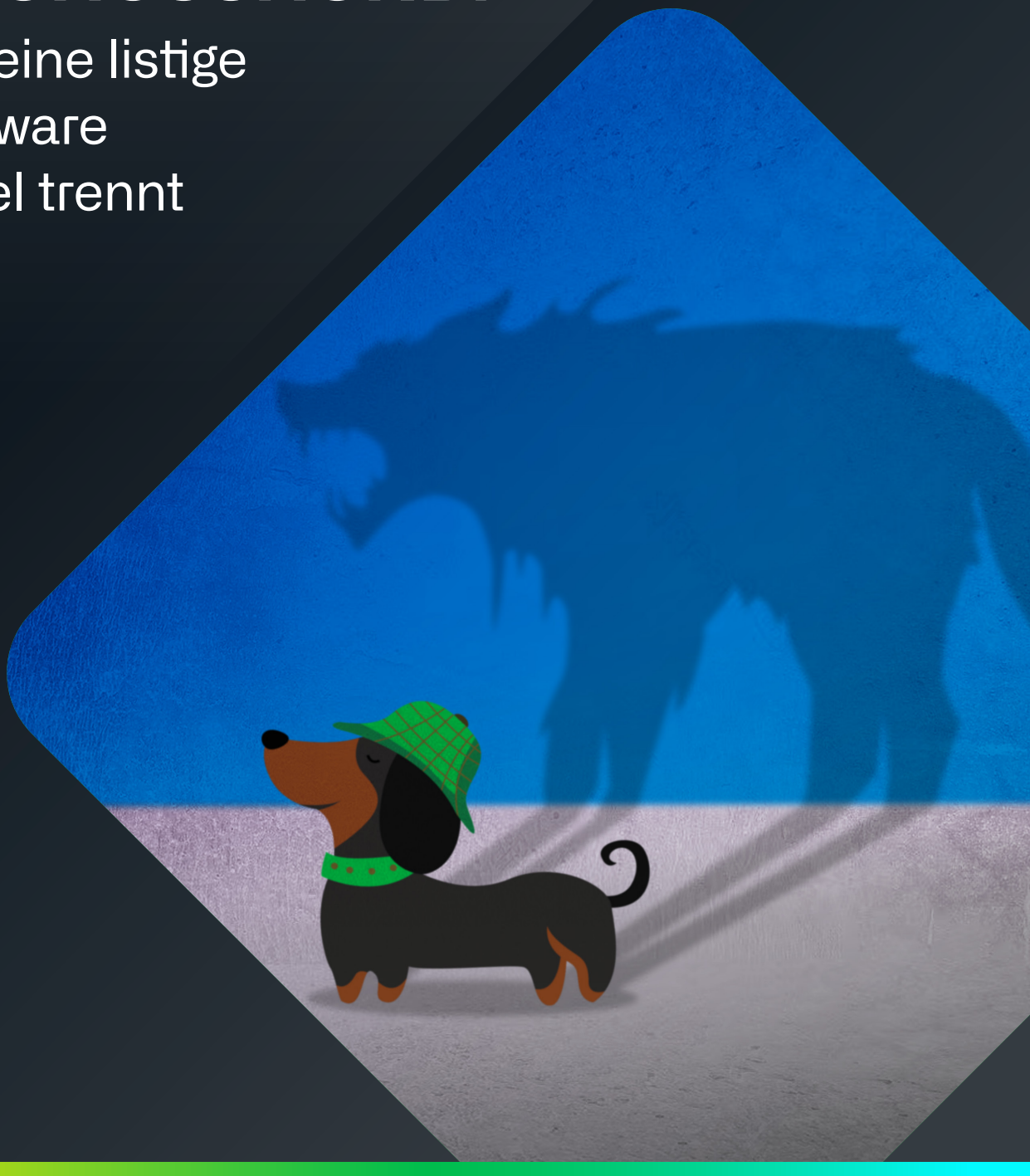
infoblox

DECOY DOG IST KEIN SCHOSSHUND:

Wie man eine listige DNS-Malware vom Rudel trennt



INHALTSVERZEICHNIS

Zusammenfassung

Decoy Dog ist ein von Infoblox entdecktes Malware-Toolkit, das das Domain Name System (DNS) verwendet, um Command and Control (C2) auszuführen. Ein kompromittierter Client kommuniziert über DNS-Abfragen mit einem Controller und erhält Anweisungen von diesem. Dieser Controller ist in einen DNS-Nameserver integriert, an den über den normalen Auflösungsprozess Abfragen übertragen werden. Wir haben die Existenz von Decoy Dog im April 2023 bekanntgegeben und am 23. April einen detaillierten Bericht über unsere ersten Erkenntnisse veröffentlicht. Die Entdeckung wurde durch die Überwachung von DNS-Daten möglich. Damalige Analysen bestätigten, dass das Toolkit auf einem Remote-Access-Trojaner (RAT) namens Pupy basiert, aber es war nicht bekannt, welche Systeme ausgenutzt wurden, wie das Toolkit bereitgestellt wurde oder ob Pupy modifiziert worden war.[1] Wir rechneten damit, dass andere in der Community mit den von uns bereitgestellten Details die kompromittierten Rechner ausfindig machen würden und somit die ganze Geschichte bekannt werden würde. Das Mysterium um Decoy Dog ist jedoch nur noch größer geworden.

Infoblox hat seit April weitere Untersuchungen zu Decoy Dog und Pupy durchgeführt. Dieser Bericht ist ihr Ergebnis. Wir haben gelernt, dass Decoy Dog ein großes Upgrade von Pupy ist, das Befehle und Konfigurationen verwendet, die nicht im öffentlichen Repository sind. Wir haben Algorithmen entwickelt, um die Kommunikation der Decoy Dog-Clients zu separieren und eine Reihe anderer Eigenschaften jedes Controllers abzuleiten. Daraus können wir mit hoher Sicherheit schließen, dass sich das Toolkit weiter verbreitet hat und unter der Kontrolle von mindestens drei Akteuren steht. Während die von uns beobachteten Aktivitäten weiterhin auf Russland und Osteuropa beschränkt bleiben, gibt es innerhalb der Controller deutliche Gruppierungen von Techniken, Taktiken und Vorgehensweisen (Techniques, Tactics, and Procedures; TTPs), die auf mehrere Akteure schließen lassen.

Jeder Decoy Dog-Akteur hat in irgendeiner Form auf unsere Enthüllungen im April reagiert, und die Abweichungen bestätigen unsere Einschätzung, dass es sich um mehrere Betreiber handelt. Unmittelbar nach der ersten Ankündigung in den sozialen Medien wurden einige Nameserver abgeschaltet. Alle verbleibenden wurden geändert, um das Verhalten zu entfernen, das wir in unserem ersten Paper hervorgehoben haben. Dies wurde jedoch je nach Controller auf unterschiedliche Weise erreicht. Eine Gruppe von Controllern begann, die Antworten auf Abfragen in Abhängigkeit vom Herkunftsland einzuschränken – eine Technik, die als Geofencing bezeichnet wird. Andere änderten ihre Antworten auf Abfragen für die Ping-Subdomain.

Ein Akteur reagierte so schnell auf unsere Enthüllung auf LinkedIn, dass wir zunächst dachten, die neuen Domains seien Copycat-Registrierungen von Sicherheitsforschern. Weitere Analysen ergaben jedoch, dass es sich dabei um Ersatzdomains handelte. Anstatt den Betrieb einzustellen, übertrug der Akteur bestehende kompromittierte Clients auf die neuen Controller. Dies ist eine außergewöhnliche Reaktion, die zeigt, dass der Akteur es für notwendig hielt, den Zugriff auf seine bestehenden Opfer aufrechtzuerhalten. Damit wurde eine klare Trennung zwischen den TTPs eines Satzes von Decoy Dog-Domains und allen weiteren geschaffen.

In den Wochen nach unserer Ankündigung waren wir überrascht, dass sich niemand gemeldet hatte, um die zugrundeliegende Malware und die Schwachstelle zu identifizieren, die Decoy Dog seine Aktivitäten erst ermöglicht hatte. Im Laufe unserer Nachforschungen wurde jedoch klar, warum die Kommunikation über ein Jahr lang unentdeckt blieb. Angriffe mit Decoy Dog laufen sehr zielgerichtet ab und jeder Controller verfügt über eine kleine Anzahl aktiver Clients. Einige Server haben über Monate hinweg durchgehend vier bis acht aktive Clients unterhalten. Während in anderen Fällen die Zahl der gleichzeitig aktiven Clients im Laufe der Zeit anstieg, lag die Gesamtzahl der betroffenen Geräte bei weniger als 100. Eine kleine Opfergruppe schließt in der Regel finanziell motivierte Akteure aus, und die

1 https://github.com/n1nj4sec/pupy

Notwendigkeit, über einen langen Zeitraum auf einem Gerät zu verharren, passt zu extrem fortschrittlichen Akteuren.

Wir konnten Teile der Decoy Dog-Kommunikationsvorgänge rekonstruieren, indem wir Signaturen aus unserem eigenen Pupy-Verkehr identifiziert haben. Wir haben einen Pupy-Server im Internet eingerichtet. In Kombination mit selektivem Reverse Engineering des Codes konnten wir DNS-Abfragen und Antworten auf bestimmte Pupy-Befehle korrelieren. Mit dieser Methode konnten wir a) feststellen, dass Decoy Dog Befehle enthält, die bei Pupy nicht vorkommen, und b) die meisten der Kommunikationsvorgänge charakterisieren. Außerdem scheinen die Decoy Dog-Akteure Pupy zu nutzen, um andere Transportschichten außer DNS für Funktionen wie den Schlüsselaustausch zu verwenden. Bedrohungsakteure betrachten dies wahrscheinlich als einen der Vorteile von Pupy als Remote-Access-Trojaner (RAT).

Die erste bekannte Bereitstellung des Decoy Dog-Toolkits fand Ende März oder Anfang April 2022 statt. Es wurde kurz darauf verkauft oder gestohlen, wie das Auftauchen eines zweiten Controllers mit anderen TTPs zeigt, der bis Mitte Mai aktiv war. Eine dritte Domain wurde im Juli 2022 registriert und strategisch bis September gealtert. Es ist möglich, dass die beiden letztgenannten Controller demselben Akteur gehören, da sie viele Merkmale teilen, darunter das Hosting im russischen IP-Raum. Allerdings weisen sie auch einige Unterschiede auf. Einige Monate später wurden zwei weitere Domains registriert, erneut mit klaren Merkmalen der vorherigen Controller. Der Akteur, der diese Domains registriert hat, hat sofort nach unserer Enthüllung Clients auf neue Domains migriert. Insgesamt überwacht Infoblox derzeit 21 Decoy Dog-Domains, von denen einige innerhalb des letzten Monats registriert und bereitgestellt wurden.

Nachdem wir durch unsere Analyse der DNS-Logs festgestellt hatten, dass sich Decoy Dog erheblich von Pupy unterscheidet, haben wir auf VirusTotal verfügbare verwandte Binärdateiproben untersucht, um zu sehen, ob die Unterschiede in den ausführbaren Dateien erkennbar waren. Das Reverse Engineering dieser Proben zeigte, dass sie zwar als Pupy erkannt wurden, jedoch viel ausgeklügelter sind als die Open-Source-Version. Die Proben umfassen a) die Möglichkeit, beliebigen Java-Code auf dem Client auszuführen, b) mehrere neue Transportmechanismen und c) neue DNS-Mechanismen zum Gewährleisten von Persistenz. Ein Mechanismus ähnelt einem traditionellen DNS-Domain-Generierungsalgorithmus (DGA) und nutzt kostenlose DDNS-Anbieter (dynamisches DNS), um sich mit sogenannten Notfall-Controllern zu verbinden. Alle Proben teilen die gleichen grundlegenden Aktualisierungen, obwohl eine der Proben über einzigartige Funktionen verfügt, die in den anderen nicht zu finden sind und mit der Verwendung von Streaming Transports zusammenhängen.

Aus noch unklaren Gründen verstößt Decoy Dog gegen die Grundprinzipien der verdeckten Kommunikation, die im Allgemeinen darauf abzielt, die Entdeckung und Wiederherstellung des Inhalts durch einen Gegenspieler zu vermeiden. Während normale Pupy-Server Kommunikationsabfragen, die von kompromittierten Clients wiederholt werden, zurückweisen, antworten Decoy Dog-Server nicht nur auf wiederholte DNS-Abfragen, sondern auf jede ordnungsgemäß formulierte Abfrage. Dieses Verhalten ähnelt Wildcard-Konfigurationen in DNS und war ein wesentlicher Faktor bei der Entdeckung von Decoy Dog durch Infoblox. Angesichts der Raffinesse von Decoy Dog spekulieren wir, dass das Wiederholungs- und Wildcard-Verhalten beabsichtigt ist. Unabhängig von der Absicht dahinter: Die weit verbreitete Wiederholung des DNS war teilweise dafür verantwortlich, dass die Branche Decoy Dog nicht als neue Malware erkennen konnte.

Aggressives Internetscans durch einen Sicherheitsanbieter führten zur erneuten Übertragung von Millionen Decoy Dog-Kommunikationsvorgängen über globale Netzwerke, wobei auch einige unserer Kunden betroffen waren. Dies wiederum führte zur unserer Entdeckung des Toolkits. Da der Anbieter nicht in der Lage war, den Datenverkehr als Malware zu identifizieren, um die Wiederholung der Abfragen zu vermeiden, wurden DNS-Verbindungen von nicht infizierten Netzwerken zu den Decoy Dog-Controllern hergestellt.

Wir sind zuversichtlich, dass kein Infoblox-Kunde infiziert wurde und dass die Abfragen an unsere Resolver allesamt das Ergebnis anomaler Anbieterscans waren. Obwohl keine unmittelbare Bedrohung für unsere Kundennetzwerke besteht, bleibt Decoy Dog ein ausgeklügeltes Toolkit mit unklarem Ursprung, das sich möglicherweise immer weiter ausbreiten wird.

Decoy Dog wurde nicht nur erst kürzlich das erste Mal „in freier Wildbahn" beobachtet, sondern ist unseres Wissens nach auch der erste Verwendungsfall der DNS-C2-Komponente von Pupy im Rahmen einer böswilligen Operation. Teilweise liegt das wahrscheinlich an der Schwierigkeit, einen Pupy-Nameserver einzurichten, wofür die Software im Repo modifiziert und das DNS korrekt konfiguriert werden muss. Die Unauffälligkeit von Pupy und Decoy Dog macht es für die Sicherheitsbranche schwieriger, sie zu erkennen und sich gegen sie zu verteidigen. Um Operationen zu stören, die diese C2-Systeme nutzen, stellen wir der Community einen Forschungsdatensatz zur Verfügung, der den von unserem eigenen Server aufgezeichneten Pupy-DNS-Verkehr und Einzelheiten über die Funktionsweise der Software enthält. Diese Dokumentation ist die erste ihrer Art und wird es anderen ermöglichen, Erkennungsalgorithmen zu entwickeln und unsere Ergebnisse zu reproduzieren.

Die Geschichte von Decoy Dog zeigt die Leistungsfähigkeit von DNS als Quelle für die Erkennung von und Reaktion auf Bedrohungen. Sie offenbart auch eine inhärente Schwäche des Malware-zentrierten Bedrohungsinformationen-Ökosystems, das die Sicherheitsbranche dominiert. Das Toolkit wurde von DNS-Bedrohungserkennungs-algorithmen entdeckt, und aktuell ist die einzige Verteidigungsmöglichkeit in diesem Fall DNS. Darüber hinaus hatten wir mehrere Controller-Domains als verdächtig gekennzeichnet und hatten sie schon bei unseren Resolvern blockiert, bevor wir erkannt haben, dass sie alle eine gängige Malware nutzten. Diese Art von Schutz, der bösartige Aktivitäten abwehrt, bevor sie erkannt werden, und oft, bevor sie operationalisiert werden, gibt es nur bei Systemen mit DNS-Erkennung- und -Reaktion.

In diesem Paper geben wir Cybersicherheitsfachleuten unser Wissen weiter, wie Pupy und Decoy Dog identifiziert werden können. Wir werden DNS C2 zwar ausführlich beschreiben, aber keine Informationen liefern, die böswilligen Akteuren helfen, Pupy einzusetzen, und wir werden auch nicht die vollständige DNS-Signatur von Decoy Dog offenlegen. Wir erklären einige Verhaltensweisen, die wir in unserem Original-Paper identifiziert haben, und gehen näher darauf ein, wie Decoy Dog sich von Pupy unterscheidet. Außerdem beschreiben wir unsere Analyse großer Mengen von Decoy Dog-DNS-Traffic, die es uns ermöglichte, die Anzahl der Clients und den Befehls-Traffic zu schätzen, ohne die Malware selbst zu besitzen oder den Nameserver zu kontrollieren. Wir beschreiben, wie sich die Decoy Dog-Proben von Pupy unterscheiden. Und schließlich erörtern wir, wie die Decoy Dog-Betreiber auf unsere Enthüllungen reagiert haben, und zeigen Gemeinsamkeiten der Controller-Untergruppen auf. Die Anhänge enthalten zusätzliche unterstützende technische Informationen.

HINTERGRUND

Infoblox entdeckte Anfang April 2023 Decoy Dog, ein C2-Toolkit („Command and Control"), das das Domain Name System (DNS) nutzt. Es basiert auf einem Open-Source-Remote-Access-Trojaner (RAT) namens Pupy[2] und transportiert verschlüsselte Kommunikation über Domainnamen-Abfragen und IP-Adressen-Antworten zwischen Clients und Servern oder Controllern. Die Entdeckung ergab sich aus Algorithmen, die Passive-DNS-Abfragen an Infoblox-Resolver auf anomales Verhalten hin überwachen. Über Sicherheitsgeräte in einer kleinen Anzahl von Kundennetzwerken wurden Abfragen für Decoy Dog-Domains getätigt. Diese Abfragen erstellten eine Signatur, die anhaltendem, unauffälligem Malware-Beaconing entspricht. Die Überprüfung der Aktivitäten durch Mitarbeiter ergab Alarmierendes:

2 https://malpedia.caad.fkie.fraunhofer.de/details/win.pupy

Obwohl das DNS eindeutig als vertraulicher Kommunikationskanal genutzt wurde, waren die Domains in den öffentlich zugänglichen Bedrohungsdaten nicht als C2 bekannt. Einige waren von Online-Reputationsprüfern sogar als „seriös“ eingestuft worden. Wir haben am 13. April einen Satz Domains veröffentlicht, um der Community zu helfen, den Traffic zu blockieren und die Art der Kompromittierung zu identifizieren.

Während unserer ursprünglichen Recherche hat Infoblox eine eindeutige DNS-Signatur identifiziert, die von der Pupy-Software unabhängig war. Die Akteure hatten ihr C2-System auf eine sehr spezifische Art und Weise bereitgestellt und genutzt. Aus diesem Grund haben wir Decoy Dog als eigenes Toolkit eingestuft. Weltweit teilte nur eine kleine Anzahl von Domains diese Signatur – bei allen handelte es sich um Decoy Dog-Nameserver.

Am 23. April haben wir einen Teil der Signatur, eine erste Analyse des Passive DNS und eine Teilmenge der Controller-Domains in unserem Bericht „Dog Hunt: Finding Decoy Dog Toolkit in Anomalous DNS Traffic“[3] veröffentlicht, in dem wir ein bestimmtes Verhalten von Pupy hervorheben, bei dem es eine Reihe von localhost-Antworten auf Abfragen für bestimmte Subdomains ausgegeben hat, die „ping“ enthielten. Der Bericht beschrieb auch eine Reihe von Trends innerhalb der DNS-Kommunikation, die wir damals nicht vollständig erklären konnten. Insbesondere haben wir überraschende Muster innerhalb der IP-Adressen festgestellt, die in Antworten ausgegeben wurden, sowie die Tatsache, dass die Server auf wiederholte Abfragen geantwortet haben, was für ein System für verdeckte Kommunikation untypisch ist.

Nach diesen Ankündigungen haben sich zahlreiche Mitglieder der Cybersicherheitscommunity, darunter Anbieter und andere Unternehmen, an uns gewandt. Viele von ihnen hatten entsprechenden Datenverkehr in ihren eigenen Netzwerken oder in den Netzwerken ihrer Kunden festgestellt, aber niemand hatte kompromittierte Geräte identifiziert oder den Umfang der Aktivitäten erkannt. Einige dieser Unternehmen stellten uns Informationen zur Verfügung, die es uns ermöglichten, das DNS in unseren eigenen Netzwerken zu isolieren und nachzuweisen, wie es generiert worden war. Andere halfen dabei, den Umfang der Aktivitäten zu bestätigen und Hypothesen zu testen. Diese informelle Zusammenarbeit war äußerst hilfreich und wir bedanken uns dafür.

Der Einfachheit halber verwenden wir den Begriff Pupy in diesem Paper speziell für Pupy DNS C2 und nicht für Pupy im Allgemeinen.

Pupy

EINE SELTENE RASSE

Pupy ist ein Open-Source-Post-Exploitation-RAT, der über ein komplexes modulares Transportsystem verfügt.[4] Während die primäre Pupy-Codebasis 2015 auf GitHub verfügbar gemacht wurde, wurde der DNS-C2-Mechanismus erst 2019 hinzugefügt. Dieses Paper ist die erste öffentliche Dokumentation von Pupy C2. Darüber hinaus stellen wir auf GitHub einen Datensatz zur Verfügung, mit dem andere unsere Arbeit reproduzieren und Abwehrmechanismen für die Zukunft entwickeln können.

Wenngleich Pupy Open Source ist, wird das DNS-C2-Protokoll selten verwendet. Wir konnten seine Verwendung außerhalb von Decoy Dog „in freier Wildbahn“ bisher nicht identifizieren.[5] Über unsere eigenen Resolver, die für Unternehmen auf der ganzen Welt

3 https://blogs.infoblox.com/cyber-threat-intelligence/cyber-threat-advisory/dog-hunt-finding-decoy-dog-toolkit-via-anomalous-dns-traffic/

4 https://github.com/n1nj4sec/pupy

5 Der Ausdruck „in freier Wildbahn“ gehört zum Jargon der Cybersicherheitsbranche und steht für die operative Bereitstellung außerhalb des Rahmens isolierter Penetrationstests oder -forschungen.

im Einsatz sind, haben wir in der Vergangenheit keine Beweise für die Verwendung von Pupy DNS C2 gefunden. Für die ersten sechs Monate des Jahres 2023 haben wir innerhalb des globalen pDNS unter Verwendung von DNS-Detektoren, die wir für Pupy entwickelt haben, keine Verwendung der Software außerhalb von Decoy Dog festgestellt. Und schließlich haben wir privat bei einer Vielzahl von Anbietern nachgefragt, von denen ebenfalls keiner die Verwendung gesehen hat. In den Fällen, in denen die Verwendung von Pupy durch APT-Akteure (Advanced Persistent Threat) gemeldet wurde, wurden die DNS-C2-Komponenten offenbar nicht eingesetzt.[6]

Die seltene Verwendung von Pupy ist wahrscheinlich zumindest teilweise auf die Schwierigkeiten bei der Bedienung des Systems zurückzuführen. Die Einrichtung der Pupy-Kommunikation über das globale DNS ist nicht einfach. Sie erfordert die korrekte Konfiguration des Nameservers und eine Modifizierung des Codes im GitHub-Repository. Darüber hinaus gibt es im DNS Komplexitäten, die abhängig vom rekursivem Resolver variieren und von der Pupy-Software nicht korrekt verarbeitet werden. Diese Herausforderungen haben wahrscheinlich die Akzeptanz sowohl bei Red Teams als auch Hackern behindert, im Gegensatz zu Fällen beliebterer Tools wie Cobalt Strike, die wir ziemlich häufig sehen.[7]

Obwohl Pupy DNS C2 aktuell selten eingesetzt wird, verbreitet sich die Verwendung von Decoy Dog, und die Wahrscheinlichkeit, dass Cybersicherheitsfachleute Pupy in irgendeiner Form begegnen, wächst. Um bei der Vorbereitung der Community mitzuhelfen, führte Infoblox umfangreiche Untersuchungen sowohl zu Decoy Dog als auch zu Pupy durch. Infoblox hat online einen Pupy-Server bereitgestellt, um dessen Verhalten mit dem von Decoy Dog zu vergleichen. Anschließend haben wir Paketdaten (pcap) und Passive-DNS-Protokolle von Infoblox-Resolvern erfasst. Wir haben unsere Pupy-Bereitstellung in Verbindung mit selektivem Reverse Engineering des Codes verwendet, um die einzigartige Beschaffenheit von Decoy Dog besser zu verstehen. In diesem Abschnitt erläutern wir die für unsere Forschung relevanten Bestandteile von Pupy. Der Einfachheit halber beschränken wir dieses Paper auf die Kommunikation mit IPv4-Antworten (A-Eintrag), obwohl Pupy, wenn verfügbar, IPv6-Antworten (AAAA) nutzt. Die in diesem Paper beschriebene Abfragecodierung ist die aktuelle Standardeinstellung für Pupy, Version 2 (sofern nicht anders angegeben).[8]

WIE PUPY FUNKTIONIERT

In unserem vorherigen Paper haben wir einen Überblick über Pupy gegeben und einige ungewöhnliche Eigenschaften von Decoy Dog hervorgehoben.[9] In diesem Paper werden wir uns eingehender mit dem Pupy-Kommunikationsprotokoll befassen, um seine Verbindungen zu Decoy Dog zu demonstrieren und zu zeigen, wie passiv erfasstes Pupy-DNS ausgenutzt werden kann, um eine laufende Operation zu verstehen.

Pupy ist darauf ausgelegt, eine kontinuierliche Kommunikation zwischen infizierten Clients und dem Server bereitzustellen, sodass die Verbindung bereits hergestellt ist, wenn der Akteur remote auf den Client zugreifen möchte. Der Akteur kann verbundene Clients überwachen und selektiv steuern, um eine Vielzahl von Aktionen durchzuführen. Das

---

6 https://www.volexity.com/blog/2022/06/15/driftingcloud-zero-day-sophos-firewall-exploitation-and-an-insidious-breach/

7 https://www.esecurityplanet.com/threats/how-cobalt-strike-became-a-favorite-tool-of-hackers/

8 Eine frühere Version von Pupy C2 enthielt keine Host-Informationen in jeder Abfrage. Wir wissen jetzt, dass Decoy Dog Version 3 des Clients ist, aber die Abfragecodierung scheint dieselbe zu sein wie bei Version 2.

9 https://blogs.infoblox.com/cyber-threat-intelligence/cyber-threat-advisory/dog-hunt-finding-decoy-dog-toolkit-via-anomalous-dns-traffic/

DNS wird nur für die C2-Kommunikation verwendet. Alle signifikanten Daten, die vom Client exfiltriert werden, werden über eine der vielen anderen von Pupy angebotenen Transportoptionen gesendet. Infolgedessen ist der Pupy-DNS-Client darauf beschränkt, Check-ins beim Controller durchzuführen, Befehle zu bestätigen und Systeminformationen bereitzustellen, neben einer Handvoll anderer Aufgaben. Zwischen der Bearbeitung von Befehlen vom Server ruht der Client.

Die DNS-Kommunikation wird vom Client initiiert und verwaltet. Der Client sendet Abfragen über seinen normalen DNS-Auflösungspfad oder via DNS über HTTPS (DoH), wenn dieses aktiviert und verfügbar ist.[10] Der Controller sendet Befehle als Antwort auf Client-Abfragen in Form von verschlüsselten IP-Adressen. Jede Abfrageantwort stellt eine vollständige Kommunikation dar, was bedeutet, dass weder der Client noch der Server Daten für einen einzelnen Befehl auf zwei DNS-Abfragen aufteilen kann. Dieses Protokoll unterscheidet sich von herkömmlichen DNS-Tunneling-Systemen wie Iodine[11], bei denen der Client eine Sitzung über DNS einrichtet, die die Rekonstruktion mehrerer Pakete an beiden Enden umfassen kann, um die Kommunikation zu verarbeiten. Der Client ist verpflichtet, die meisten Befehle zu bestätigen, und der Server antwortet auf jede gültige Client-Abfrage entweder mit Befehlen oder mit einer Bestätigung. Das Client-Vokabular ist extrem begrenzt. Es verfügt über neun Arten von Abfragen, mit denen es Sitzungen verwaltet, Befehle bestätigt, Systeminformationen sendet und Schlüssel einrichtet. Benutzerdefinierte Befehle können durch das Schreiben zusätzlicher Funktionen hinzugefügt werden, erfordern jedoch ein umfassendes Verständnis der Software.

Bei der Reaktivierung fragt der Client den Server auf eine von zwei verschiedenen Arten ab, je nachdem, ob ein gemeinsamer Schlüssel eingerichtet wurde. Diese Abfrage versorgt den Server mit aktuellen Informationen über das System und den Status des Pupy-Clients oder führt eine einfache Abfrage durch, die dazu dient, eine neue verschlüsselte Sitzung zu initiieren. Es ist zwar möglich, verschlüsselte Sitzungen zu deaktivieren, doch dies ist nicht die Standardeinstellung und wurde bei Decoy Dog bisher nicht beobachtet. Als Antwort bestätigt der Controller entweder die Abfrage, fordert den Client auf, einen Schlüsselaustausch durchzuführen oder sendet neue Befehle. Sobald der gesamte Befehlssatz abgeschlossen wurde, geht der Client für das festgelegte Intervall in den Ruhemodus, standardmäßig für 60 Sekunden. Dieser Vorgang wird wiederholt, während der Client läuft wird. Eine allgemeine Übersicht über die Kommunikation zwischen Pupy-Client und -Server ist in Abbildung 1 dargestellt. Eine detailliertere Ansicht des Client-Prozesses finden Sie in Anhang A.

10 Pupy nutzt standardmäßig Quad9-Server für DoH.

11 https://github.com/yarrick/iodine

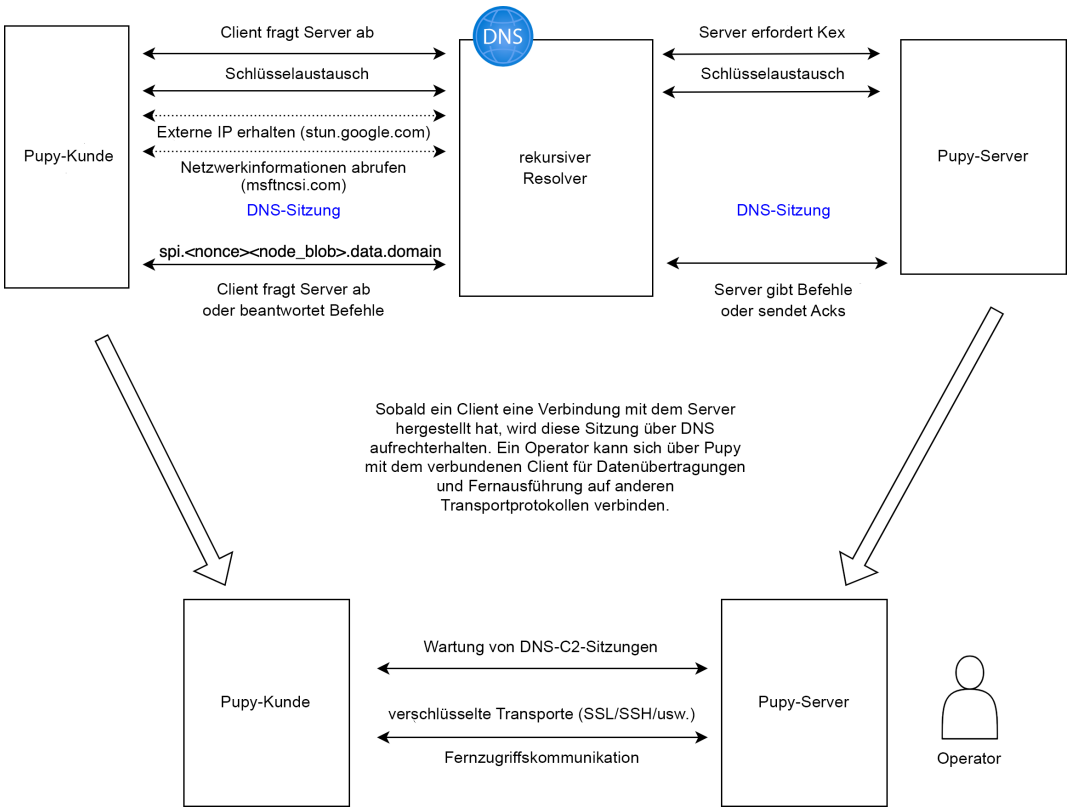


Abbildung 1. Ein allgemeiner Überblick über die Pupy-Kommunikation

Der Pupy-Akteur interagiert mit Clients über das Controller-Befehlszeilenprogramm. Wenn der Client den Controller kontaktiert, werden alle Befehle in der Warteschlange in der DNS-Antwort codiert. Der Betreiber stellt eine Verbindung auf einem geöffneten Client-Port her und gibt die Transportschicht an, die für die Exfiltration verwendet werden soll. Die Server-DNS-Kommunikation ist immer noch ziemlich eingeschränkt, jedoch umfangreicher als die des Clients. Es gibt eine Vielzahl von Befehlen, die zu einer einzigen Antwort an den Client zusammengefasst werden können. Wenngleich der Client den Kommunikationsaustausch initiiert, ist der Server für die Gewährleistung der Sicherheit der Kommunikation verantwortlich. Dazu erzwingt er sogenannte Sitzungen mit jedem Client, die dazu dienen, die Codierungsschlüssel zu rotieren. Dieser Ablauf wird im nächsten Abschnitt beschrieben.

INITIIEREN EINER SITZUNG

Pupy erfordert, dass eine verschlüsselte Sitzung zwischen dem Client und dem Controller eingerichtet wird, bevor die Befehle des Akteurs übertragen werden. Diese Sitzung läuft ab, sobald die Kommunikation mit dem Client unterbrochen wird, und kann aus anderen Gründen, einschließlich Fehlern bei der DNS-Abfragedecodierung oder einem Neustart des Clients, zur Erneuerung gezwungen werden. Sitzungen werden durch das Vorhandensein eines Sicherheitsparameterindex-Labels (SPI) in der Abfrage identifiziert und mit einem vorübergehenden gemeinsamen Schlüssel verschlüsselt. Da die Kommunikationsdetails von einer Vielzahl von Faktoren abhängen, einschließlich der Frage, ob der Client schon einmal eine Verbindung zum Server hergestellt hat, kann das genaue Protokoll für die Sitzungsinitialisierung variieren, was zu Abweichungen in den beobachteten DNS-Austauschvorgängen führt. Der typische Austausch sieht jedoch wie folgt aus:

- Der Client führt entweder ohne eine bestehende Sitzung oder mit einer abgelaufenen Sitzung einen Check-in beim Server durch (Abfrage 1).

- Der Server antwortet mit einem Befehl, der einen Schlüsselaustausch- und Client-System-Informationen erfordert.[12]
- Der Client bestätigt die Anforderung von Systeminformationen (Abfrage 2).
- Der Client generiert mithilfe eines auf elliptischen Kurven basierenden Algorithmus ein zufälliges Schlüsselpaar nach Public-Key-Verschlüsselungsverfahren und sendet es an den Server. Der Server tut dasselbe und antwortet mit seinem neuen Schlüssel (Abfrage 3).
- Der Client und der Server nutzen diesen Austausch, um einen neuen gemeinsamen Sitzungsschlüssel einzurichten, der zum Verschlüsseln von Paketen nach AES verwendet wird, und erstellen auch den SPI zur Identifizierung der Sitzung.
- Der Client sammelt Informationen über sein Netzwerk, einschließlich seiner externen IP-Adresse, mithilfe zusätzlicher DNS-Abfragen an andere Dienste.
- Der Client überträgt diese Informationen mithilfe des gemeinsamen Codierungsschlüssels und signalisiert das Vorhandensein einer aktiven Sitzung unter Angabe des SPI in der Abfrage (Abfrage 4).
- Der Client sendet zusätzliche Systemstatusinformationen (Abfrage 5).

Der gemeinsame Schlüssel und der SPI werden in der Regel nach drei Abfragen erstellt, obwohl der Schlüsselaustausch technisch gesehen eine einzige Abfrage und Antwort darstellt. Während einer Sitzung wird jede Abfrage und Antwort mit diesem gemeinsamen Schlüssel verschlüsselt. Für die Verschlüsselung wird außerdem eine vom Client generierte 32-Bit-Nonce verwendet, die sich bei jeder Abfrage ändert. Wenn eine neue Sitzung eingerichtet wird, werden die Schlüssel neu generiert, aber der Client-Nonce-Wert bleibt bestehen, während der Client aktiv ist. Dieser Sachverhalt wird im folgenden Abschnitt näher erläutert.

ABFRAGE-CODIERUNG

Der Client generiert Abfragen, die verschlüsselte Kommunikation an den Server enthalten. Dazu können Schlüsselaustauschinformationen oder eine Antwort auf Befehle vom Server gehören. In jeder Abfrage können maximal 52 Bytes an übertragenen Daten kommuniziert werden. Neben den übermittelten Daten umfasst jede Abfrage folgende Bestandteile:

- nonce, ein ansteigender 4-Byte-Wert, der vom Client generiert wird
- version, ein 1-Byte-Wert, der die Pupy-DNS-C2-Version angibt
- cid, ein 4-Byte-Wert aus der Konfiguration des Clients, der zufällig generiert wird, wenn der Client erstellt wird
- iid, ein 2-Byte-Wert, der die untersten 16 Bits des Pupy-Client-Prozesses enthält
- node id, ein 6-Byte-Wert vom Client, in der Regel die MAC-Adresse des Geräts
- Optional: SPI, ein 4-Byte-Wert, der während des Schlüsselaustauschs generiert wird und in Abfragen enthalten ist, die eine Sitzung auf dem Server für einen bestimmten Client darstellen

Jede Client-Abfrage enthält diese 13 Bytes an Client-Informationen sowie eine 4-Byte-Prüfsumme über die zugrundeliegende Nutzlast. Die zugrundeliegende Nutzlast ist verschlüsselt und besteht aus einer Reihe von Befehlen und zugehörigen Daten.

12 Dieser Vorgang erfolgt typischerweise in Form von zwei Befehlen, die auf Serverseite als „Policy“ und „Poll“ bezeichnet werden.

Der Client verschlüsselt und codiert Daten, die als vollständig qualifizierter Domainname (FQDN) an den Server übertragen werden sollen. Dieser FQDN wird im DNS-Protokoll als Abfragename (qname) bezeichnet. Der gesamte Prozess, unten dargestellt in Abbildung 2, umfasst das Verschlüsseln, Anordnen und Codieren sowohl der übertragenen Daten als auch zusätzlicher vom Server benötigten Informationen. Die Funktionsweise ist wie folgt:

- Den zu übertragenden Daten werden hostspezifische Informationen angehängt.
- Diese zusammengesetzte Byte-Zeichenfolge wird mit einem gemeinsamen symmetrischen Schlüssel und der aktuellen Nonce verschlüsselt.
- Die ersten verschlüsselten Bytes der übertragenen Daten, bis zu 35 Bytes, werden codiert und für das erste bzw. ganz rechte Label des qname verwendet.
- Den restlichen verschlüsselten Bytes, die bis zu 17 Bytes der übertragenen Daten enthalten können, wird der aktuelle Nonce-Wert vorangestellt und sie werden für die Erstellung des zweiten Labels des qname codiert.
- Wenn der Sicherheitsparameterindex (SPI) im Client vorhanden ist, wird er codiert und im dritten oder ganz linken Label des qname verwendet. Dieser Wert wird nach einem Schlüsselaustausch mit dem Server festgelegt.
- Die Nonce wird um die Länge der verschlüsselten Daten im Client erhöht und so für die nächste Abfrage verwendet.

Die Codierung von verschlüsselten Bytes zu einem Domainnamen-Label wurde in unserem vorherigen Paper beschrieben. Dafür wird eine angepasste Karte in Kombination mit 32-Bit-Codierung verwendet, um sicherzustellen, dass das Endergebnis ein gültiger Domainname ist. Die zugrundeliegende Datennutzlaststruktur wird in Anhang B beschrieben.

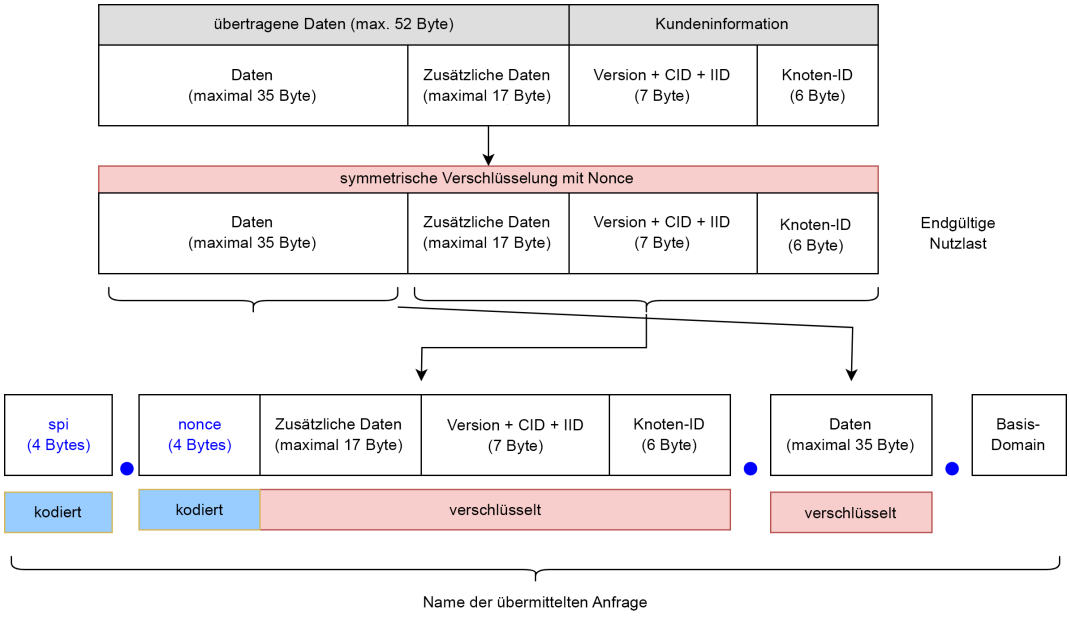


Abbildung 2. Der Prozess zum Konvertieren von Daten vom Client in einen qname für eine DNS-Abfrage. Die Basisdomain ist der Domainname des Pupy-Servers.

Bei der Verschlüsselung von DNS-Abfragen verwendet Pupy standardmäßig AES. Wenn ein gemeinsamer Schlüssel mit dem Client eingerichtet wurde, verwendet Pupy diesen, um die komplette Bytezeichenfolge symmetrisch zu verschlüsseln; andernfalls verwendet es den eingerichteten öffentlichen Schlüssel. In beiden Fällen wird die aktuelle Nonce auch in der Verschlüsselung verwendet, um sicherzustellen, dass die codierte Abfrage einzigartig ist, selbst wenn die zugrundeliegenden übertragenen Daten über mehrere Abfragen hinweg gleich bleiben. Das ist ein Standardmechanismus zum Schutz vor

kryptografischen Angriffen. Infolgedessen kann der abgefragte Domainname decodiert werden, um die verschlüsselten Daten anzuzeigen, aber die verschlüsselten Daten können ohne den Schlüssel nicht entschlüsselt werden. Der Nonce-Wert wird mit einem zufälligen 32-Bit-Wert initialisiert und bei jeder Abfrage um die Länge der Nutzlast erhöht.

Wenn der Pupy-Nameserver eine Abfrage erhält, decodiert er den Domainnamen, um den SPI-Wert, die Nonce und die verschlüsselte Nutzlast anzuzeigen. Um sicherzustellen, dass er gültige Client-Kommunikation empfängt, prüft der Server, ob der SPI gültig ist, wenn er vorhanden ist, und ob die Nonce größer ist als die letzte, die für den Client aufgezeichnet wurde. Er führt verschiedene weitere Überprüfungen der Daten durch, einschließlich einer Überprüfung der Versionsnummer, die in der Nutzlast verschlüsselt ist. Wenn eine dieser Überprüfungen fehlschlägt, wird ein Fehler an den Client ausgegeben.

Insbesondere beantwortet Pupy nicht dieselbe Abfrage zweimal, und jeder unmodifizierte Pupy-Server reagiert auf eine Abfrage, die er bereits in der Vergangenheit erhalten hat, mit der Antwort NXDOMAIN (nicht existente Domain). Wir haben dieses Verhalten mit unserem eigenen Pupy-Server überprüft, indem wir versucht haben, einen zuvor abgefragten Domainnamen abzufragen. Das ist wichtig, da eine Eigenschaft von Decoy Dog darin besteht, dass es auf wiederholte DNS-Abfragen mit Antworten reagiert, die dem Pupy-C2-Protokoll entsprechen.

Da die DNS-Abfrage eine umkehrbare Codierung der Nonce enthält und die Nonce bei jeder Abfrage um die Nutzlastlänge erhöht wird, können wir Abfrage-Threads im Zusammenhang mit einem bestimmten Client rekonstruieren. Wie wir später in diesem Paper sehen werden, können wir diese Rekonstruktion bei gegebener Passive-DNS-Erfassung für eine Pupy- oder Decoy Dog-Domain verwenden, um die Anzahl der Clients sowie in bestimmten Fällen die Art der Kommunikation zu schätzen.

SPEZIELLER UMGANG MIT DOMAINNAMEN

Beim Empfang einer Abfrage analysiert der Server den Abfragenamen und bestimmt, ob er mit der geeigneten Struktur für ein verschlüsseltes Paket von einem Client übereinstimmt. Es gibt einige Sonderfälle, in denen eine spezifische Verarbeitung erfolgt. Abgesehen von diesen Sonderfällen wird jede Abfrage abgelehnt, die nicht dem erwarteten Format entspricht. Einer dieser Sonderfälle sind Ping-Abfragen, die wir in unserem vorherigen Paper beschrieben haben. Eine Abfrage für eine Subdomain pingN, wobei N eine Ganzzahl ist, führt zur Ausgabe einer Folge von localhost-Antworten mit der Länge N. Eine Abfrage für ping selbst führt zur Ausgabe von 15 solcher Antworten, und eine Abfrage für die Basisdomain führt zur Ausgabe einer einzelnen localhost-Antwort, also 127.0.0.1.

Außerhalb der Ping-Abfragen kann der Server so konfiguriert werden, dass er auf Single-Label-Abfragen mit einer einzigen IP-Adresse antwortet. Der Zweck dieser Funktion ist unbekannt und sie scheint im Client nicht verwendet zu werden. Sie wird im Quellcode als DNS-Aktivierungsanforderung bezeichnet. Diese Fähigkeit ist nicht dokumentiert und um sie zu nutzen, müsste der Akteur verstehen, wie die Server-Software funktioniert.

Die besondere Behandlung von Single-Label-Subdomains wird durch die Konfiguration von „Aktivierungs“-Einträgen erreicht, bei denen es sich um Schlüssel-Wert-Paare von Zeichenketten handelt. Der Wert wird dann zusammen mit dem privaten Schlüssel des Servers verwendet, um eine Antwort-IP-Adresse zu erstellen. Diese Antwort wird mit einer One-Way-Hash-Funktion erstellt und kann nicht umgekehrt werden. Der Hash unterscheidet zwischen Groß- und Kleinschreibung und ist definiert als

MD5(subdomain_label + activation_value + private_key)

ANTWORTCODIERUNG

Wenn der Server eine Abfrage von einem Client erhält, wird von ihm eine Decodierung und Entschlüsselung durchgeführt, die Ergebnisse überprüft und die Client-Daten verarbeitet. Insbesondere muss eine ordnungsgemäß formatierte Client-Kommunikation zwei oder drei Label enthalten, entsprechend der Beschreibung weiter oben im Abschnitt zur Abfragecodierung. Der Server stellt dann eine Antwort an den Client zusammen, die einen oder mehrere Befehle enthält. Er kann zwar sowohl IPv4- (A) als auch IPv6- (AAAA) Abfragen ausgeben, aber der Einfachheit halber beschränken wir unsere Beschreibung auf IPv4-Abfragen (A).

Die Serverantwort ist eine verschlüsselte binäre Zeichenfolge, die dann in einen oder mehrere A-Einträge codiert wird.[13] Der Prozess für diese Codierung ist in Abbildung 3 unten dargestellt. Die maximale Anzahl von Bytes in der Antwort beträgt 64, die in 3-Byte-Segmenten codiert sind, was maximal 22 IPv4-Adressen in der Antwort ergibt.

- Im ersten Schritt berechnet der Server die Länge der Antwort und stellt dies den Antwortdaten voran. Dann hängt er zufällige Bytes an, um eine zusammengesetzte Zeichenfolge zu erstellen, die ein Vielfaches von 3 Bytes lang ist.[14] Wir nennen diese zusammengesetzte Zeichenfolge die Nutzlast.
- Im zweiten Schritt werden IPv4-Adressen iterativ aus 3-Byte-Segmenten der Nutzlast erstellt. Jede IPv4-Adresse wird durch einen 32-Bit-Wert dargestellt, wobei Bit 0 das höchstwertige Bit ist.
- Die 3 höchstwertigen Bits jeder Adresse sind zufällig.
- Jedes Segment verfügt über einen Index, der es dem Client ermöglicht, die Daten beim Empfang zu ordnen; dieser wird durch 5 Bits dargestellt. Er befindet sich in den Bits 3–7 des Ergebnisses.
- Das Nutzlast-Segment befindet sich in den Bits 8–30, wodurch das höchstwertige Bit des Nutzlastsegments gezwungenermaßen das niedrigstwertige Bit des ersten Oktetts in der IPv4-Adresse ist.
- Schließlich ist das niedrigstwertige Bit, Bit 31, ein Prüfbit, das im Nutzlastsegment generiert wird. Aufgrund der Beschaffenheit dieser Prüfsumme ist dieses Bit in 75 % der IPv4-Adressen 1.
- Die resultierende 32-Bit-Zeichenfolge wird als IPv4-Adresse interpretiert und an die Antwort angehängt.

13 Eine Reihe von Befehlen wird zusammengestellt und dann vor der Codierung mit einem gemeinsamen Schlüssel und der aktuellen Nonce verschlüsselt, sofern ein Schlüsselaustausch abgeschlossen wurde. Andernfalls wird der private Schlüssel des Servers zusammen mit dem Nonce verwendet, um die Daten mit einem Elliptische-Kurven-Algorithmus mit öffentlichem Schlüssel zu verschlüsseln.

14 Im Code ist dieser Vorgang komplizierter, hat aber das gleiche Ergebnis.

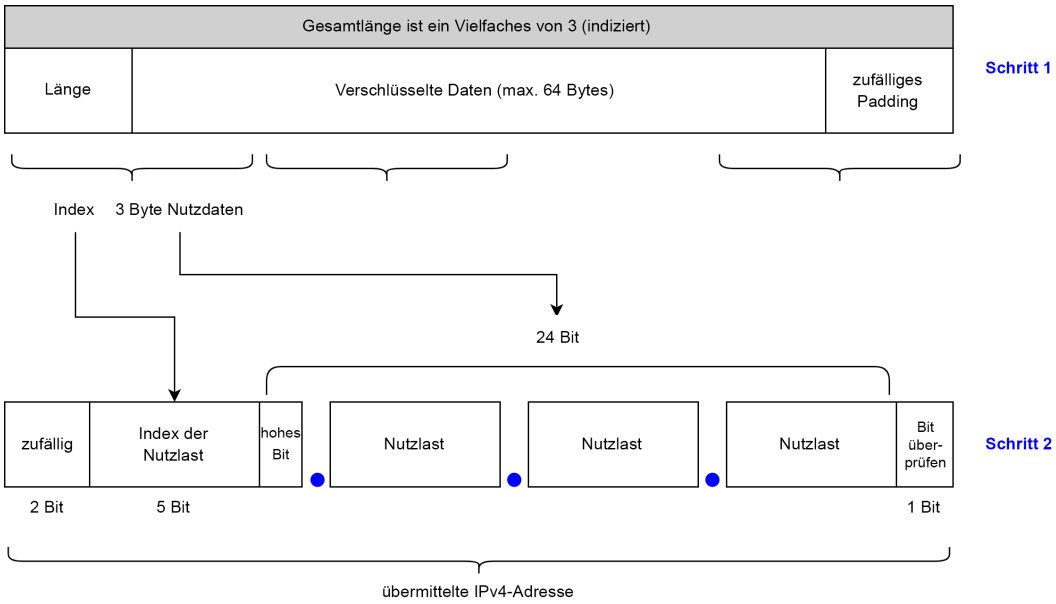


Abbildung 3. Pupy-Servercodierung einer IPv4-Antwort auf eine Client-Abfrage. Die Daten werden in eine Reihe von IPv4-Adressen codiert, wobei 3 Bytes der Nutzlast in jeder Adresse verwendet werden.

In unserem vorherigen Paper haben wir festgestellt, dass Decoy Dog eine überraschende Verteilung von IPv4-Antworten aufweist. Wir wissen jetzt, dass dies ein Artefakt der Pupy-Antwortcodierung war. Die Verwendung von drei zufälligen Bits und einem inkrementierenden Index als die höchstwertigen 7 Bits des ersten Oktetts in jeder Antwort garantiert, dass die resultierenden IPv4-Adressen in bestimmten Bereichen liegen und dass diese Bereiche direkt mit der Anzahl der Antworten in der Reaktion korrelieren, die wiederum von der Größe der Daten bestimmt wird, die an den Client übertragen werden. Insbesondere liegt die erste IP-Adresse immer im Bereich 64.0.0.0/8, 128.0.0.0/8 oder 192.0.0.0/8.

Jedes Mal, wenn der Index erhöht wird, werden die Auswahlmöglichkeiten für das erste Oktett der IP-Adresse um zwei verschoben. Konkret bedeutet das:

- Die erste IP-Adresse beginnt mit 64, 128 oder 192, da der Index 0 ist und die Länge maximal 64 beträgt. Daher werden nur die höchstwertigen 3 Bits in der ersten IP-Adresse der Antwort gesetzt.
- Die zweite IP-Adresse beginnt mit 66, 67, 130, 131, 194 oder 195, weil der Index 1 ist, was 2 zu den zufällig generierten 3 höchstwertigen Bits addiert, und das höchstwertige Bit der Datennutzlast kann eine 0 oder eine 1 sein.
- Die dritte IP-Adresse beginnt mit 68, 69, 132, 133, 196 oder 197 usw.

Das Ergebnis dieses Algorithmus für eine zunehmende Anzahl von Antworten wird unten in Abbildung 4 gezeigt. Wir verwenden hier eine Hilbert-Karte, um zu demonstrieren, wie das erste Oktett der IP-Adressen mit der Gesamtzahl der Antworten für 3, 12 und 15 Antworten korreliert.

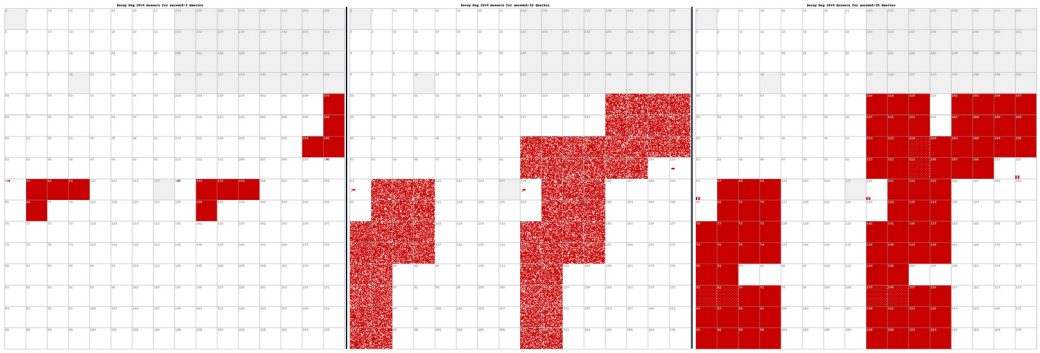


Abbildung 4. Hilbert-Karten zeigen die Verteilung von IPv4-Adressen in Pupy-Antworten, die jeweils 3, 12 und 15 Antworten enthalten.

Die Struktur der IPv4-Adressen ermöglicht es jedem, der die vollständige Antwort beobachtet, die übertragenen Daten zu rekonstruieren. Obwohl diese Daten verschlüsselt sind, kann ein Profiling der Antworten mithilfe der Längen- und Zeitreihenanalyse durchgeführt werden. Diese Art der Analyse kann Informationen über die Kommunikation enthüllen, wie wir später in diesem Paper sehen werden.

PASSIVE DATENANALYSE

Auch wenn Pupy-Kommunikation stark verschlüsselt ist, sind die Informationen, die zum Entschlüsseln und Verfolgen der Pakete erforderlich sind, umkehrbar codiert. Wenn die DNS-Abfragen und -Antworten erfasst werden, können sie aggregiert analysiert werden, um Informationen über die Pupy-Bereitstellung und die Clients abzuleiten. Die passive Erfassung von DNS-Daten, die allgemein als Passive DNS (auch pDNS) bezeichnet wird, erfolgt an vielen Stellen im Internet, einschließlich Unternehmens-Resolvern, öffentlichen rekursiven Resolvern sowie Root- und TLD-Servern. In den folgenden Abschnitten zeigen wir, wie die Passive-DNS-Erfassung bei Pupy-Abfragen genutzt werden kann, um Informationen über die Kommunikation zu erhalten.

Wir können via Passive DNS einiges an Informationen über einen Pupy-Controller und seine Clients wiederherstellen, insbesondere folgende:

- Die ungefähre Anzahl aktiver Clients zu einem beliebigen Zeitpunkt
- Welche Arten von Austausch zwischen dem Server und den Clients stattfinden
- Signaturen der Bereitstellung, etwa das Client-Ruheintervall
- Eine Zeitleiste der Client-Schlüsselaustauschvorgänge und der Gesamtaktivität

Wir haben diese Techniken verwendet, um Datenverkehr von unserem eigenen Server sowie Decoy Dog-Servern zu analysieren. Dadurch konnten wir Einblicke gewinnen, wie sehr Decoy Dog Pupy ähnelt und in welchem Umfang die Server einander gleichen. Letztendlich ermöglichten uns diese Techniken das Profiling jeder Decoy Dog-Bereitstellung. Technische Einzelheiten zu den verwendeten Methoden werden in Anhang C näher beschrieben.

PUPY-NUTZLAST-SIGNATUREN

Die Art der Kommunikationsvorgänge zwischen einem Client und einem Server kann bis zu einem gewissen Grad durch passive Datenanalyse abgeleitet werden. Das Vokabular eines Clients, also die unterschiedlichen Nutzlasten, die er erzeugen kann, ist stark eingeschränkt: Es gibt nur neun Arten der Client-Kommunikation. Zwei Typen haben die gleiche Nutzlastlänge, während ein anderer Typ mehrere Längen haben kann. Ein Akteur kann in Pupy benutzerdefinierte Ereignisse erstellen und so potenziell eine größere Vielfalt an Nutzlastlängen schaffen.

Der Server hat ein flexibleres Vokabular und kann mehrere Befehle in einer einzigen DNS-Antwort übermitteln, was das Profiling erschwert. Der Großteil der Kommunikationsvorgänge in einem Pupy-System bezieht sich jedoch auf die Sitzungsinitialisierung, den Schlüsselaustausch und die Heartbeats des Clients an den Server. Die Kommunikation mit dem Server wird dominiert von Bestätigungen von Client-Anfragen, Fehlermeldungen – einschließlich der Notwendigkeit, eine neue Sitzung einzurichten – und dem Austausch von Schlüsseln.

Dadurch können Signaturen für verschiedene Arten der Kommunikation erstellt werden, indem die Länge der zugrundeliegenden Nutzlasten von DNS-Abfragen und -Antworten genutzt wird. Mithilfe dieser Signaturen können wir gewöhnliche Wartungsaktivitäten von wichtigen Befehlen vom Server trennen und die Verwendung benutzerdefinierter Ereignistypen isolieren. Diese können verwendet werden, um ein Profiling des Gesamtverhaltens eines passiv überwachten Pupy-Clients und -Servers, einschließlich Decoy Dog, durchzuführen.

Unten in Abbildung 5 zeigen wir eine Heatmap der Nutzlastlängen, die bei Client-Abfragen und Serverantworten in unseren eigenen Pupy-Daten beobachtet wurden. Während die Serverlängen aufgrund von Befehlsargumenten und verketteten Befehlen mehr Variation aufweisen, sind die Client-Kommunikationsvorgänge klar definiert. Für ein Profiling der Kommunikationsvorgänge verwenden wir die Länge der zugrundeliegenden Nutzlast, einschließlich Prüfsummen und Netzwerkknoteninformationen. Infolgedessen ist zum Beispiel die Client-Bestätigung (Ack) 19 Byte lang und die Server-Ack 6 Byte lang. Anhang D enthält Tabellen mit gängigen Client- und Server-Nutzlastlängen und deren Beziehung zu Befehlen.

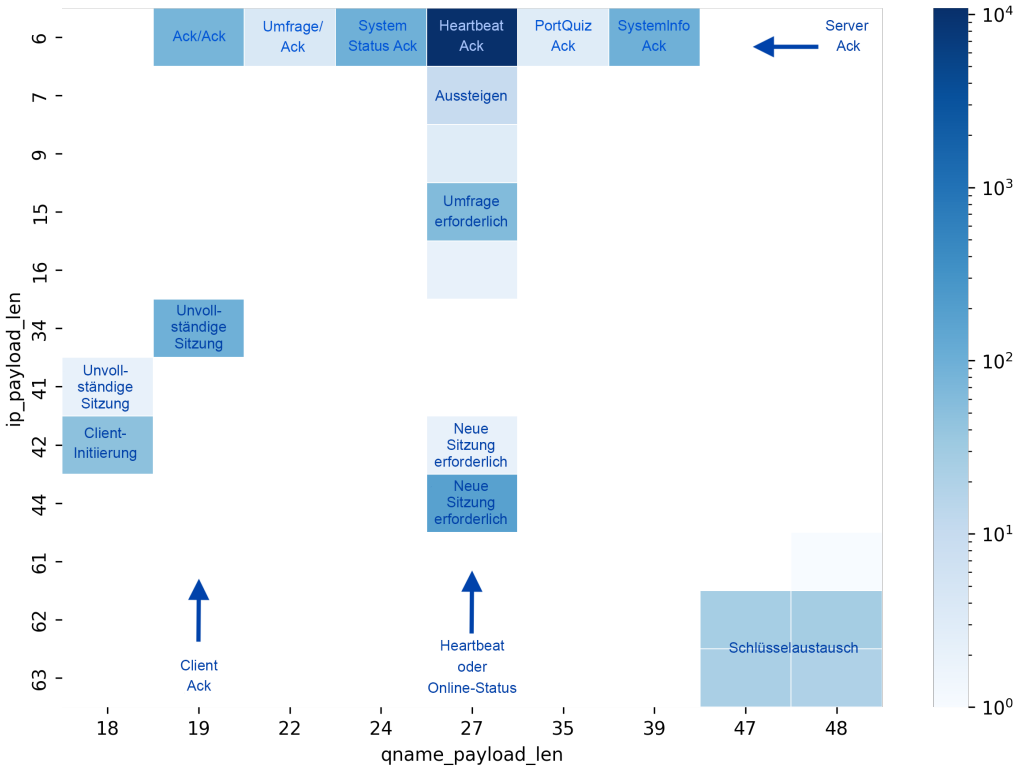


Abbildung 5. Eine kommentierte Verteilung häufiger Nutzlastlängenpaare, die im Pupy-Traffic beobachtet wurden. Bei der Nutzlast handelt es sich um die verschlüsselten Daten, die entweder in der Abfrage oder in der Antwort übertragen werden. Dieses Diagramm enthält keine komplexen DNS-C2-Befehle vom Server und Zellen ohne Anmerkung sind nicht vollständig identifiziert. Die Länge ist in Bytes angegeben.

Decoy Dog

Decoy Dog-Kommunikation wurde nicht nur bei Infoblox-Resolvern beobachtet, sondern auch bei vielen öffentlichen und kommerziellen Resolvern. Um die Funktionsweise von Decoy Dog besser verstehen zu lernen und um zu sehen, wie sich das Toolkit von Pupy unterscheidet, haben wir andere Passive-DNS-Aufzeichnungen verwendet, um unsere eigenen zu ergänzen. Insgesamt deckt unsere Analyse über 15 Millionen DNS-Ereignisse im Zeitraum vom 29. März 2022 bis einschließlich 16. Juni 2023 ab. Außerdem haben wir die Nameserver aktiv untersucht und den passiv erfassten DNS-Traffic mit dem von unserem eigenen Pupy-Client und -Server erzeugten verglichen.

Wir haben eine Reihe von Techniken eingesetzt, um Decoy Dog und seine Funktionsweise besser zu verstehen. Wir haben auch Proben, die wir im öffentlichen Repository VirusTotal gefunden haben, einem Reverse Engineering unterzogen, wodurch wir unsere DNS-Ergebnisse bestätigen und weitere Funktionen enthüllen konnten. In den folgenden Abschnitten beschreiben wir unsere Analyse im Detail und präsentieren die Ergebnisse. Die wichtigsten Erkenntnisse aus dieser Arbeit sind:

- Bei Decoy Dog handelt es sich nicht um Pupy, sondern um eine umfangreiche Überarbeitung, die die Fähigkeiten der Malware erheblich erweitert und die Persistenz auf dem kompromittierten Gerät sicherstellt.
- Es wird von einer Handvoll Akteure betrieben, die unterschiedliche TTPs anwenden und unterschiedlich auf unsere Enthüllung des Toolkits im April 2023 reagiert haben.
- Die Anzahl der insgesamt betroffenen Geräte ist gering, teilweise nur vier für einen Controller.
- Neue Controller, die seit April 2023 registriert wurden, haben sich mittels einer Abschwächung der in unserem ursprünglichen Paper beschriebenen Merkmale angepasst; unter anderem durch Geofencing-Mechanismen zur Begrenzung von Antworten auf Client-IP-Adressen auf bestimmte Standorte.
- DNS-Analysen haben sich als leistungsfähiges Werkzeug erwiesen – nicht nur für die Erkennung von Decoy Dog, sondern auch für ein besseres Verständnis seiner Verwendung und seiner Unterschiede zu Pupy. In Verbindung mit selektivem Reverse Engineering ergibt sich so ein solides Bild von Decoy Dog und der von ihm ausgehenden Bedrohung.

SCHLÜSSELAUSTAUSCH-VORGÄNGE

Wie zuvor beschrieben beginnt eine Sitzung, wenn der Schlüsselaustausch abgeschlossen und der SPI-Wert festgelegt ist. Theoretisch kann eine einzelne verschlüsselte Sitzung unbegrenzt fortgesetzt werden; in der Praxis gibt es jedoch eine Reihe von Bedingungen, unter denen der Controller das Einrichten einer neuen Sitzung verlangt. Somit kann eine einzelne laufende Instanz des Clients in der Regel viele Sitzungen haben. Mithilfe von Pupy-Nutzlast-Signaturen können wir ermitteln, wann gemeinsame Schlüssel zwischen einem Client und einem Server generiert wurden, und grobe Schätzungen der Anzahl der Client-Initialisierungen für jeden Controller im Laufe der Zeit vornehmen, die entweder aufgrund einer erneuten Kompromittierung oder eines Neustarts des Clients durchgeführt wurden.

Abbildung 6 unten zeigt die Zeitleiste der Schlüsselaustauschvorgänge für mehrere Decoy Dog-Controller. Bei einigen Controllern gibt es Lücken bei den beobachteten Schlüsselaustauschvorgängen. Der letzte Schlüsselaustausch für claudfront[.]net wurde im Dezember 2022 beobachtet, obwohl die Client-Aktivität im Jahr 2023 nicht nur anhielt, sondern sogar noch zunahm; über 70 % aller eindeutigen SPI-Werte wurden erstmals im Jahr 2023 beobachtet. In ähnlicher Weise gab es für den Controller allowlisted[.]net ab Dezember 2022 bis nach unserer Enthüllung im April 2023 keine Schlüsselaustauschvorgänge. Schließlich zeigt cbox4[.]ignorelist[.]com auch einen langen Zeitraum ohne Schlüssel-austauschvorgänge, wobei eine kleine Anzahl direkt vor dem Betriebsstopp der Domain stattfand. Wir vermuten, dass die Akteure die Clients neu konfiguriert haben, um den Schlüsselaustausch über einen anderen Transportweg als DNS durchzuführen.

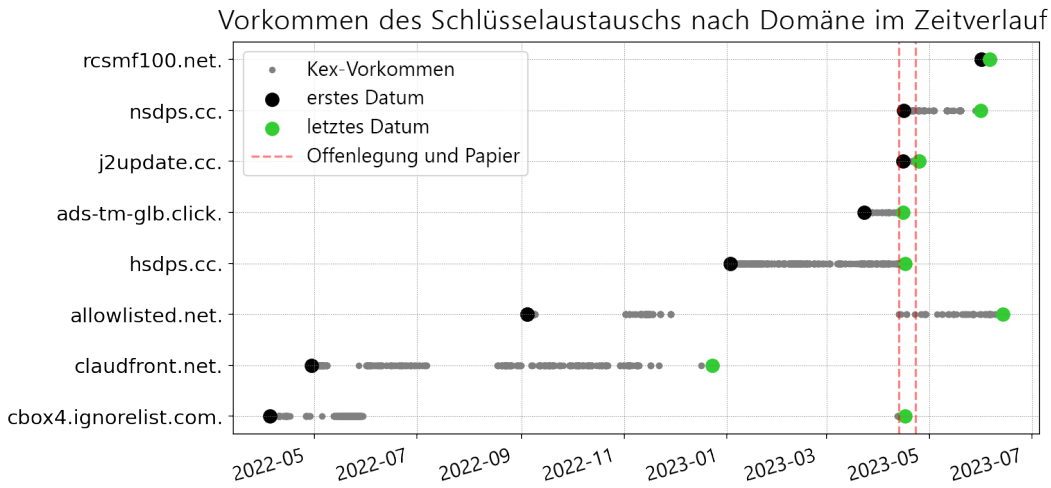


Abbildung 6. Zeitleiste der beobachteten Schlüsselaustauschvorgänge für ausgewählte Decoy Dog-Domains.

CLIENT-ZEITLEISTEN

Zusätzlich zur Gesamtzahl der Clients wollten wir ermitteln, wie viele aktive Clients jeder Controller gleichzeitig unterhielt und wie lange die Clients aktiv mit dem Server kommunizierten. Wir haben die in Anhang C beschriebene Methode zur Gruppierung von Nonce-Werten verwendet. Diese Analyse führte zu wichtigen Erkenntnissen über die Decoy Dog-Operationen über einen langen Zeitraum, wie die folgenden Grafiken zeigen. Insbesondere fanden wir Folgendes heraus:

- Sämtliche Controller verwalten jeweils nur eine kleine Anzahl von Clients. Einige sogar nur vier, und wahrscheinlich verwaltet kein Client mehr als 49.
- Die ursprüngliche Domain cbox4[.]ignorelist[.]com ist einer der größeren Controller und weist zu mehreren Zeitpunkten einen sprunghaften Anstieg bei den Clients auf. Sie verwaltet auch eine kleine Anzahl bereits sehr lange laufender Clients.
- Der zweite beobachtete Controller claudfront[.]net zeigt im Februar 2023 einen dramatischen Aktivitätsanstieg.
- Der dritte beobachtete Controller allowlisted[.]net hatte durchweg eine kleine Anzahl gleichzeitiger Clients.
- Die Controller ads-tm-glb[.]click und hsdps[.]cc haben nach unserer Enthüllung Clients an neue Controller übertragen.
- Claudfront[.]net und allowlisted[.]net haben den Betrieb als Reaktion auf unsere Enthüllung nicht angepasst, cbox4[.]ignorelist[.]com stellte den Betrieb ein, und sowohl hsdps[.]cc als auch ads-tm-glb[.]click haben Clients auf neue Domains übertragen.

Während es schwierig ist, die Gesamtzahl an Clients über den gesamten Zeitraum zu schätzen, deutet die geringe Anzahl gleichzeitig aktiver Clients darauf hin, dass diese Operationen äußerst zielgerichtet ablaufen. Dies erklärt auch, warum Sicherheitsanbieter die Aktivität nicht erkannt und noch keine infizierten Geräte gefunden haben. Die infizierten Clients befinden sich in einer sehr geringen Anzahl von Netzwerken, die offenbar nicht in der Lage sind, die C2-Kommunikation im DNS zu erkennen und zu blockieren.

In den folgenden Liniendiagrammen stellen wir die Aktivität eines einzelnen Clients als Linie dar. Wir nennen sie einen Client-Thread. Die Y-Achse zeigt verschiedene Client-Threads, die durch eine Nonce-Kette identifiziert werden. Wenn ein Pupy-Client durch einen Reboot oder auf andere Weise neu gestartet wird, wird eine neue Nonce generiert und es wird ein neuer Thread beobachtet. In einigen Diagrammen gibt es deutliche Aktivitätsunterbrechungen, die wahrscheinlich auf Client-Neustarts hinweisen. Die X-Achse gibt die Zeit an.

Abbildung 7 zeigt die Client-Aktivität für die erste Decoy Dog-Domain cbox4[.]ignorelist[.]com. Der erste Client-Thread beginnt Ende März 2022 und der längste Thread dauerte fast ein Jahr. Wir können sehen, dass dieser Controller anfangs nur wenige Clients hatte, aber Mitte Mai 2022 kam es zu einer Änderung, die zu fast 40 gleichzeitig aktiven Clients führte. In regelmäßigen Abständen kam es zu ähnlichen Anstiegen der Client-Threads, wobei der größte monatliche Anstieg im August 2022 zu verzeichnen war; wenn neue Client-Threads begannen, endeten jedoch andere. Im gesamten Aktivitätsjahr scheint die Anzahl gleichzeitiger Clients stets unter 50 zu liegen. Aus Abbildung 7 geht auch hervor, dass ein Viertel der Client-Threads sechs Monate oder länger bestehen blieb, was auf einen dauerhaften Betrieb hindeutet. Sämtliche Kommunikationsvorgänge wurden nach dem LinkedIn-Beitrag eingestellt und keine weiteren beobachtet.

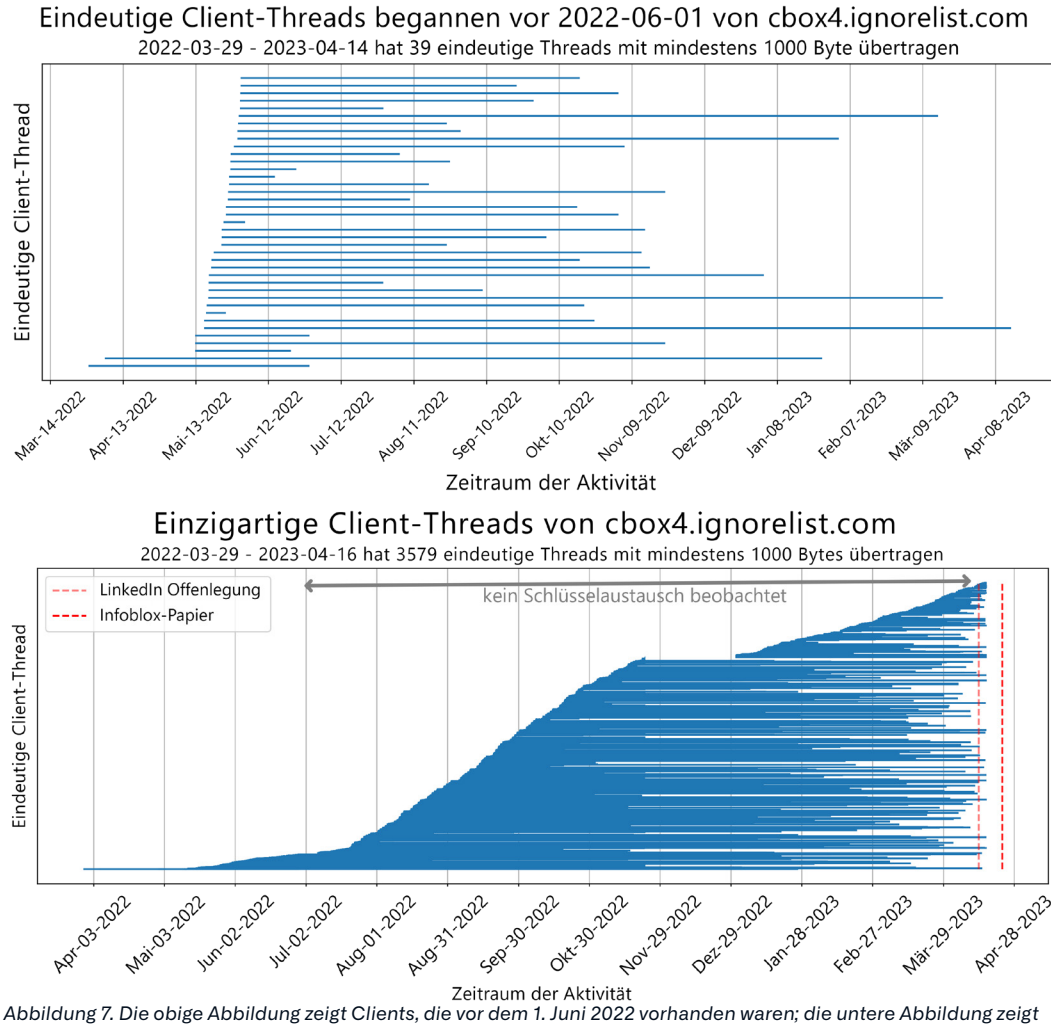


Abbildung 7. Die obige Abbildung zeigt Clients, die vor dem 1. Juni 2022 vorhanden waren; die untere Abbildung zeigt Client-Threads im Zeitverlauf.

Die DNS-Aktivität für claudfront[.]net, chronologisch gesehen die zweite aufgetauchte Decoy Dog-Domain, unterscheidet sich stark von cbox4. Wie in Abbildung 8 unten dargestellt, gab es bis Anfang Februar 2023 weniger als zehn gleichzeitig aktive Clients dieses Controllers. Nach diesem Zeitraum stieg die Zahl der Clients erheblich an, wenn auch nicht in dem Maße, wie man es bei einer weit verbreiteten Infizierung erwarten würde. Der Zeitpunkt dieser Erhöhung liegt kurz vor der Übermittlung einer die Controller-Domain enthaltenden Binärdateiprobe an VirusTotal am 13. Februar.[15] Anders als bei cbox4[.]ignorelist[.]com gab es im Anschluss an unsere Enthüllung keine nennenswerten Änderungen bei den claudfront[.]net- Abfragen.

15 0375f4b3fe011b35e65751335394410 09d015ebecbee78b578c3ed04e0f22568, erstmals 2023 übermittelt

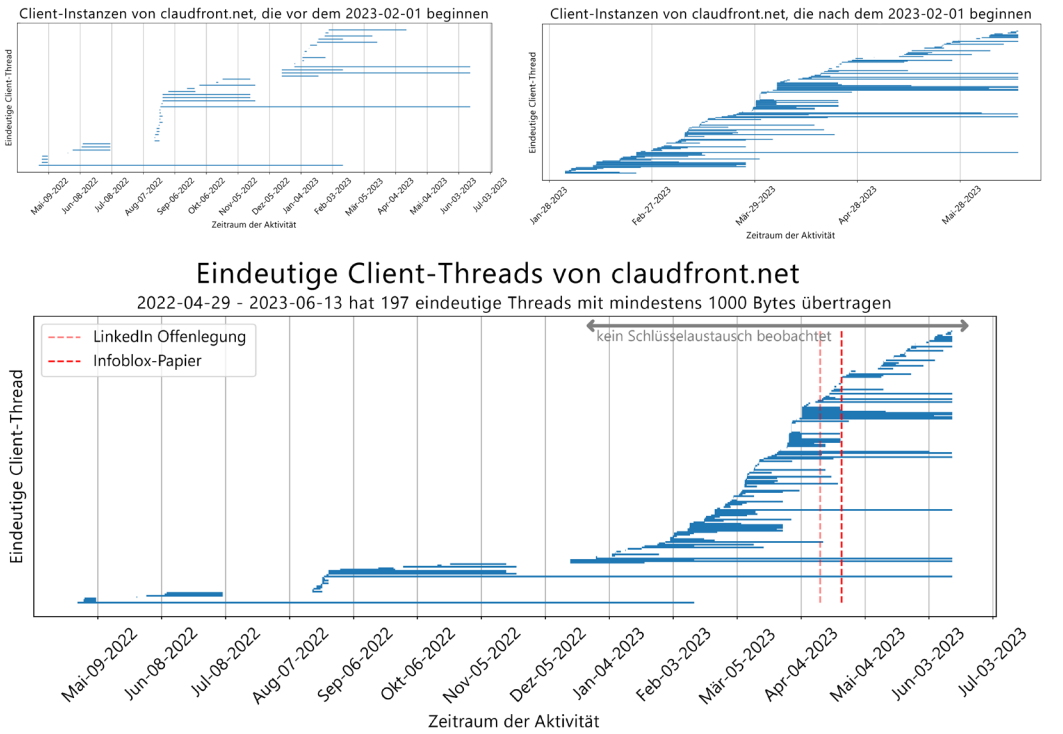


Abbildung 8. Client-Threads für claudfront[.]net, basierend auf der zeitlichen Entwicklung der Nonce-Threads. Anfang Februar 2023 ist eine wesentliche Änderung zu verzeichnen, die mit eigenen Abbildungen unterschiedlicher Zeiträume vergrößert dargestellt wird.

Die dritte Domain allowlisted[.]net zeigt eine weitere Variation im Verhalten. In diesem Fall ist die Anzahl von Clients durchweg gering: stets weniger als 10. Anders als bei claudfront[.]net gab es im Februar 2023 keine Änderung und es ist keine verfügbare Binärdateiprobe bekannt, die allowlisted[.]net enthält. Wie in Abbildung 9 dargestellt, sind von Mitte November 2022 bis kurz nach unserer Enthüllung, die mit dem abrupten Ende der Client-Aktivitäten und dem Neustart mehrerer Threads im April 2023 zusammenfällt, keine Schlüsselaustauschvorgänge zu sehen.

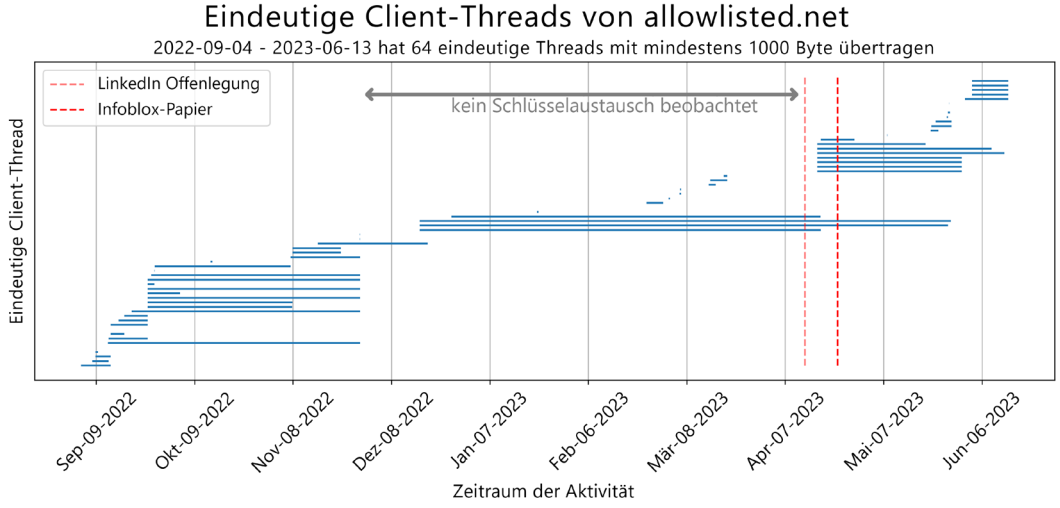


Abbildung 9. Client-Threads für allowlisted[.]net. Historisch gibt eine sehr kleine Anzahl von Clients auf allowlisted[.]net und dies hat sich seit der Enthüllung nicht geändert.

Februar 07:39:55 Uhr (UTC)

Schließlich beobachteten wir verwandte Aktivitäten von hsdps[.]cc, nsdps[.]cc, ads-tm-glb[.]click und j2update[.]cc. Die Domains hsdps[.]cc und ads-tm-glb[.]click wurden nach unserer Enthüllung in den sozialen Medien eingestellt, aber mehrere ihrer Clients wurden auf nsdps[.]cc bzw. j2update[.]cc übertragen. Wir haben das erkannt, indem wir im Laufe der Zeit über alle Domains hinweg Nonce-Chains erstellt und Threads identifiziert haben, die mit einem Controller zu kommunizieren begannen und mit einem anderen endeten.[16]

Die neuen Domains nsdps[.]cc und j2update[.]cc wurden weniger als 48 Stunden nach unseren Social-Media-Enthüllungen registriert. Aus den Client-Thread-Diagrammen ist ersichtlich, dass ein Satz Domains seine Aktivität einstellt, sobald andere ihre Aktivität aufnehmen. Die Controller begannen fast unmittelbar danach, aktiv mit Clients zu kommunizieren. Nach der Entdeckung der Client-Übertragung per DNS-Analyse fanden wir in Binärdateiproben Hinweise auf einen Befehl, der für diese Änderung verantwortlich war. Auf diesen gehen wir später noch genauer ein.

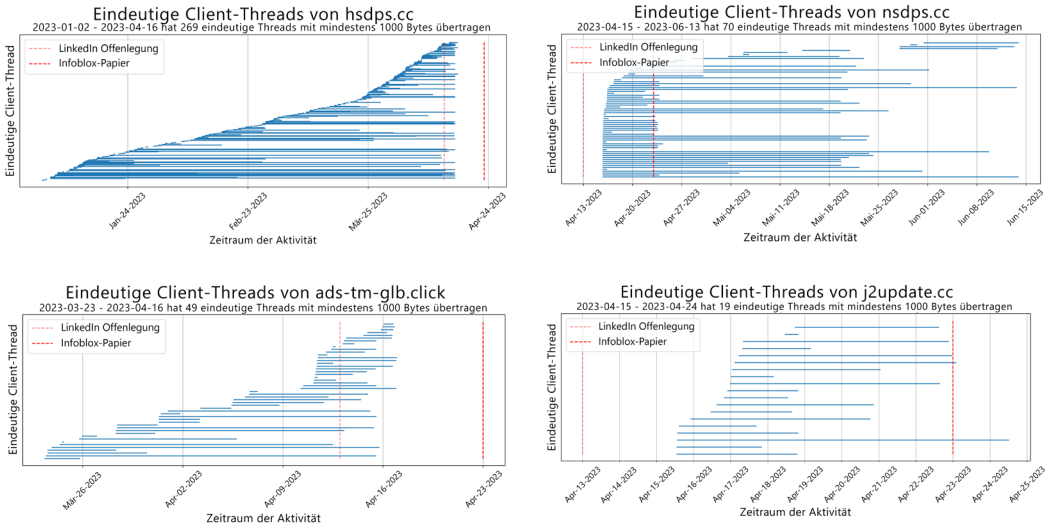


Abbildung 10. Ein Zeitleistenvergleich von vier Decoy Dog-Controller-Domains. Die Controller hspds[.]cc und ads-tm-glb[.]click beenden ihre Kommunikation nach der Infoblox-Enthüllung und die Domains nsdps[.]cc und j2update[.]cc beginnen mit der Kommunikation. Wir haben auch Client-Übertragungen zwischen diesen Domains beobachtet.

Seit unserem ursprünglichen Paper sind weitere Controller aktiv geworden, die jeweils nur eine sehr kleine Anzahl von Clients haben. Das hier gezeigte Client-Verhalten in Verbindung mit der Reaktion auf unsere Ankündigung deutet darauf hin, dass das Decoy Dog Toolkit von mehreren Akteuren genutzt wird.

DECOY DOG-NUTZLAST-SIGNATUREN

Wir haben die Client- und Server-Nutzlastlängen von 15,5 Millionen Abfrageantworten decodiert, die in einem Zeitraum von 13 Monaten im globalen pDNS beobachtet wurden. Dann verglichen wir die Pupy-Signaturen für Client-Server-Nutzlasten mit den beobachteten Decoy Dog-Daten, um das Verhalten der Server besser zu verstehen. Wir stellten zwar fest, dass die Gesamtverteilung des Datenverkehrs mit Pupy übereinstimmte, aber es gab auch deutliche Unterschiede. Decoy Dog-Clients verwenden einen größeren Anfragensatz, auch Vokabular genannt, als im Standard-Pupy.

16 Die Wahrscheinlichkeit, dass dies bei einer 32-Bit-Zufalls-Nonce zufällig geschieht, ist extrem gering, und die Anzahl der Noncen-„Übertragungen“ von einem Controller zu einem anderen war für diese Domains hoch.

Abbildung 11 zeigt die relativen Verteilungen der Nutzlastlängenpaare übergreifend für alle Decoy Dog-Systeme. Anhand unserer Pupy-Signaturen (wie in Anhang D beschrieben) können wir einige unmittelbare Schlussfolgerungen ziehen:

- Es waren mehr als die neun erwarteten Client-Nutzlasten vorhanden.
- Es gab Server-Nutzlastlängen, die wir in unserem Labor nicht beobachtet hatten.
- Der Großteil der Kommunikationsvorgänge bezog sich auf die Aufrechterhaltung von Sitzungen und Schlüsselaustauschvorgänge.
- Ein großer Prozentsatz der Abfragen an die Decoy Dog-Server erhielt eine Fehlerreaktion und wies Abweichungen auf, die eher auf eine Überprüfung durch einen Dritten als durch einen echten Client schließen lassen. Die meisten davon ereigneten sich nach unseren Enthüllungen.

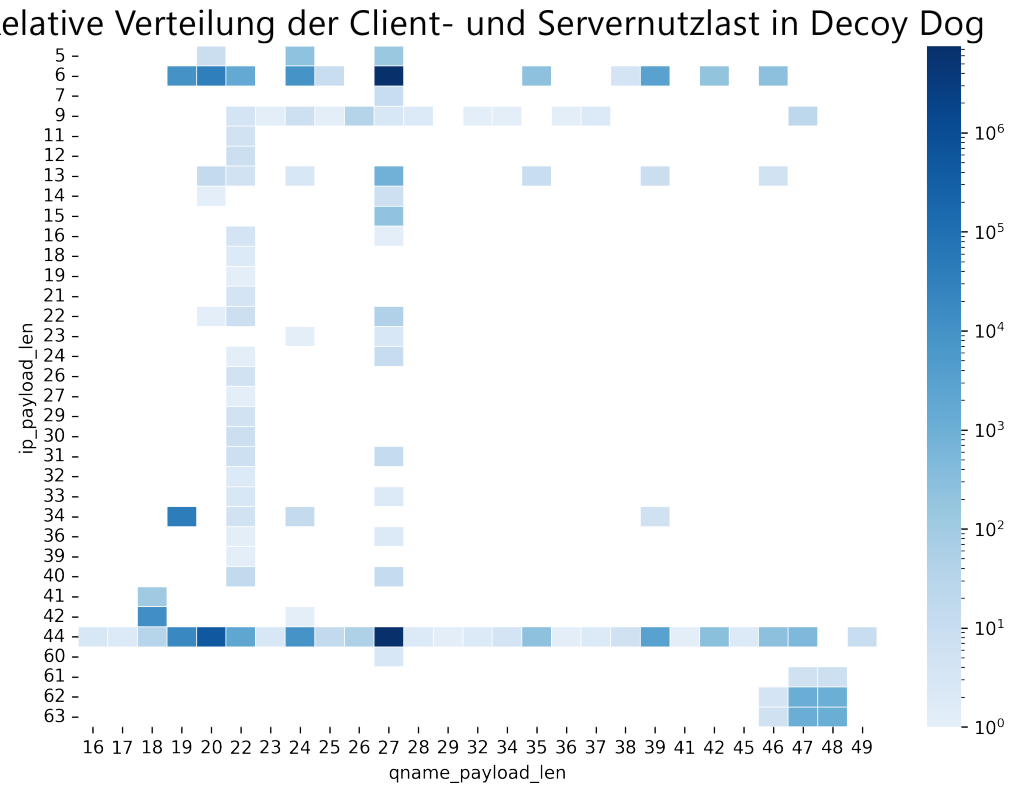


Abbildung 11. Die relative Verteilung der Client- und Server-Nutzlastlängen, wie sie bei der Decoy Dog-Kommunikation beobachtet wird.

Die eindeutigen Client-Nutzlasten umfassten die Längen 20, 25, 38, 42 und 46. Einige davon könnten mit einer anderen Schlüsselkonfiguration oder einer Änderung der Polling-Parameter zusammenhängen; wir können nicht feststellen, worum es sich bei der Kommunikation handelte, aber die Abweichung existiert. Darüber hinaus gab es zusätzliche Antwortnutzlastlängen, die über die bei Pupy beobachteten hinausgingen. Vor allem verfügt Decoy Dog über eine Server-Nutzlast von 13 Bytes, die sich im Laufe der Zeit in Form von Aktivitätsschüben bemerkbar macht. Wir können nicht feststellen, um welche Nutzlast es sich hierbei handelt; sie stimmt jedoch mit einem einzelnen Befehl überein, der die Übertragung von 8 Bytes an Daten an den Client erfordert. Wir sahen auch eine Reihe von Serverantworten, die eine Nutzlast von 5 Byte enthielten – eine weitere Länge, die in unseren Pupy-Daten nicht beobachtet wurde und die auf einen einzelnen Befehl hindeutet, der keine Datenübertragung an den Client erfordert. Die folgende Abbildung 12 fasst die einzigartigen Nutzlastpaare zusammen, die bei Decoy Dog gefunden und in unseren Pupy-Experimenten nicht gesehen wurden.

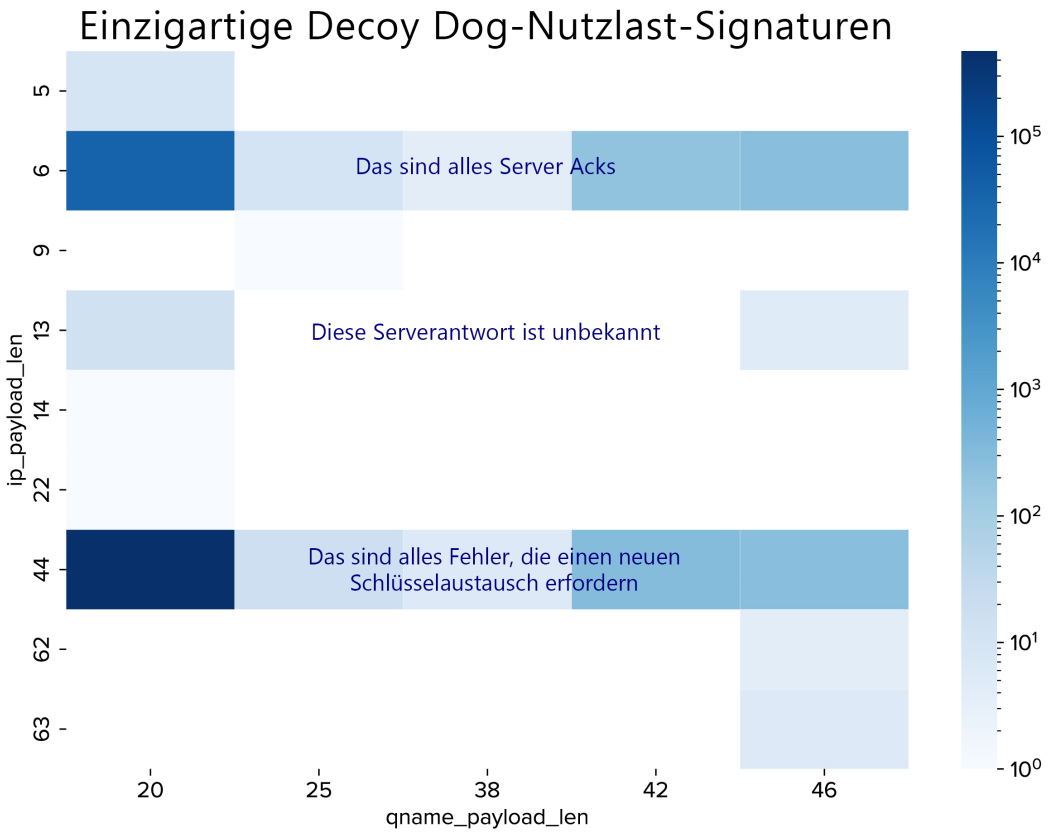


Abbildung 12. Eine Zusammenfassung der in Decoy Dog beobachteten Client-Server-Nutzlastlängenpaare, die in der standardmäßigen Pupy-Kommunikation nicht gefunden wurden.

Wir haben auch Zeitreihen verwendet, um Änderungen an den Standardkonfigurationen zu identifizieren. Im Rahmen einer eingerichteten Pupy-Sitzung führt der Client alle 30 Sekunden einen Check-in durch. Mithilfe einer statistischen Analyse der Schwankungen bei den Heartbeat-Anfragen der Clients haben wir neben den standardmäßigen 30-Sekunden-Heartbeat-Intervallen auch Heartbeat-Intervalle von 2 Minuten und 30 Minuten festgestellt.

Durch diese Analyse konnten wir die Kommunikationsarten für die einzelnen Decoy Dog-Domains verstehen lernen, und wie man Routine-Kommunikation zur Aufrechterhaltung von Sitzungen von Fernzugriffsbefehlen unterscheidet. Wir konnten auch augenscheinliche Anpassungen von Pupy isolieren, die für alle und innerhalb Teilmengen von Decoy Dog-Servern verwendet werden. Wir haben herausgefunden, dass der überwiegende Teil des Decoy Dog-Datenverkehrs aus routinemäßigen Bestätigungen und Fehlern besteht, und dass die Fehlerkommunikationsvorgänge in keinem Verhältnis zu dem standen, was wir aufgrund unserer Beobachtungen bei Pupy erwartet hatten. Wir teilen die Ergebnisse unserer Untersuchung dieses Phänomens der Fehlerreaktionen im nächsten Abschnitt.

WILDCARD AND GEOFENCING BEHAVIOR

Wir haben in unserem ursprünglichen technischen Paper berichtet, dass Decoy Dog-Server wiederholte DNS-Abfragen beantwortet haben. Diese Tatsache ist nach wie vor verwirrend. Als wir versucht haben, zu verstehen, wann und wie Decoy Dog auf eine Abfrage reagieren würde, die ursprünglich Tage oder Wochen zuvor gestellt worden war, haben wir ein noch überraschenderes Verhalten entdeckt. Einige der Decoy Dog-Server reagieren nicht nur auf Wiederholungen, sondern auf jede Abfrage, die der Pupy-Codierung entspricht. Im DNS

nennt man dies eine Wildcard-Antwort. Wo ein normaler Pupy-Server eine NXDOMAIN- oder SERVFAIL-Antwort ausgeben würde, gibt der Decoy Dog-Server in der Regel 15 IP-Adressen aus.

Abbildung 13 unten zeigt Antworten auf randomisierte Abfragen. In diesem Fall haben wir die Ausdrücke „wild“ und „wildcard“ in den Abfragenamen eingefügt und daraufhin 15 Antworten von zwei verschiedenen Decoy Dog-Servern erhalten. Die Antworten sind für jede Abfrage unterschiedlich und entsprechen dem Pupy-Codierungsschema. Bei unseren Recherchen haben wir herausgefunden, dass Decoy Dog fast alle Fehler auf diese Weise verarbeitet, anstatt die erwarteten NXDOMAIN-Antworten auszugeben. Weitere Informationen zur Fehlerverarbeitung finden Sie in Anhang E.

```
; <<>> DiG diggui.com <<>> @ns1.rtupdates.net wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net A
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22151
;; flags: qr aa rd; QUERY: 1, ANSWER: 15, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. IN A

;; ANSWER SECTION:
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 64.88.80.242
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 131.163.188.250
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 68.221.203.220
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 198.206.187.196
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 200.37.65.250
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 75.195.241.234
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 141.67.92.44
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 142.153.85.81
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 209.92.80.161
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 147.26.100.52
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 213.83.7.105
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 150.143.51.118
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 153.171.88.194
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 219.226.5.44
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.rtupdates.net. 60 IN A 157.111.237.108

;; Query time: 150 msec
;; SERVER: 5.252.179.232#53(5.252.179.232)
;; WHEN: Sat Jun 03 15:29:11 UTC 2023
;; MSG SIZE  rcvd: 321
```

```
; <<>> DiG diggui.com <<>> @ns1.allowlisted.net wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net A
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 33023
;; flags: qr aa rd; QUERY: 1, ANSWER: 15, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. IN A

;; ANSWER SECTION:
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 64.88.161.73
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 67.179.145.230
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 69.153.193.38
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 71.14.146.226
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 73.22.176.2
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 138.151.231.153
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 141.232.226.212
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 79.241.118.178
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 209.158.29.150
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 147.248.180.89
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 148.158.234.156
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 215.63.12.236
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 153.141.240.250
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 219.18.219.74
wildcard5yt1j46ra2gwossxhw1q9999.2vzxfwild3999999.allowlisted.net. 60 IN A 156.250.150.9

;; Query time: 151 msec
;; SERVER: 83.166.240.52#53(83.166.240.52)
;; WHEN: Sat Jun 03 15:31:15 UTC 2023
;; MSG SIZE  rcvd: 323
```

Abbildung 13. Wildcard-Antwortverhalten von zwei autoritativen Decoy Dog-Servern. In beiden Fällen antworteten die Server mit 15 IP-Adressen im Einklang mit der Pupy-Codierung auf dieselbe zufällige Abfrage, die die Zeichenfolgen „wild“ und „wildcard“ enthielt.

Noch überraschender ist, dass einige der Decoy Dog-Server auch abhängig von der IP-Adresse des rekursiven Resolvers, der die Abfrage im Namen des Clients stellt, unterschiedlich reagieren. In Abbildung 14 zeigen wir die Wiederholung einer Abfrage an die Decoy Dog-Domain nsdps[.]cc, die ursprünglich mehrere Wochen zuvor stattfand. Beim Durchführen der Abfrage über die öffentlichen Resolver von Yandex erhielten wir eine Antwort mit 15 IP-Adressen. Wir haben außerdem 15 IP-Adressen von den russischen öffentlichen Resolvern von TimeWeb erhalten. Von den über dreißig öffentlichen Resolvern, die wir ausprobiert haben, gab jedoch kein weiterer eine Antwort aus. Diese Art von Verhalten passt zu Geofencing, bei dem ein Server basierend auf der Geolokalisierung der IP-Adresse auf DNS-Abfragen antwortet. Wir entdeckten dieses Verhalten im Juni 2023 und stellten fest, dass einige der Server nur reagierten, wenn wir DNS-Abfragen über russische IP-Adressen leiteten, während andere auf jede korrekt beschaffene Abfrage von jedem Standort aus antworteten. Durch diese Art des selektiven Antwortens wird sichergestellt, dass der Controller nur mit Clients kommuniziert, die sich scheinbar in Russland befinden. Wir wissen, dass diese Funktion nach der Enthüllung hinzugefügt wurde, da die Controller zuvor Abfragen von den rekursiven Resolvern von Infoblox aufgelöst hatten.

```
; <<>> DiG diggui.com <<>> @77.88.8.8 qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc A
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 42579
;; flags: qr rd ra; QUERY: 1, ANSWER: 15, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. IN A

;; ANSWER SECTION:
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 46 IN A 72.11.125.198
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 36 IN A 203.92.202.218
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 45 IN A 76.74.229.130
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 44 IN A 207.26.86.188
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 31 IN A 80.154.112.164
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 43 IN A 146.160.113.9
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 52 IN A 148.235.159.60
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 41 IN A 151.103.182.130
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 54 IN A 89.76.7.130
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 45 IN A 218.111.60.250
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 42 IN A 93.43.159.18
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 49 IN A 128.88.84.164
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 45 IN A 195.161.207.129
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 51 IN A 68.172.178.156
qnomwvxags5u1pawkzmkllkmuoda9999.abp11i4yk5y1fqyd4tpzm1q9.nsdps.cc. 49 IN A 199.24.240.30

;; Query time: 459 msec
;; SERVER: 77.88.8.8#53(77.88.8.8)
;; WHEN: Tue Jun 20 14:31:11 UTC 2023
;; MSG SIZE  rcvd: 335
```

```
; <<>> DiG diggui.com <<>> @74.82.42.42 hoxlgxq9.yopzgoha3rlp4pdcclosfb63yodq9999.enueh2eluu6uqnjtjpid4lq9.nsdps.cc A
; (1 server found)
;; global options: +cmd
;; connection timed out; no servers could be reached
```

```
; <<>> DiG diggui.com <<>> @ns2.nsdps.ns2.name hoxlgxq9.yopzgoha3rlp4pdcclosfb63yodq9999.enueh2eluu6uqnjtjpid4lq9.nsdps.c
; (1 server found)
;; global options: +cmd
;; connection timed out; no servers could be reached
```

Abbildung 14. Ein Vergleich der Antworten auf eine wiederholte Decoy Dog-Abfrage von öffentlichen Yandex-Resolvern, öffentlichen Hurricane Electric-Resolvern und dem autoritativen Resolver. Diese Abfragen erfolgten nacheinander über einen Tor-Browser. Lediglich die Abfrage über Yandex erhielt eine Antwort.

Wenn eine Abfrage für einen Domainnamen erfolgt, der nicht mit der Pupy-Standardcodierung decodiert werden kann (wir haben für diesen Test zusätzliche Zeichen hinzugefügt), geben die nsdps[.]cc- Server eine IP-Adresse aus, die im Wesentlichen ein Sinkloch ist. Wie unten in Abbildung 15 gezeigt, haben wir die Abfrage leicht geändert, sodass sie nicht richtig decodiert werden kann. In diesem Fall wurde innerhalb des Bereichs 172.0.0.0/8 eine zufällige IP-Adresse ausgegeben. Normalerweise würde Pupy eine NXDOMAIN-Antwort ausgeben.

```
; <<>> DiG diggui.com <<>> @77.88.8.1 hoxlgxq9.yopzgoha3rlp4pdcclosfb63yodq9999.wildenueh2eluu6uqnjtjpid4lq9.nsdps.cc A
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 2019
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;hoxlgxq9.yopzgoha3rlp4pdcclosfb63yodq9999.wildenueh2eluu6uqnjtjpid4lq9.nsdps.cc. IN A

;; ANSWER SECTION:
hoxlgxq9.yopzgoha3rlp4pdcclosfb63yodq9999.wildenueh2eluu6uqnjtjpid4lq9.nsdps.cc. 32 IN A 172.67.132.113

;; Query time: 1695 msec
;; SERVER: 77.88.8.1#53(77.88.8.1)
;; WHEN: Tue Jun 20 23:44:46 UTC 2023
;; MSG SIZE  rcvd: 124
```

Abbildung 15. Eine Abfrage für einen ungültigen Pupy-Domainnamen für Controller nsdps[.]cc gibt eine zufällige IP-Adresse im Bereich 172.0.0.0/8 anstelle der erwarteten NXDOMAIN-Antwort aus.

Ein Teil dieses Verhaltens könnte als Artefakt der DNS-Auflösung des Clients erklärbar sein. Wenn ein Host im DNS abgefragt wird, versuchen einige Resolver, potenziell verwandte Domainnamen aufzulösen, um sich auf potenzielle zukünftige Abfragen vorzubereiten. Beispielsweise kann ein rekursiver Resolver, der eine Abfrage für www[.]baddomain[.]com erhält, versuchen, zusätzlich zu www[.]baddomain[.]com auch noch baddomain[.]com aufzulösen. Wir haben dieses Verhalten auf unserem eigenen Pupy-Server beobachtet, als wir Client-Abfragen über einige öffentliche Resolver geleitet haben.

SINGLE-LABEL-ANTWORTEN

Standardmäßig lehnt Pupy eingehende Abfragen an Label ab, die nicht mit der Struktur einer Client-Kommunikation oder einer etablierten Ping-Abfrage übereinstimmen. Wie wir jedoch oben im Abschnitt „Spezieller Umgang mit Domainnamen“ erklärt haben, ermöglicht die Funktion zur DNS-Aktivierungsanfrage einem Akteur, den Pupy-Server so zu konfigurieren, dass dieser auf Abfragen bezüglich benutzerdefinierter Ressourcen antwortet. In den globalen pDNS-Protokollen haben wir Abfragen mit einer Single-Label-Subdomain identifiziert. Die einzige solche Subdomain war „m“ und wir stellten die Hypothese auf, dass die Auflösung dieser Domains über die Aktivierungsfunktion möglich ist. Es liegt in der Natur der Aktivator-Hash-Funktion, dass für diese Abfragen eine einzelne statische IP-Adresse ausgegeben werden sollte. Wir haben dieses Verhalten bei 4 Domains festgestellt: hsdps[.]cc, nsdps[.]cc, j2update[.]cc, und ads-tm-glb[.]click. Es ist ein weiteres gemeinsames Merkmal dieser Gruppe von Domains, das bei anderen Controllern nicht zu finden ist. Jede von ihnen gab eine einzelne IP-Adresse aus. Allerdings fanden wir in den Antworten anstelle der erwarteten statischen IP-Adresse 104 einmalige Adressen. Dies scheint auf einen Funktionsunterschied zum Standard-Pupy hinzuweisen, aber wir kennen seinen Zweck nicht.

ANALYSE VON BINÄRDATEIPROBEN

Im Anschluss an unsere DNS-Entdeckungen haben wir uns die in VirusTotal verfügbaren Binärdateiproben angesehen, um festzustellen, ob die Ursache der Unterschiede zu Pupy leicht zu erkennen sein würde. Durch die Analyse der Importe und Funktionstabellen von zwei Decoy Dog-Proben haben wir eine eindeutige Signatur speziell für Decoy Dog-Implantate identifiziert, mit der wir weitere Decoy Dog-Proben entdecken konnten. Das Reverse Engineering dieser Proben bestätigte außerdem unsere Erkenntnis, dass sich Decoy Dog erheblich von Pupy unterscheidet und dass der ausgereifteste Code möglicherweise von einem zweiten Entwickler erstellt wurde. Der Client hat ein Upgrade auf Python 3.8 erhalten und enthält eine Reihe neuer Transporte, verbesserte Verschlüsselung, benutzerdefinierte Befehle und neue DNS-Funktionen. Die mit dem Controller claudfront[.]net in Verbindung stehende Probe enthält Funktionen, die in den anderen nicht zu finden sind. In diesem Abschnitt werden einige der wichtigsten Erkenntnisse und der Prozess

beschrieben. Weitere technische Details finden Sie in Anhang F. Analysedaten zu den Binärdateien werden ebenfalls zu unserem GitHub-Repo hinzugefügt.

Die erste Probe wurde im September 2022 hochgeladen und die anderen wurden im Jahr 2023 hochgeladen; drei davon nach unserer Enthüllung. Wir haben die Konfigurationen der verschiedenen Decoy Dog-Proben extrahiert und verglichen, was gezeigt hat, dass sich die Verschlüsselungsschlüssel der einzelnen Servern unterscheiden. Alle Proben, die mit cbox4[.]ignorelist[.]com kommuniziert haben, enthalten dieselben RSA- und SSL-Schlüssel, was darauf hindeutet, dass das Vorhandensein unterschiedlicher Proben nicht mit Änderungen der Serverschlüssel zusammenhängt. Eine vollständige Liste der entschlüsselten Schlüssel finden Sie im in Anhang I genannten GitHub-Repository. Das früheste SSL-Zertifikat in den Proben wurde am 26. Dezember 2021 generiert und gehört zu cbox4[.]ignorelist[.]com, dem ersten beobachteten Controller.

Eine wichtige Entdeckung war, dass der Pupy-Client von Decoy Dog angepassten Code enthält, der es den Angreifern ermöglicht, während der Laufzeit Java-Module zu senden und auszuführen, indem diese in einen Java-Virtual-Machine-Thread (JVM) eingeschleust werden. Diese Funktion gibt es in Standardversionen von Pupy nicht. Dieser Code wurde in allen Decoy Dog-Proben gefunden und ist für alle Instanzen identisch. Die verbleibenden Binärfunktionen in allen bekannten Decoy Dog-Clientproben sind identisch mit den Funktionen in den Pupy-Basis-Clients.

Die Einbindung von Java-Modulen wirft mehr Fragen auf als sie beantwortet. Pupy ist bereits in der Standardversion enorm leistungsfähig und unterstützt die Verwendung von Python-Modulen „out of the Box“. Das Erweitern dieser Funktionen und das Schreiben von Python-Modulen ist ein unkomplizierter Prozess, der keine Änderungen auf Serverseite oder Änderungen am Client-Binärprogramm erfordert. Man könnte leicht ein Python-Modul für das Ausführen von Java-Modulen erstellen. Im Kontrast dazu ist das Injizieren von Java-Modulen während der Laufzeit ohne die Verwendung von jni.h (oder den Rest der Standard-Java/C-API) keine triviale Aufgabe und erfordert spezielle Kenntnisse. Es ist daher wahrscheinlich, dass das Hinzufügen dieser Java-Module es Angreifern ermöglicht, Systeme anzugreifen, auf denen kein Python läuft oder auf denen eine privilegierte oder nicht überwachte virtuelle Java-Maschine läuft – oder sie für Szenarien dient, in denen die Angreifer vermeiden wollen, Beweise auf dem Rechner zu hinterlassen, indem sie keine Dateien erstellen.

Die Clients verfügen außerdem über neue Funktionen, die im Laufe der Zeit ausgereift sind. Die Client-Software wird erstellt, indem eine Python-Konfigurationsdatei in eine bestimmte Binärdatei gemarshallt wird. Die Konfigurationsdatei enthält Einstellungen, alle für die Kommunikation erforderlichen Schlüssel (RSA, SSL-Zertifikate, Passwörter usw.) und Client-Python-Module. Die in den Proben gefundenen Module, die von den kompromittierten Geräten entpackt und ausgeführt werden, unterscheiden sich erheblich vom öffentlich verfügbaren Pupy-Code.

Das Extrahieren und Analysieren von eingebetteten Modulen offenbart eine faszinierende Geschichte der Decoy Dog-Entwicklung und spezieller Anpassungen. Erstens wurde eine beträchtliche Anzahl von Pupy-Modulen einfach aus Decoy Dog entfernt, möglicherweise, weil die Angreifer sie für nutzlos hielten. Zweitens weisen ähnliche Proben eine große Anzahl von Unterschieden bei den Modulen auf, manchmal mit sehr unterschiedlichen Funktionen. Drittens zeigen die große Anzahl von Änderungen und die durch neue Funktionen hinzugekommene Komplexität, dass die Feinabstimmung von Pupy beträchtlichen Entwicklungsaufwand und Ressourcen erfordert hat. Darüber hinaus wurden die Pupy-Codebasis und -Module von Python 2.7 in Python 3.8 portiert, was die Qualität des Codes, die Stabilität der Speicherabläufe und die Kompatibilität mit Windows verbessert hat. Die Proben enthalten eine Client-Version, die sich im Laufe der Zeit von 3 auf 4 ändert; der aktuellste verfügbare Pupy-Client ist Version 2. Eine Zeitleiste, die die Übermittlungsdaten im Vergleich zur Codereife und den wichtigsten Funktionen zusammenfasst, finden Sie unten in Abbildung 16.

Durch die Analyse der Art und Anzahl der geänderten Module konnten wir feststellen, dass aus Sicht der Codereife die Probe mit dem Hash ad186df91282cf78394ef3bd60f04d859bcacccbcdcbfb620cc73f19ec0cec64 die früheste öffentlich verfügbare Decoy Dog-Binärdatei darstellt. Sie kommuniziert mit dem cbox4[.]ignorelist[.]com- Nameserver. Obwohl sie den meisten Code mit Pupy gemeinsam hat, wurde diese Probe erst am 27. April 2023 bei VirusTotal hochgeladen, also einige Tage nach der Veröffentlichung unseres Papers. Auf Grundlage des enthaltenen SSL-Zertifikats könnte diese Probe bereits im Dezember 2021 entstanden sein. Der Entwickler hat spezielle Polling-Funktionen, eine XOR-Funktion, neue Transporte und volle Unterstützung für Multithreading-Netzwerkkommunikation hinzugefügt. Interessanterweise zielen einige neue Module speziell auf Win32 ab, auch wenn es sich bei allen bisherigen Proben um Linux-Bibliotheken handelt. In dieser ausführbaren Datei ist der Code, der für die Verarbeitung der DNS-Kommunikation verantwortlich ist, derselbe wie beim Standard-Pupy.

Im Laufe der Zeit wurden die mit cbox4[.]ignorelist[.]com kommunizierenden Proben immer komplexer. Im Laufe einer Serie von drei Proben wurden immer mehr Kommunikationsmodule hinzugefügt, darunter ein komplettes Modul für die Kommunikation mit bidirektionalen Streams über synchrones HTTP (BOSH) sowie komplette Umschriften der SSL-, TCP- und UDP-Module. Die Akteure hinter Decoy Dog fügten außerdem eine Reihe von Skripten hinzu, um die vorhandenen Exploit- und Kommunikationsmodule auf Windows-Plattformen zu portieren, schrieben den picocmd-Client, der für die DNS-Kommunikation zuständig ist, neu und implementierten eine Reihe von QoL- und Stabilitätsverbesserungen in den alten Code. Verweise auf Windows im Code deuten auf die Existenz eines aktualisierten Windows-Clients hin, der die neuen Decoy Dog-Funktionen beinhaltet, obwohl alle aktuellen Proben auf Linux abzielen.

Die späteren Versionen enthalten außerdem ein Notfallmodul, das es einer kompromittierten Maschine ermöglicht, einen DNS-Server eines Dritten zu kontaktieren, wenn die Malware über einen längeren Zeitraum daran gehindert wird, mit dem C2-Server zu kommunizieren. Dieses Modul verwendet einen DGA, um Domains auszuwählen, die der Client innerhalb kostenloser DDNS-Dienste abfragen kann. Diese Versionen ermöglichen auch Bootstrapping, um den C2-Controller zu lokalisieren, Beacon-Domains einzurichten und CNAME-Abfragen in den Emergency-Dienst zu integrieren. Umfangreiche Persistenzmechanismen, die ab Client-Version 3 zu finden sind, sind Funktionen, die am häufigsten mit nachrichtendienstlichen Operationen in Verbindung gebracht werden als mit solchen, die von finanziell motivierten Akteuren oder Red Teams durchgeführt werden.

Der ausgereifteste Code mit Verbindung zum Controller claudfront[.]net beinhaltet zwei neue Befehle namens AlterDnsCncDomain und CompromisedNode. Wie zuvor beschrieben, haben wir anhand der Analyse von Client-Nonce-Werten festgestellt, dass einige der Decoy Dog-Akteure nach unserer Enthüllung Clients auf neue Controller umgestellt haben. Basierend auf dem öffentlich verfügbarer Pupy-Quellcode haben wir nicht erkannt, wie das ohne die Verwendung benutzerdefinierter Befehle möglich ist. Es wirkt wahrscheinlich, dass der Befehl AlterDnsCnCDomain die Quelle für diese Client-Übergänge ist und damit die mit nsdps[.]cc verknüpften Controller möglicherweise den fortschrittlichsten Code verwenden. Die große Abweichung dieses Codes vom Rest könnte darauf hindeuten, dass ein neuer Entwickler beteiligt war. Der Code enthält Version 4 des Clients.

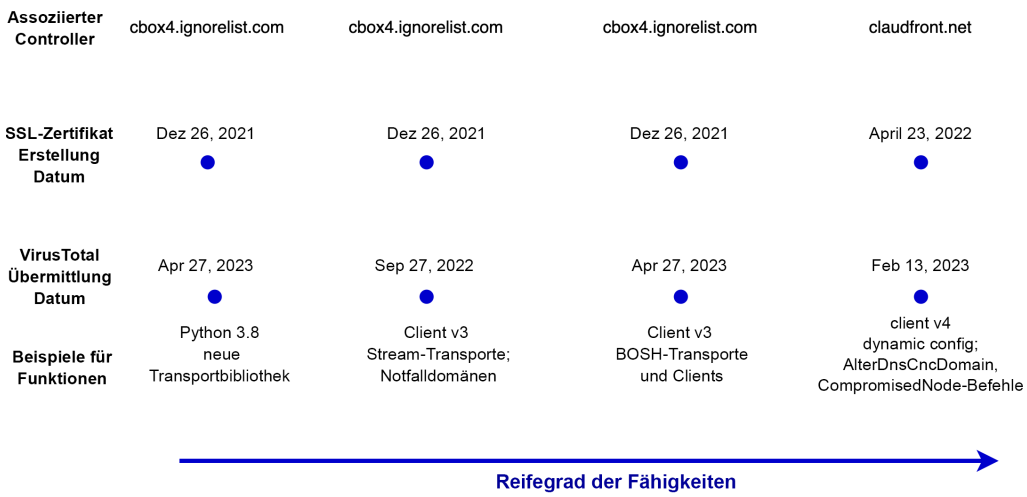


Abbildung 16. Eine Zeitleiste der Übermittlungen an VirusTotal im Zusammenhang mit Decoy Dog und der Reife des Codes.

Es ist erwähnenswert, dass trotz aller Verbesserungen von Decoy Dog die YARA-Regeln, die für einfachere Versionen von Pupy entwickelt wurden, die Malware trotzdem noch erkennen können. Sie können jedoch nicht erkennen, dass die Proben erheblich vom bekannten Code und den bekannten Funktionen abweichen. Dies könnte Malware-Forscher zu der falschen Annahme verleiten, dass es sich bei Decoy Dog-Proben nur um einfaches Pupy handelt, da beide Malware-Arten durch dieselbe Regel gemeldet werden. Aus diesem Grund haben wir in Anhang G eine neue YARA-Regel für Decoy Dog beigefügt.

VERGLEICH VON CONTROLLERN

Infoblox verfolgt derzeit 21 Decoy Dog-Domains. Einige davon hatten nur geringe oder keine erkennbare C2-Aktivität und wir geben sie zum jetzigen Zeitpunkt nicht bekannt. Einige Controller haben sich nach unserer ersten Enthüllung in den sozialen Medien geändert; der Rest hat sich geändert, nachdem wir unser erstes Paper veröffentlicht hatten. Sie alle reagierten, indem sie entweder den Betrieb einstellten, Clients auf neue Controller verlagerten oder das „ping“-Verhalten änderten, das wir im Paper beschrieben hatten. Einige haben sogar Geofencing hinzugefügt. Diese Reaktionen in Verbindung mit anderen verwendeten TTPs lassen uns zu dem Schluss kommen, dass aktuell mindestens drei Akteure das Toolkit nutzen. Unten in Tabelle 1 haben wir eine Teilmenge von Controller-Domains basierend auf deren Verhalten und ähnlichen Eigenschaften gruppiert.

Domaingruppe	Eigenschaften
cbox4.ignorelist[.]com	• Erste aktive Domain und wahrscheinlicher Ursprung des Decoy Dog-Toolkits • Nach Enthüllung deaktiviert • Verwendung von Afraid-DDNS • Heartbeat-Intervall von 30 Sekunden • Kein Geofencing • Mindestens drei verschiedene Client-Software-Iterationen • Von uns erstmals Ende März 2022 beobachtet, könnte aber bereits im Dezember 2021 vorhanden gewesen sein • Client v2 und v3
claudfront[.]net allowlisted[.]net maxpatrol[.]net atlas-upd[.]com	• Zweiter Satz aktiver Controller, ab Mai 2022 • Weiterführung des Betriebs nach Enthüllung • Registriert bei Namecheap • Abfragen an ping12.<domain> Bevor verschlüsselte Fernzugriffskommunikation erstmals gesehen wurde • Ping-Antwort in eine NODATA-Antwort geändert • Russisches IP-Hosting • Heartbeat-Intervall von 30 Sekunden • Kein Geofencing • Client v3 und v4 • Es gibt einige Unterschiede zwischen allowlisted[.]net und claudfront[.]net, die auf verschiedene Akteure hinweisen könnten
hsps[.]cc nsdps[.]cc j2update[.]cc ads-tm-glb[.]click	• Dritter Satz aktiver Controller, ab Dezember 2022 • Nach der Enthüllung wurden Clients zwischen Controllern verschoben • Original-Controller geparkt • Heartbeat-Intervalle von 2 Minuten und 30 Minuten • Geofencing seit Enthüllung • Ping-Antwort auf eine einzelne nicht-lokale Loopback-IP-Adresse geändert • Verwendung eines einzelnen Domain-Labels: m • Möglicherweise Client v4
rcmsf100[.]net	• Erstmals im Juni 2023 beobachtet • Teilt Hosting mit allowlisted[.]net • Ping-Antwort von NODATA • Geofencing

Tabelle 1. Ein Vergleich verschiedener Decoy Dog-Controller.

DECOY DOG IN INFOBLOX-NETZWERKEN

Infoblox hat festgestellt, dass unsere Resolver durch einen Scanner eines Sicherheitsanbieters ausgelöst wurden, der die Abfragen von Decoy Dog wiederholt hat. Eine Kombination aus dem Verhalten des Scanners und dem Verhalten von Decoy Dog erzeugte das entdeckte Signal. Internet-Scanning hat sich zu einem bedeutenden Geschäft entwickelt, und die Scanvorgänge machen inzwischen einen großen Teil des Internet-Traffics aus. Sie werden sowohl von legitimen als auch von bösartigen Akteuren durchgeführt. Im Rahmen einer aktuellen Studie wurde ein Darknet-Teleskop verwendet, um die Auswirkungen dieser Scans verstehen zu lernen.[17] Während die meisten Scans auf Port-Scans beschränkt sind, die versuchen, offene Ports im globalen IP-Bereich zu identifizieren, gibt es in der Umgebung auch eine Vielzahl anderer Scan-Aktivitäten. Beispielsweise gibt es Scanner, die nach offenen Verzeichnissen und offenen DNS-Resolvern suchen. Einige Unternehmen dokumentieren ihre Scan-Aktivitäten vollständig, viele jedoch nicht.

„Aggressives Scannen“ ist eine nicht autorisierte oder umfangreiche Scan-Aktivität, die die Leistung eines Netzwerks potenziell beeinträchtigt. Sie kann zu einem Denial of Service in einem Netzwerk führen oder, wie im Fall von Decoy Dog, falsche Sicherheitsereignisse hervorrufen.[18] Aggressives Scannen nützt dem Betreiber auf Kosten von Netzwerken, deren Besitzer dieser Aktivität nicht zugestimmt haben. Im April 2023 wendeten die Sicherheitsteams von Netzwerken, in denen Decoy Dog erkannt wurde, erhebliche Ressourcen auf, um die Ursache dieser DNS-Abfragen zu finden und sicherzustellen, dass ihre Systeme nicht kompromittiert waren. Diese Abfragen waren besonders besorgniserregend, weil sie überwiegend von Firewalls ausgingen und die Firewall-Branche in den letzten Monaten verstärkte Besorgnis über Angriffe auf Firewalls geäußert hatte.[19]

Die Art und Weise, wie Decoy Dog-Abfragen bei unseren Resolvern eingingen, und warum sie ein Signal auslösten, das einem gezielten Malware-C2-Beacon ähnelte, ist kompliziert. Um Cybersicherheitsfachleute dabei zu unterstützen, ähnliche Aktivitäten zu erkennen, stellen wir in Abbildung 17 eine kurze Erklärung und eine Illustration bereit.

Damit Infoblox Decoy Dog-DNS-Abfragen erhält, muss ein Kundennetzwerk Infoblox als DNS-Anbieter nutzen. Außerdem muss der Kunde über Sicherheits-Appliances wie Firewalls verfügen, die sowohl über die Filterung eingehender URLs als auch DNS-Weiterleitung von diesem Gerät an unsere Resolver verfügen. Allein diese Kriterien sind restriktiv. Wenn sie erfüllt sind, findet der folgende Ablauf statt:

- Der Scanner versucht, Inhalte für das Malware-C2 direkt von einer IP-Adresse innerhalb des Netzwerks abzurufen. Dies geschieht, obwohl es sich bei dieser DNS-C2-Kommunikation nicht um Webinhalte handelt.
- Die Sicherheits-Appliance fängt die Anfrage ab und versucht, den Domainnamen aufzulösen.
- Die DNS-Anfrage wird an Infoblox weitergeleitet, wo die Abfrage gelöst und die Antwort ausgegeben wird. Wenn sich die Domain auf einer vom Kunden konfigurierten DNS-Blockliste befindet, werden keine Ergebnisse ausgegeben.
- Handelt es sich bei der vom Anbieter gescannten Domain nicht um Decoy Dog oder andere Malware, wird sie aufgelöst und der Inhalt der Website entsprechend der Firewall-Regeln an den Scanner ausgegeben.

---

17 Aggressive Internet Wide Scanners: Network Impact and Longitudinal Characterization, Mai 2023, Anand, Dainotti, Sippe, Kallitsis. https://arxiv.org/pdf/2305.07193.pdf

18 https://live.paloaltonetworks.com/t5/general-topics/spurious-hits-from-the-expanse-webcrawler/td-p/447239 , zuletzt aufgerufen am 11. Juni 2023

19 https://blog.talosintelligence.com/state-sponsored-campaigns-target-global-network-infrastructure/, zuletzt aufgerufen am 11. Juni 2023

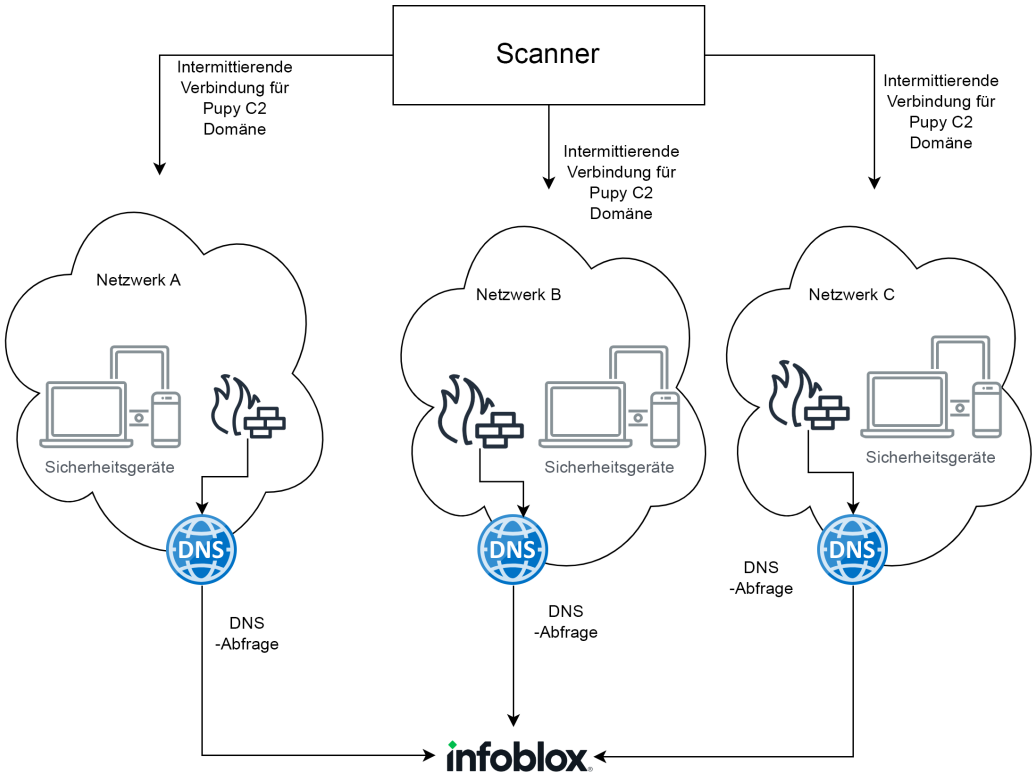


Abbildung 17. Abfragen für Decoy Dog-DNS-C2-Domains wurden von Geräten in verschiedenen Netzwerken an Infoblox-Resolver gesendet. Sie wurden durch einen kommerziellen Scanner verursacht und intermittierend ausgelöst.

Infoblox hat festgestellt, dass der Anbieter auch dann Scans durchführt, wenn die IP-Adresse keine bekannten offenen Ports hat, und dass er zusätzlich zu den üblichen Ports seltene Ports verwendet. Wir wissen nicht, wie der Anbieter entscheidet, welche IP-Adressen und Ports verwendet werden sollen. Die Folge eines wahllosen, aggressiven Scannens dieser Art ist, dass sehr sensible Geräte kompromittiert erscheinen können, obwohl sie es nicht sind. Während der Anbieter offenbar umfassend und kontinuierlich nach Inhalten scannt, beobachtete Infoblox DNS-Abfragen nur, wenn die oben genannten Kriterien erfüllt waren. Obwohl die Anzahl der vom Anbieter durchgeführten Scans sehr hoch war, was auf aggressives Scannen hindeutet, lösten wir im Laufe der Zeit immer wieder nur eine kleine Anzahl von Abfragen auf. Diese Art der Konfiguration bietet einem Akteur auch die Möglichkeit, bestimmte Netzwerke auszukundschaften; wir beschreiben diese in Anhang H.

Infoblox Intelligence erfasst historische Aufzeichnungen aller DNS-Aktivitäten und verwendet diese, um aggregierte Statistiken der Domain-Aktivitäten in unseren Netzwerken und im globalen DNS zu erstellen und zu verwalten. Wir verwenden diese Aggregationen, um ein breites Spektrum an Bedrohungen zu identifizieren, einschließlich anomalem Verhalten, das mit Malware-C2-Beacons übereinstimmt. Insbesondere suchen wir nach Domains, für die im Laufe der Zeit Abfragen in einer ungewöhnlich großen Anzahl von Kundennetzwerken auftreten, die auf Datenexfiltration hindeutende Subdomains haben und die im Verhältnis zu ihrem erwarteten Verhalten eine geringe Anzahl an Abfragen aufweisen. Dafür verwenden wir die Statistiken aller Domains, die wir über mehrere Jahre hinweg beobachtet haben, sowie Billionen von DNS-Abfragen.

Einmal entdeckt, sehen Decoy Dog und andere Malware-C2-Beacons sehr verdächtig aus. Es ist jedoch sehr schwierig, sie zu entdecken. DNS-Traffic ist naturgemäß sehr variabel und enthält einen hohen Prozentsatz an Ausreißern, also Domains, die selten gesehen werden und eine Domainnamen-Struktur aufweisen, die auf Datenexfiltration schließen lässt. Allerdings sind DNS-Exfiltration und Beaconing außerhalb etablierter Penetrationstest-

Aktivitäten sehr selten. Darüber hinaus unterscheidet sich die DNS-Signatur von Penetrationstests von Malware-C2-Beacons. Decoy Dog erwies sich zwar als DNS-C2-Kommunikation einer Variante des Pupy-RAT, eines Systems mit hohem Volumen, aber es schien ein unauffälliges Beacon zu sein, da der Datenverkehr vom Sicherheitsanbieter in die Netzwerke eingeschleust wurde.

Obwohl die Decoy Dog-Abfragen an unsere Resolver vom Scanner initiiert wurden, wurden sie aufgrund des ungewöhnlichen Verhaltens der Decoy Dog-Nameserver erkannt. Wie in unserem vorherigen Paper offengelegt, reagierten die Nameserver von Decoy Dog auf wiederholte Anfragen, wenngleich manchmal mit Unterbrechungen. Dies passt nicht zu Pupy und anderen verschlüsselten Kommunikationsprotokollen. Wir haben nun außerdem gelernt, dass die Controller auf jede korrekt formulierte Abfrage antworten. Das kombinierte Verhalten führte dazu, dass unsere Systeme einen intermittierenden Beacon mit geringem Volumen erkannten. Diese Art des Scannens und der offenen DNS-Weiterleitung innerhalb eines Netzwerks birgt zusätzliche Sicherheitsrisiken für Unternehmen. Indem ein Angreifer einer externen Partei ermöglicht, DNS-Abfragen von innerhalb eines Netzwerks auszulösen, kann er ein Netzwerk auskundschaften. Wir beschreiben diese Sicherheitslücke in Anhang H ausführlicher.

Zusammenfassung

Decoy Dog ist eindeutig eine ernsthafte Bedrohung. Eine Handvoll Bedrohungsakteure nutzt das Toolkit seit über einem Jahr und die einzigen dokumentierten Erkennungen stammen aus der Überwachung von DNS-Daten. Er wird bei sehr gezielten Operationen eingesetzt und wir haben nur beobachtet, dass seine Controller mit einer sehr begrenzten Anzahl aktiver Clients interagieren. Obwohl wir viel über Decoy Dog herausfinden konnten, wird es eine ernsthafte Bedrohung bleiben, bis die Sicherheitslücken, über die es Fuß fassen konnte, identifiziert und entschärft wurden.

Nach unserer erstmaligen Enthüllung von Decoy Dog haben die Bedrohungsakteure auf verschiedene Arten reagiert, um anhaltenden Zugang zu den Systemen sicherzustellen, die ihnen zum Opfer gefallen sind. Diese Reaktionen umfassten das Ändern des DNS-Antwortverhaltens von Controllern, das Hinzufügen von Geofencing-Beschränkungen zu Controllern und das Umziehen von Clients auf neue Controller. Trotz dieser Anpassungen ist Infoblox den Akteuren weiterhin auf der Spur und konnte mehr über Decoy Dog und die Unterschiede zum Pupy-RAT in Erfahrung bringen.

Die Änderungen, die an Pupy vorgenommen wurden, um Decoy Dog zu kreieren, sind beträchtlich und deuten auf einen besonders raffinierten Bedrohungsakteur hin. Diese Änderungen umfassen:

- Pupy wurde in Python 2.7 geschrieben. Decoy Dog benötigt Python 3.8 und beinhaltet zahlreiche Verbesserungen, darunter Windows-Kompatibilität und verbesserte Speicherabläufe.
- Pupy hat ein sehr begrenztes Kommunikationsvokabular. Decoy Dog erweitert dieses Vokabular durch das Hinzufügen mehrerer neuer Kommunikationsmodule erheblich.
- Decoy Dog reagiert auf Wiederholungen früherer DNS-Abfragen, Pupy nicht.
- Pupy antwortet nicht auf Wildcard-DNS-Anfragen, Decoy Dog jedoch schon. Dadurch verdoppelt sich im Wesentlichen die Anzahl der Auflösungen im Passive DNS. Tatsächlich antwortet Decoy Dog auf DNS-Anfragen, die nicht der Struktur einer gültigen Kommunikation mit einem Client entsprechen.
- Decoy Dog bietet die Möglichkeit, beliebigen Java-Code auszuführen, indem es ihn in einen JVM-Thread einschleust, und fügt eine Reihe neuer Methoden hinzu, um die Persistenz auf dem Gerät eines Opfers aufrechtzuerhalten.

Die Raffinesse dieser Änderungen macht die Entscheidung von Decoy Dog, auf jede korrekt beschaffene Abfrage zu antworten, noch merkwürdiger. Obwohl diese Entscheidung auf den ersten Blick wie ein Fehler erscheint, gibt es wahrscheinlich noch unbekannte Gründe dafür. Derzeit ist dieser Sachverhalt nur ein weiteres Mysterium, das Decoy Dog umgibt.

Während die Geheimnisse rund um Decoy Dog in Zukunft weiter untersucht werden, sollten Cybersicherheitsfachleute Folgendes beachten:

- IPs in Pupy und Decoy Dog sind verschlüsselte Daten. Sie stellen keine echten IPs dar, die für die Kommunikation verwendet werden. Alle Verbindungen zu echten IPs, die mit Malware in Zusammenhang stehen, sind unecht.
- Obwohl die in DNS-Antworten ausgegebenen IPs nicht aussagekräftig sind, enthalten die DNS-Abfragen und -Antworten selbst aussagekräftige Informationen, die zur Nachverfolgung verwendet werden können. Das Kommunikationsvolumen ist jedoch gering, was bedeutet, dass ein langer Protokollverlauf erforderlich ist, um erkannte Kommunikation nachzuverfolgen.
- Die Wildcard-Antworten des Toolkits in Kombination dem aggressiven Scannen des Sicherheitsanbieters können fälschlicherweise den Anschein einer Kompromittierung erwecken.
- Es gibt eine YARA-Regel, die den Decoy Dog-Client auf einer Opfermaschine erkennen kann. Sie ist in der Lage, Decoy Dog von der öffentlich verfügbaren Version von Pupy zu unterscheiden.

Decoy Dog wurde ausschließlich mithilfe von Algorithmen zur DNS-Bedrohungserkennung enttarnt. Bis heute gibt es keine öffentlichen Informationen, die Erkennungen der Malware selbst beschreiben, und der volle Umfang ihrer Fähigkeiten ist noch nicht bekannt.
Die Tatsache, dass sie so lange unentdeckt agiert hat, unterstreicht eine Schwachstelle, die auftritt, wenn sich die Branche zu sehr auf Malware-basierte Erkennung verlässt. DNS-Erkennung und -Reaktion ist derzeit die einzige Möglichkeit, sich gegen Decoy Dog zu verteidigen, und könnte auch dann noch die beste Option sein, wenn die Schwachstellen der Opfer und Decoy Dog selbst vollständig bekannt sind.

Indikatoren

Die Decoy Dog-Indikatoren für die in diesem Bericht beschriebenen Controller und Proben sind unten aufgeführt und in unserem offenen Github-Repository verfügbar.[20]

Domaingruppe	Eigenschaften
ads-tm-glb[.]click	Decoy Dog-C2-Domain
allowlisted[.]net	Decoy Dog-C2-Domain
atlas-upd[.]com	Decoy Dog-C2-Domain
cbox4[.]ignorelist[.]com	Decoy Dog-C2-Domain
claudfront[.]net	Decoy Dog-C2-Domain
hsdps[.]cc	Decoy Dog-C2-Domain
j2update[.]cc	Decoy Dog-C2-Domain

20 https://github.com/infobloxopen/threat-intelligence/tree/main/cta_indicators

maxpatrol[.]net	Decoy Dog-C2-Domain
nsdps[.]cc	Decoy Dog-C2-Domain
rcmsf100[.]net	Decoy Dog-C2-Domain
13[.]248[.]169[.]48	Decoy Dog-C2-Nameserver-IP
156[.]154[.]132[.]200	Decoy Dog-C2-Nameserver-IP
194[.]31[.]55[.]85	Decoy Dog-C2-Nameserver-IP
5[.]199[.]173[.]4	Decoy Dog-C2-Nameserver-IP
5[.]252[.]176[.]63	Decoy Dog-C2-Nameserver-IP
5[.]252[.]176[.]22	Decoy Dog-C2-Nameserver-IP
5[.]252[.]179[.]18	Decoy Dog-C2-Nameserver-IP
67[.]220[.]81[.]190	Decoy Dog-C2-Nameserver-IP
69[.]65[.]50[.]194	Decoy Dog-C2-Nameserver-IP
69[.]65[.]50[.]223	Decoy Dog-C2-Nameserver-IP
70[.]39[.]97[.]253	Decoy Dog-C2-Nameserver-IP
83[.]166[.]240[.]52	Decoy Dog-C2-Nameserver-IP
4996180b2fa1045aab5d36f46983e91dadeebf d4f765d69fa50eba4edf310acf	Decoy Dog-Binärdatei SHA256
ab8e333ef9bc5c5a7d1ed4cab08335861e150 b0639d3d0ca4c30b7def5cdccde	Decoy Dog-Binärdatei SHA256
ad186df91282cf78394ef3bd60f04d859bcaccc bcdcbfb620cc73f19ec0cec64	Decoy Dog-Binärdatei SHA256
6c8f413111f1abfee788dad4ee7cca37e0c259 7cca66d155af958c535faf55cc	Decoy Dog-Binärdatei SHA256
0375f4b3fe011b35e6575133539441009d015 ebecbee78b578c3ed04e0f22568	Decoy Dog-Binärdatei SHA256
6c8f413111f1abfee788dad4ee7cca37e0c259 7cca66d155af958c535faf55cc	Decoy Dog-Binärdatei SHA256
t1fde0f101c9395f39ecd16430b41041a59107 c73c904087309fb8d0e8d87e0077129f3f	Decoy Dog-Telfhash-Signatur[21]

21 https://github.com/trendmicro/telfhash

ANHANG A: VERARBEITUNG VON CLIENT-BEFEHLEN

Abbildung 18 zeigt den Betriebszyklus des im Paper beschriebenen Clients. Der Client wechselt wiederholt zwischen dem Ruhezustand, dem Polling des Servers und dem Reagieren auf Befehle.

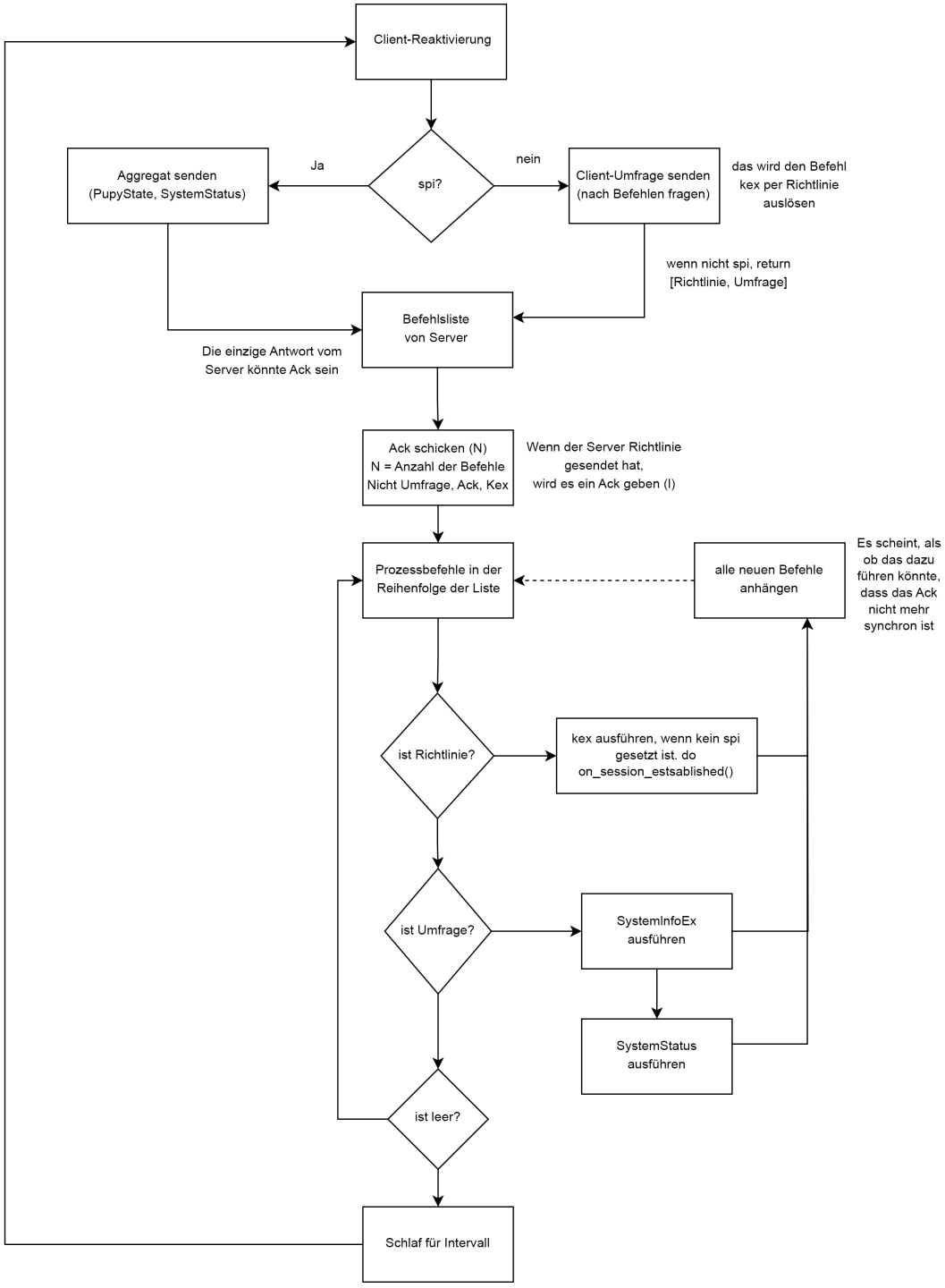


Abbildung 18. Client-Workflow.

ANHANG B: KOMMUNIKATIONSNUTZLAST-STRUKTUR

Die Struktur der verschlüsselten Nutzlast für Client und Server ist identisch, aber es gibt Unterschiede bei der Verarbeitung. Insbesondere enthält der Client, wie zuvor beschrieben, bei jeder Abfrage neben der Datennutzlast 13 Bytes an Client-Informationen.

Der Client und der Server verwenden beide den Befehlsbegriff für den Informationstyp, den sie an den Empfänger übermitteln. Wenn also der Client beim Aufwachen den Server kontaktiert, wird das als Client-Befehl betrachtet. Die Befehle werden registriert, sodass der Client oder Server eine bestimmte Verarbeitung auf die Daten anwenden kann. In einer einzelnen Kommunikation kann es mehr als einen Befehl geben, vom Client aus kommt dies jedoch selten vor.

Die Nutzlast, die zur Codierung und Übertragung gesendet wird, hat das folgende Format:

- Eine 4-Byte-Prüfsumme
- Verkettete Befehlspakete, die eine 1-Byte-Befehlskennung und einen variablen befehlsabhängigen Datenteil enthalten

Die Gesamtlänge der Nutzlast darf 52 Byte nicht überschreiten.

ANHANG C: REKONSTRUKTION VON CLIENTS AUS PASSIVEN DATEN

Wie bereits beschrieben, umfassen Pupy-Abfragen verschlüsselte Daten und zwei codierte Werte, Nonce und SPI, die eine gewisse Sicherheit bieten und es dem Server ermöglichen, die Kommunikation mit den Clients zu ordnen. Der SPI-Wert wird speziell zur Identifizierung einer laufenden Sitzung innerhalb des Servers verwendet und ist in Abfragen nach einem erfolgreichen Schlüsselaustausch vorhanden. Daher ist es fast garantiert, dass Abfragen, die denselben SPI enthalten und zeitlich nah beieinander liegen, vom selben Client stammen. Andererseits wird ein einzelner Client im Laufe der Zeit viele Sitzungen und viele SPI-Werte haben, sodass der SPI allein keine Unterscheidung der Clients ermöglicht. Stattdessen verwenden wir die Nonce-Werte, um die Client-Kommunikationsvorgänge zu trennen.

Wenn der Client initialisiert wird, generiert er nach dem Zufallsprinzip einen 32-Bit-Nonce-Wert, der als Ausgangspunkt dient. Bei jedem Paket wird diese Nonce um die Länge der übertragenen Daten erhöht. Der Server verwendet die Nonce als kleine Sicherheitsüberprüfung, indem er sicherstellt, dass sie sich mit jeder empfangenen Abfrage erhöht, aber ihr Hauptzweck ist die korrekte Entschlüsselung und Interpretation der zugrundeliegenden Kommunikation. Aus einer Reihe beobachteter Pupy-Abfragen können wir diese Nonce-Werte decodieren und die nächste Nonce in der Serie berechnen, wie unten in Abbildung 19 dargestellt.

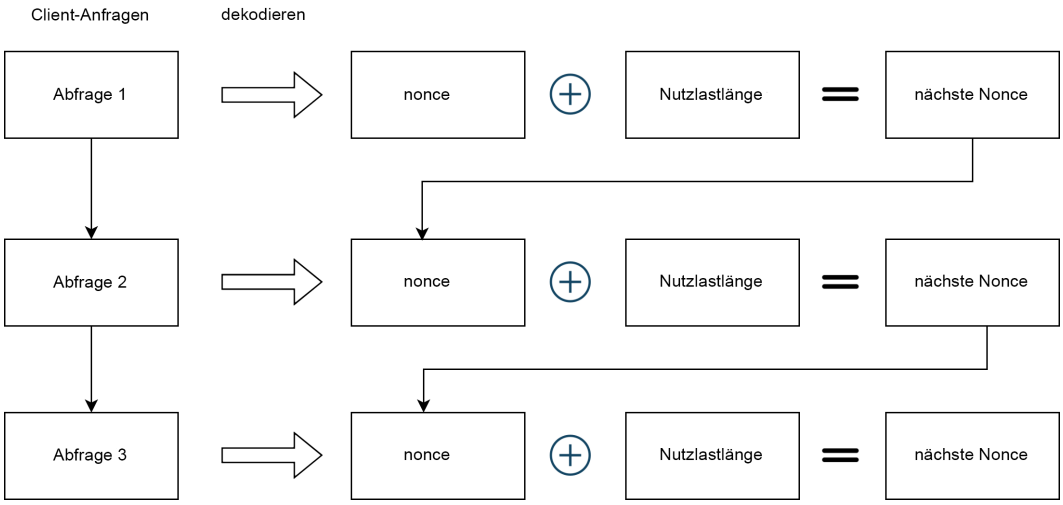


Abbildung 19. Die Beziehung von Nonce-Werten innerhalb einer Reihe von Pupy-Abfragen.

So können wir sowohl die Abfragen eines einzelnen Clients ordnen als auch bestätigen, dass eine Reihe von Abfragen zu einem einzigen Client gehört. Bei der passiven Erfassung einer Pupy-Bereitstellung können die Abfragen von zahlreichen Clients stammen und sich zeitlich überschneiden. Aufgrund des Aufbaus der Nonce können wir diese Beobachtungen jedoch mit einem hohen Maß an Sicherheit in separate Client-Aktivitäten aufteilen. Da die Nonce zur Verschlüsselung der Nutzlast verwendet wird, hat der Entwickler sie mit einem leistungsstarken Zufallszahlengenerator erstellt. Dadurch wird sichergestellt, dass jeder Client einzigartige Start-Nonce-Werte erzeugt.[22] Die Nonce wird jedes Mal neu erstellt, wenn der Client neu gestartet wird.

Die zusätzliche Sicherheit für die Verschlüsselung bietet auch einen Mechanismus zur Unterscheidung von Clients in aggregierten Beobachtungen. Dazu berechnen wir für jede Abfrage sowohl die codierte Nonce als auch den nächsten Nonce-Wert. Anschließend verketten wir die Abfragen mithilfe der sequentiellen Nonce-Werte, wie in Abbildung 20 unten dargestellt. Während die zugrundeliegenden Daten verschlüsselt bleiben, können wir die Anzahl der Clients schätzen und Beobachtungen über die Dauer ihrer Aktivität machen. Weiterhin können wir mithilfe der Nutzlastlängen und dem Client-übergreifenden Vergleich von Zeitreihen Informationen über die Kommunikation selbst ableiten.

22 Es gibt seltene Wahrscheinlichkeiten, dass dieselbe Nonce gleichzeitig von zwei verschiedenen Clients generiert werden könnte.

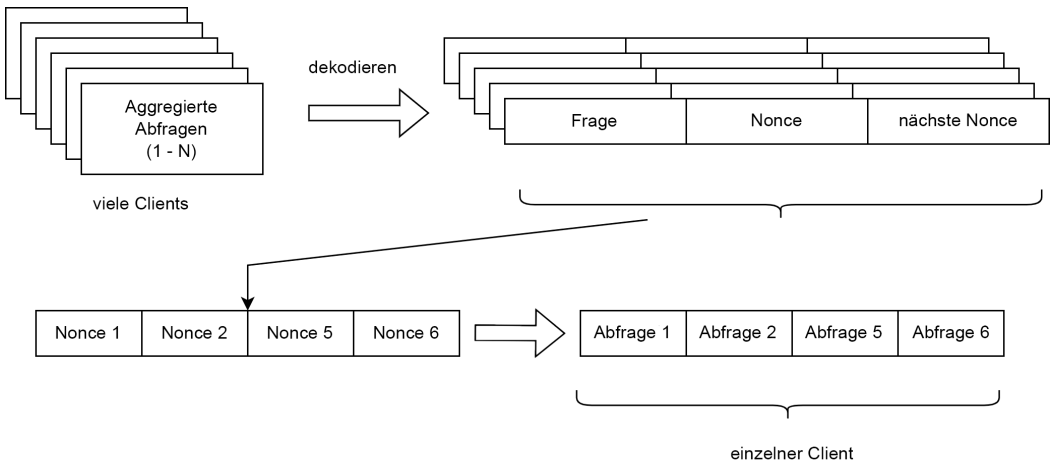


Abbildung 20. Trennen eines Client-Threads von Abfragen von einem aggregierten Satz von Beobachtungen unter Verwendung der Nonce-Werte.

Bei dieser Art der Ausnutzung gibt es zwei Herausforderungen: Änderungen im DNS-Resolver des infizierten Clients und verworfene Pakete. Standardmäßig verwendet Pupy den Standard-DNS-Resolver des Clients und die Wahl des Resolvers unterliegt möglicherweise nicht der Kontrolle des Akteurs. Im Falle von Client-Roaming verwendet der Client je nach lokaler Umgebung möglicherweise unterschiedliche rekursive Resolver. In Unternehmensnetzwerken kann DNS-Infrastruktur von Anbietern wie Infoblox verwendet werden, bei der die DNS-Abfragen unabhängig von den Einstellungen des Clients über die rekursiven Resolver des Unternehmens gezwungen werden.[23] Wenn DNS über UDP transportiert wird, ist außerdem ein Paketverlust unvermeidlich. Das Ergebnis daraus ist, dass wir wahrscheinlich nicht jede Abfrage nur im Passive DNS beobachten können, wodurch möglicherweise erhebliche Lücken in der wiederhergestellten Nonce-Kette entstehen.

Wir können die Client-Threads jedoch immer noch rekonstruieren, indem wir uns die Tatsache zunutze machen, dass die Nonce ein zufällig generierter Wert ist. Der Entwickler hat einen starken Zahlengenerator verwendet, der sicherstellt, dass es extrem unwahrscheinlich ist, dass unabhängige Pupy-Clients einen Nonce-Wert teilen. Da außerdem jeweils nur 52 Byte an Daten übertragen werden können und der Nonce-Wert um die Nutzlast erhöht wird, ist es unwahrscheinlich, dass sich zwei unabhängig generierte Nonce-Ketten überschneiden. Dadurch können Clients durch das Ordnen der Nonce-Werte und die Gruppierung statistisch ähnlicher Nonce-Werte getrennt werden. Ein einzelner Client hat jeweils nur eine Nonce, sodass wir die Anzahl der aktiven Clients zu jedem beliebigen Zeitpunkt schätzen können. Wie wir im Hauptteil des Papers zeigen, haben wir festgestellt, dass diese Technik bei der Wiederherstellung von Decoy Dog-Client-Abfrageketten sehr effektiv ist.

23 Who is Answering My Queries: Understanding and Characterizing Interception of the DNS Resolution Path, Baojun Liu, et al. 2018, https://www.usenix.org/conference/usenixsecurity18/presentation/liu-baojun

ANHANG D: NUTZLAST-SIGNATUREN

Die Tabellen in diesem Abschnitt enthalten die Nutzlastlängen für bestimmte Befehle, die häufig in Pupy-Kommunikation beobachtet werden. Insbesondere stellen sie die verschlüsselte Nutzlastlänge für jeden Standard-Client-Befehl bereit. Server-Nutzlasten sind flexibler als die der Clients; die gängigsten sind unten aufgeführt.

Client-Befehl	Nutzlastlänge
Client check-in (initial)	18
Ack	19
Client check-in (rare variant)	22
System status	24
Online status	27
Client check-in (in session)	27
Port quiz	35
System information extended	39
Key exchange	47, 48

Tabelle 2. Client-Befehle und Nutzlastlängen.

Serverbefehl	Nutzlastlänge
Ack	6
Need session: policy, poll	42
Session incomplete: ack, policy	34
Error: message, policy, poll	44
Need system info: poll	15
Key exchange	62, 63
Exit	7

Tabelle 3. Allgemeine Serverbefehle und Nutzlastlängen.

ANHANG E: FEHLERVERARBEITUNG

Pupy umfasst eine benutzerdefinierte Behandlung für eine Vielzahl von Fehlern, die auf dem Server auftreten können. Eine Domain, die nicht ordnungsgemäß decodiert oder wiederholt wird, führt zu einer NXDOMAIN-Antwort vom Server. Der folgende Codeausschnitt zeigt die Verarbeitung der Serverabfrage. Wenn keine Antwort ausgegeben wird, wird eine NXDOMAIN-Antwort ausgegeben.

```
answers = self.process(qtype, qname.stripSuffix(self.domain).idna()[:-1])
klass = SUPPORTED_METHODS[qtype]


if answers:
    for answer in answers:
        reply.add_answer(RR(qname, qtype, rdata=klass(answer), ttl=600))


    if self.edns:
        reply.add_ar(EDNS0(udp_len=512))
else:
    reply.header.rcode = RCODE.NXDOMAIN
```

Abbildung 21. Pupy-Server-Quellcode, der Client-Abfragen verarbeitet.

In Decoy Dog führen viele Client-Abfragen, die zu einem NXDOMAIN vom Server führen sollten, stattdessen zur Ausgabe einer Antwort, die in der Regel aus 15 IP-Adressen besteht. Dies scheint auf eine Codeänderung zurückzuführen zu sein, durch die Decoy Dog auf eine Vielzahl möglicher Fehler intern mit einer DnsCommandServerException reagiert. Die DNSCommandServerException wird zu einer Antwort an den Client führen, in der die Art des aufgetretenen Fehlers angegeben wird und der Client angewiesen wird, einen neuen Schlüsselaustausch durchzuführen, gefolgt von der Übertragung von Systeminformationen. Der Codeblock für diese Fehlerverarbeitung ist unten dargestellt.

```
except DnsCommandServerException as e:
    nonce = e.nonce
    version = e.version
    responses = [e.error, Policy(self.interval, self.kex), Poll()]
    emsg = 'Server Error: {} (v={})'.format(e, version)
    logger.debug(emsg)
    if node:
        node.warning = emsg
```

Abbildung 22. Quellcode des Pupy-Servers, der einen Fehler an den Client ausgibt.

Bei normaler Kommunikation zwischen einem Pupy-Server und einem Client wird diese Art von Ausnahme ausgelöst, wenn es keine aktive Sitzung für einen bekannten Client gibt. Sie wird auch verwendet, wenn die Client-Nutzlast ungültig ist oder eine falsche Prüfsumme hat. In allen anderen Fällen ist das Ergebnis NXDOMAIN.

ANHANG F: ANALYSE VON BINÄRDATEIPROBEN

Pupy-Client-Binärdateien

Wenn der Pupy-Server zum ersten Mal eingerichtet wird, kompiliert er Pupy-Bibliotheksdateien und erstellt für jede Architektur eine statische Vorlagendatei. Diese Vorlagendateien sind komprimiert, stark verschlüsselt und von sämtlichen Symbolen befreit.

Client-Binärdateien können dann manuell über die Verwendung von pupygen.py auf dem Server erstellt werden. Das Skript erstellt C2-spezifische Binärdateien, indem spezifische Konfigurationsbytes (Remote-Host, Transporttyp, Debug-Flag usw.) in die statische Vorlage gemarshallt werden, die der Zielarchitektur und dem Dateityp entspricht.

Die Pupy-Client-Binärdateien bieten eine Vielzahl von fortschrittlichen Funktionen und sind für praktisch jede Plattform geeignet, einschließlich Windows, macOS, Linux, Solaris und Android. Insbesondere können sie im Speicher verbleiben, mit dem Server interagieren, vollständige Reverse-Shell-Funktionen bieten, dateilose Kopien erstellen usw. Wenn die Binärdatei ausgeführt wird, erstellt sie Kopien von sich selbst im Speicher, um einer Entdeckung zu entgehen und sich selbst widerstandsfähiger gegen Techniken zum Beenden von Prozessen zu machen.

Beispiel einer Java-Injection-Funktion

Decoy Dog-Binärdateien enthalten eine Reihe neuer Funktionen im Zusammenhang mit Java-Injection. Dies ist ein Beispiel für eine dieser Funktionen.

```
undefined8 FUN_00105903(void)
{
  int iVar1;
  long lVar2;
  long lVar3;
  long lVar4;
  undefined8 uVar5;
  char *pcVar6;
  undefined local_20 [8];
  undefined8 local_18;

  local_18 = 0;
  if (DAT_005fbda0 == 0) {
     pcVar6 = "JVM was not loaded yet";
  }
  else {
       jvm_address = check_jvm_is_running(0);

       if (jvm_address == 0) {
       return 0;
       }
       classloader_address = find_classloader(lVar2);
       if (classloader_address == 0) {
       pcVar6 = "Preferred classloader was not found";
       }
       else {
       thread_class_address = find_jv_thread(lVar2);
       if (thread_class_address == 0) {
       pcVar6 = "Could not find Thread class";
       }
       else {
       iVar1 =
inject_in_thread(jvm_address,thread_class_address,"currentThread","()Ljava/lang/Thread;",&lo
cal_18);
       if (iVar1 == 0) {
       iVar1 = inject_in_class(jvm_address,local_18,"setContextClassLoader","(Ljava/lang/ClassLoader;)V",
                                             local_20,classloader_address);
       if (iVar1 == 0) {
          uVar5 = (*DAT_005fb748)(1);
          return uVar5;
       }
       pcVar6 = "Iteration failed";
       }
       else {
       pcVar6 = "Could not find current JVM Thread";
       }
  return 0;
```

Abbildung 23. Teilweise zerlegte Decoy Dog-Funktion, die versucht, den aktuell laufenden JVM-Thread für die Injektion zu finden.

ANHANG G: YARA-REGEL FÜR DECOY DOG

Die folgende YARA-Regel kann verwendet werden, um die Decoy Dog-Proben zu erkennen, die wir ab Juli 2023 beobachtet haben.

```
/*
This rule only detects Decoy Dog. It was adapted from Florian Roth's Pupy Rule
original author : Florian Roth / @neo23x0
original link : https://github.com/Neo23x0/signature-base/blob/master/yara/gen_pupy_rat.yar
*/

/* Rule Set ----------------------------------------------------------------- */
import "elf"
import "pe"

rule DecoyDog_Backdoor {
   meta:
     description = "Detects Decoy Dog backdoor"
     license = "Detection Rule License 1.1 https://github.com/Neo23x0/signature-
base/blob/master/LICENSE"
     author = "Infoblox Inc."
     reference = "https://github.com/n1nj4sec/pupy-binaries"
     date = "2023-07-11"

   strings:
     $x1 = "reflectively inject a dll into a process." fullword ascii
     $x2 = "ld_preload_inject_dll(cmdline, dll_buffer, hook_exit) -> pid" fullword ascii
     $x3 = "LD_PRELOAD=%s HOOK_EXIT=%d CLEANUP=%d exec %s 1>/dev/null 2>/dev/null" fullword ascii
     $x4 = "reflective_inject_dll" fullword ascii
     $x5 = "ld_preload_inject_dll" fullword ascii
     $x6 = "get_pupy_config() -> string" fullword ascii
     $x7 = "[INJECT] inject_dll. OpenProcess failed." fullword ascii
     $x8 = "reflective_inject_dll" fullword ascii
     $x9 = "reflective_inject_dll(pid, dll_buffer, isRemoteProcess64bits)" fullword ascii
     $x10 = "linux_inject_main" fullword ascii
     $x11 = "jvm.PreferredClassLoader" fullword ascii
     $x12 = "jvm.JNIEnv capsule is invalid" fullword ascii

   condition:
     (3 of them and $x11 ) or (3 of them and $x12)
     or (uint16(0) == 0x5a4d and pe.imphash() == "84a69bce2ff6d9f866b7ae63bd70b163" and
$x11) or (elf.telfhash() ==
"t1fde0f101c9395f39ecd16430b41041a59107c73c904087309fb8d0e8d87e0077129f3f")
}
```

Abbildung 24. YARA-Regel zum Erkennen von Decoy Dog-Proben.

ANHANG H: AUFGEDECKTE SICHERHEITSLÜCKEN

Wenn ein Gerät so konfiguriert ist, dass es bei einer eingehenden Verbindung eine DNS-Abfrage durchführt, erlaubt es einer externen Entität, sein Verhalten und seine Ressourcen teilweise zu kontrollieren.[24] Insbesondere kann diese Konfiguration Bedrohungsakteuren eine Möglichkeit zur Auskundschaftung, offenen Auflösung und möglichen Beteiligung an einem Denial-of-Service-Angriff bieten. Da DNS eine komplexe Angelegenheit ist, sind sich sowohl Anbieter als auch Netzwerkbetreiber dieser Risiken möglicherweise nicht bewusst. Während die Sicherheits-Appliances, die die von uns erkannten Abfragen übermittelten, über neuartige Funktionen verfügen sollten, setzt die Verwendung von DNS in diesen Funktionen das Netzwerk Auskundschaftungsaktivitäten und potenziellen weiteren Bedrohungen aus.

24 https://knowledgebase.paloaltonetworks.com/KCSArticleDetail?id=kA10g000000PLRaCAO, zuletzt aufgerufen am 11. Juni 2023

Ein Gerät innerhalb eines Netzwerks, das DNS-Abfragen für eine externe Entität bereitstellt, wird als offener Resolver bezeichnet. In einigen Fällen kann es vorkommen, dass ein Gerät aufgrund einer Vielzahl von Umständen Antworten ausgibt, externe DNS-Abfragen jedoch nicht vollständig auflöst. In jedem Fall stellen solche Geräte ein Risiko für das Netzwerk selbst und für die Nutzung des Netzwerks zur Verstärkung von DDOS-Angriffen (Distributed Denial of Service) dar. Die Risiken offener DNS-Resolver sind gut dokumentiert und offene Resolver sind aufgrund dieser Risiken im Rahmen vieler Serviceverträge, auch in denen von Infoblox, untersagt.

Im Fall der Decoy Dog-Abfragen waren die Sicherheits-Appliances keine offenen Resolver, erlaubten aber dennoch einer externen Partei, DNS-Abfragen auszulösen. Diese Art der Konfiguration kann nicht für einen Verstärkungsangriff verwendet werden, kann aber von einem Bedrohungsakteur für andere Zwecke genutzt werden. Zum Beispiel kann ein Bedrohungsakteur ein Netzwerk ausspionieren wie in der Abbildung unten dargestellt. Der Akteur erstellt eine Domain und konfiguriert den entsprechenden Nameserver, um eingehende Abfragen zu protokollieren. Der Akteur verwendet dann einen Scanmechanismus, um maßgeschneiderte Domainnamen für die Verbindung mit dem Netzwerk zu senden. Im Fall einer Suche nach einem offenen Resolver kann es sich hierbei um DNS-Abfragen handeln. Im Falle von Decoy Dog waren es HTTPS-Verbindungen. Unter beiden Umständen generiert das interne Gerät eine DNS-Abfrage, die an den vom Akteur kontrollierten Nameserver gesendet wird. Der Akteur kann dann den Domainnamen und die originale IP-Adresse mit der erhaltenen Abfrage verknüpfen. Obwohl diese Arten von Angriffen bei jedem Versuch nur eine begrenzte Menge an Informationen einbringen, sind sie bewährte Mechanismen, um interne Netzwerke für spätere Angriffe zu kartographieren.

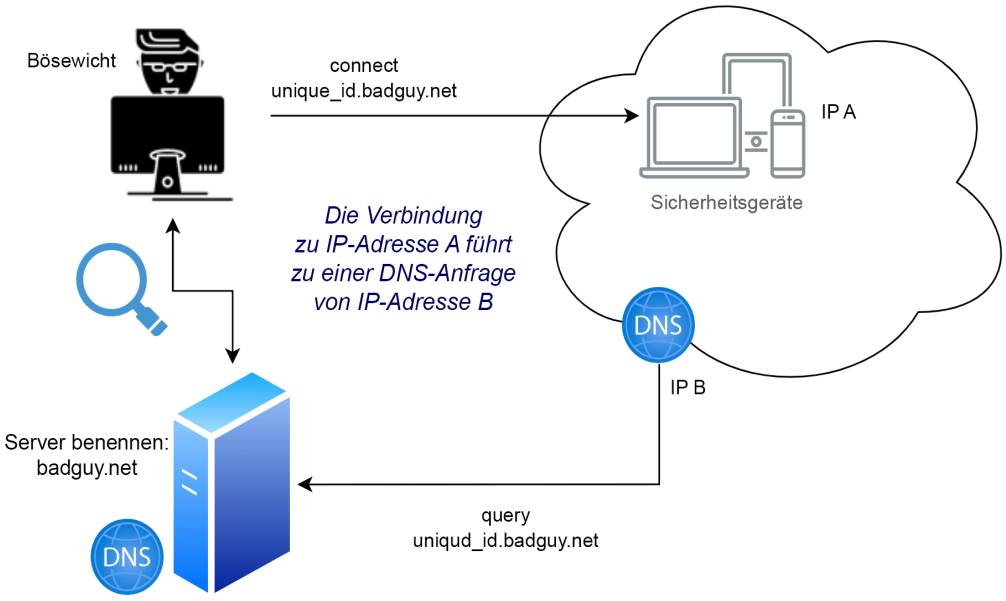


Abbildung 25. Ein Akteur führt Auskundschaftungsmaßnahmen in einem Netzwerk durch, indem er eindeutige Domainnamen erstellt, die DNS-Abfragen an seinen Nameserver erstellen.

ANHANG I: FORSCHUNGSDATEN

Für unsere Recherchen haben wir einen Pupy-Server eingerichtet und Kommunikations-vorgänge zwischen dem Server und den Clients über unsere rekursiven Resolver geleitet. Wir haben diese DNS-Abfrageprotokolle für unsere Analyse erfasst und stellen die Protokolle für Forschungszwecke zur Verfügung. Die Daten umfassen mehrere Tage unterschiedlicher Aktivität. Die meiste Zeit haben wir die Clients kontrolliert, indem wir einen Reverse-Proxy eingerichtet haben; Befehle wurden über SSL gesendet. Wir vermuten, dass dies auch bei

Decoy Dog der Fall ist. Wir haben jedoch alle verfügbaren Befehle über DNS-Antworten vom Server ausgeführt. Darüber hinaus gibt es Zeiträume, in denen mehrere Clients gleichzeitig aktiv waren und zahlreiche Client-Neustarts erfolgt sind. Der enthaltene Aktivitätsumfang sollte es ermöglichen, die hier beschriebenen Ergebnisse zu reproduzieren.

Die Daten sind in unserem öffentlichen GitHub-Repository infobloxopen verfügbar: threat-intelligence.[25] Die Abfrage-Antwortprotokolle enthalten A-Eintrag-Ergebnisse und sind in eine CSV-Datei mit den folgenden Feldern enthalten:

- timestamp: der Zeitpunkt der Abfrage in Sekunden seit der Unix-Epoche
- query: der vollständig qualifizierte Domainname, der in der Client-Abfrage übermittelt wird
- response: der vom Server ausgegebene Satz von IP-Adressen
- client_payload_len: die Anzahl der Nutzlastbytes innerhalb der Abfrage, einschließlich der Host-Informationen
- server_payload_len: die Anzahl der Nutzlastbytes innerhalb der Antwort

Das Repo enthält auch die Indikatoren in diesem Paper; weitere Indikatoren stehen Cybersicherheitsfachleuten auf Anfrage als TLP:RED-Informationen zur Verfügung. Darüber hinaus stellen wir Daten zur Verfügung, die aus Reverse-Engineering-Binärdatenproben stammen, die auf VirusTotal verfügbar sind. Dazu gehören:

- Eingebettete Konfigurationsparameter für jede Probe
- Eingebettete kryptografische Schlüssel und Passwort für jede Probe
  - BIND_PAYLOADS_PASSWORD
  - DCONFIG_PUBLIC_KEY (only for client v4)
  - DNSCNC_PUB_KEY_V2
  - ECPV_RC4_PRIVATE_KEY
  - ECPV_RC4_PUBLIC_KEY
  - SCRAMBLESUIT_PASSWD
  - SIMPLE_RSA_PUB_KEY
  - SIMPLE_RSA_PRIV_KEY
  - SSL_BIND_CERT
  - SSL_BIND_KEY
  - SSL_CA_CERT
  - SSL_CLIENT_CERT
  - SSL_CLIENT_KEY
- Eine YARA-Regel und ein TELF-Hash, die Decoy Dog-Binärdateien erkennen können

25 https://github.com/infobloxopen/threat-intelligence

infoblox.

Infoblox vereint Netzwerk- und Sicherheitslösungen für ein unübertroffenes Maß an Leistung und Schutz. Wir bieten Echtzeit-Transparenz und Kontrolle darüber, wer und was sich mit Ihrem Netzwerk verbindet, damit Ihr Unternehmen schneller arbeiten und Bedrohungen früher stoppen kann. Darauf vertrauen Fortune-100-Unternehmen und aufstrebende Innovatoren.

Hauptsitz der Gesellschaft
2390 Mission College Blvd, Ste. 501
Santa Clara, CA 95054

+1.408.986.4000
www.infoblox.com

RP-072023 v1