CHAPITRE 4

OUTILS DE SIMULATION D'INFECTION ET DE PROPAGATION D'ÉPIDÉMIE

Au cours de ce chapitre, nous allons présenter les divers modèles et outils décrivant les propagations d'agents pathogènes dans une population. En effet, ce problème, à l'origine médical, est extrêmement proche de notre problématique et plusieurs équipes de chercheurs les ont utilisés afin de décrire l'évolution de la taille de réseaux de zombies. Nous y détaillerons ensuite notre modèle et le logiciel que nous avons développé. Puis, dans une seconde partie, nous présenterons notre méthodologie au travers des divers paramètres utilisés pour nos expériences. Nous décrirons ainsi les diverses simulations effectuées. Enfin la quatrième section présente les résultats de chaque simulation. Le but de créer un outil de simulation modélisant les populations de réseaux de zombies, est de pouvoir simuler certains comportements clés impactant la vitesse de propagation et la taille du réseau de zombies. Grâce à cela, nous pouvons exploiter au mieux notre taxonomie, en déterminant quelles fonctionnalités sont les plus performantes, et donc axer les travaux de défense sur ces dernières qui auront une plus grande probabilité d'être utilisées par les réseaux de zombies.

4.1 LES MODÈLES DE SIMULATIONS D'INFECTIONS DE RÉSEAUX DE ZOMBIES

Dans cette section nous allons analyser les outils existants permettant de modéliser la propagation d'un ver informatique. Nous allons ensuite détailler le programme que nous avons créé afin modéliser la propagation d'un ou de plusieurs vers en concurrence. Enfin, nous expliciterons nos résultats.

4.1.1 LES MODÈLES EXISTANTS

Nous avons montré dans le précédent chapitre que les programmes malveillants qui créent des réseaux de zombies évoluent. Nous avons également montré que certaines fonctionnalités tendent à disparaître alors que d'autres tendent à perdurer. Nous émettons l'hypothèse que les fonctionnalités qui apportent plus d'avantages pour le réseau auront tendance à survivre. Ainsi, notre objectif est de modéliser l'impact que peuvent avoir diverses fonctionnalités sur l'efficacité d'un réseau de zombies. Pour ce faire, nous observerons les incidences sur la taille du réseau (nombre de zombies) ainsi que sur la vitesse de propagation de l'infection. En effet, un réseau qui se construit plus vite et qui obtient plus de zombies sera plus efficace pour remplir ces objectifs, quels qu'ils soient. Enfin, nous modéliserons la concurrence entre divers réseaux afin de montrer l'impact réel de ces fonctionnalités.

Notre objectif étant de mesurer la taille d'un réseau et la vitesse de la propagation de l'infection, nous nous sommes tournés vers les modèles infectieux. Cependant, il existe d'autres modèles de propagation, analysés par Wainwright et Kettani (2019). Dans leur article, les auteurs font un état de l'art de l'ensemble des modèles de réseaux de zombies existants. Afin d'analyser les divers modèles existants, ils introduisent un cadre d'analyse basé sur le cycle de vie des réseaux de zombies. Pour ce faire, ils utilisent les critères suivants : Conception, Recrutement,

Interaction, Marketing et Exécution des attaques. Ils appellent ce cadre « CRIME ».

Un modèle décrivant la phase de conception devra lister les motivations de l'attaquant avec les fonctionnalités qu'il devra implémenter pour les satisfaire. La phase de recrutement correspond à l'ensemble des descriptions des méthodes de propagation et d'infection. Les interactions décrivent les communications internes ainsi que les méthodes de gestion du réseau de zombies. La phase de marketing correspond à l'ensemble des actions possibles que peut faire un attaquant pour vendre ou rentabiliser son réseau. Enfin, la partie exécution décrit les buts finaux du réseau, donc l'ensemble des attaques qu'il peut créer et leurs conséquences. Le but des auteurs est de déterminer, pour chaque modèle, quelles phases du cycle de vie il décrit.

Parmi les sept modèles analysés, seuls trois décrivent la phase de recrutement : les modèles infectieux, les modèles d'apprentissage machine et les modèles issus de la théorie des jeux. Le premier est dérivé des modèles épidémiologiques utilisés en médecine. Au départ, les modèles utilisaient un schéma S.I.C (Susceptible Infecté Contrôlé), utilisant une population homogène où tous les hôtes sont Susceptibles d'être infectés. Il existe une faible probabilité que chaque hôte devienne Infecté puis Contrôlé. Plus tard un autre modèle S.I.S (Susceptible Infecté Susceptible) est apparu pour modéliser l'utilisation de plusieurs méthodes d'exploitation par les réseaux de zombies. Ces modèles permettent donc de modéliser l'évolution de la population de zombies au cours du temps.

Le second modèle analysé est celui concernant l'apprentissage machine. Ces modèles se basent sur les données récoltées par les réseaux de pots de miels ainsi que sur les systèmes de détection d'intrusion (IDS). Leur but est de classer les données afin d'identifier le trafic malicieux d'un réseau de zombies au travers de l'ensemble des communications réseau enregistrées. Ces modèles se concentrent sur les phases de communication et de recrutement. Cependant, contrairement au premier modèle, ils ne permettent pas d'évaluer ou de prédire la taille d'un réseau de zombies.

Les modèles de la théorie des jeux évolutionnaire sont utilisés conjointement avec les modèles épidémiologiques afin d'analyser la persistance d'un réseau de zombies par rapport aux autres. Cela permet de décrire les interactions entre les possesseurs de ces réseaux, telles que la collaboration ou la compétition. Ces modèles discutent principalement des motivations et légèrement de la phase de recrutement. Ici aussi, les modèles ne permettent pas de prédire la taille ou la vitesse de propagation d'une infection.

De ce fait, nous nous sommes concentrés sur les modèles infectieux et leurs applications concernant les réseaux de zombies. Nous avons donc analysé quatre modèles épidémiologiques, en nous basant sur des articles les utilisant afin de modéliser la propagation de certains vers comme Code Red ou Morris (Zou *et al.*, 2002).

4.1.2 CRITÈRES D'ANALYSE DES MODÈLES EXISTANTS

Afin de les comparer, nous avons défini plusieurs critères. Le premier est le niveau de simplification du modèle (ou simplicité). Nous disons d'un modèle qu'il est simplifié s'il combine plusieurs facteurs en un seul afin de simplifier les modélisations et les calculs. Un tel modèle peut difficilement décrire les impacts que pourrait avoir une fonctionnalité sur l'efficacité d'un réseau de zombies.

Notre second critère est l'extensibilité du modèle. Un modèle extensible pourra prendre en compte facilement l'ajout de nouvelles fonctionnalités. Par exemple, si une nouvelle méthode de détection des victimes apparaît, le modèle extensible pourra la décrire facilement sans avoir à modifier sensiblement le modèle. Un outil non extensible engendrera une grande complexité et un besoin en calcul important pour décrire un nouveau comportement. Enfin, nous déterminerons si les modèles existants permettent de décrire la compétition entre deux réseaux de zombies.

4.1.3 ANALYSE DES MODÈLES SÉLECTIONNÉS

Les premiers modèles apparus sont les modèles S.I (Susceptible Infecté), S.I.R (Susceptible Infecté Rétabli) et S.I.S (Susceptible Infecté Susceptible). Ces modèles se basent sur l'état des individus d'une population pour modéliser la propagation d'une épidémie. Dans le modèle S.I.R, les ordinateurs vulnérables à un ver sont susceptibles, lorsqu'ils sont exploités ils deviennent infectés et lorsque le ver est supprimé ils deviennent rétablis. Dans le modèle S.I.S, les individus peuvent redevenir Susceptible si la vulnérabilité n'est pas corrigée ou si le ver utilise plusieurs méthodes de contamination. Dans une population donnée, tous les agents ne sont pas forcément susceptibles. La taille maximale de l'épidémie est donc déterminée par la proportion de machines vulnérables.

Dans ces modèles, on utilise un taux de naissance ou taux de propagation, souvent constants pour décrire la vitesse de propagation. On utilise aussi un taux de mort pour représenter les individus qui sortent temporairement du réseau (ordinateur éteint, virus supprimé, etc.). Ce modèle est très simplifié, il néglige les changements d'états internes de l'individu, abstrait le phénomène de transmission, d'infection et de récupération par les taux de naissance et de mort. Or, ces taux sont influencés par de nombreux paramètres comme la méthode de scan, la congestion du réseau, le nombre de machines qui se connectent ou se déconnectent, etc. Le modèle S.I est une version encore plus simplifiée du modèle S.I.S où l'on ne prend pas en compte la perte d'hôtes infectés.

Ces modèles sont très utilisés pour modéliser la propagation de vers informatiques utilisant une stratégie de détection aléatoire (Zou *et al.*, 2002; Staniford *et al.*, 2002; Moore *et al.*, 2003). En effet, ils supposent qu'une certaine proportion de la population est vulnérable au ver et que ce dernier va se propager aléatoirement au cours du temps. Chaque nouvel hôte va attaquer au hasard un individu de la population afin d'essayer de propager l'infection. Dans l'ensemble, ces modèles utilisent des équations différentielles pour abstraire les comportements et modéliser la

propagation de l'infection.

Dans l'ensemble de ces modèles, on ne considère pas le phénomène de patch de vulnérabilité, car la propagation d'une épidémie est bien plus rapide que ne peut l'être une réponse technique permettant de rapidement désinfecter les ordinateurs corrompus. De plus, ces outils sont très spécifiques à certaines infections et modélisent très mal certains comportements. C'est ce que montre l'équipe de Chen et Ji (2005). En effet, ils souhaitent modéliser la propagation de ver utilisant la structure du réseau pour trouver de nouvelles victimes. D'après eux, il est très difficile de le faire en utilisant simplement les modèles S.I.S classiques, car dans une propagation topologique, le taux d'infection est différent pour chaque noeud. Ainsi, ils développent un modèle basé sur les chaînes de Markov afin de mieux décrire ces phénomènes. L'équipe de Zou *et al.* (2002) en a fait de même pour correctement simuler l'attaque du ver Code Red. Ils ont rajouté plusieurs fonctions au modèle de base afin de décrire la chute du taux d'infection du vers. Ils ont ainsi créé une nouvelle version du modèle S.I.R, plus complexe et plus précise que la version basique.

Un modèle un peu plus proche du fonctionnement de ces vers est celui décrit par (Chen *et al.*, 2003). Dans leur article, ils définissent un nouveau modèle de propagation des vers informatique, appelé AAWAP et utilisant le scan aléatoire. Pour ce faire, ils se basent sur les épidémies engendrées par les vers Morris, Code Red et Nimba. Contrairement aux modèles épidémiologiques, ils utilisent un système discret se basant sur l'état de la population à un temps t pour déterminer l'état probable de la population à un temps $t+1$. De plus, ils prennent en compte le taux de correctif (patch). Pour modéliser un comportement de scan topologique, ils divisent l'espace des machines en sous-espaces, de tailles différentes et ayant chacun une probabilité d'être choisies. Cela leur permet d'étendre facilement leur modèle. Cependant, les auteurs combinent l'ensemble des fonctionnalités de scan et d'infection en un taux de scan, représentant le nombre de machines que peut scanner un zombie par unité de temps.

On voit ici que ces modèles sont très simplifiés, ils fusionnent l'ensemble des comportements de scan et d'infection en un seul paramètre. De plus, ils ne sont pas (ou peu) extensibles ; l'ajout d'un nouveau comportement oblige à modifier profondément le modèle. Enfin, ces outils ne sont utilisés que pour décrire la propagation d'un seul ver. Si l'on souhaite modéliser le phénomène de concurrence, il faudrait redéfinir les équations utilisées, car chaque réseau peut influencer les autres de manière significative.

4.1.4 ZOMBOTS : LE JEU DE SIMULATION DES PANDÉMIES DE ROBOTS ZOMBIES

Ainsi afin d'observer les impacts de chaque fonctionnalité sur l'efficacité d'un réseau de zombies, nous avons souhaité créer notre propre outil. Nous avions besoin d'un outil facilement extensible et très proche du fonctionnement réel des vers à étudier. Ainsi, plutôt que de définir des équations différentielles ou des suites arithmétiques, nous avons développé un programme de simulation sous forme d'un jeu tour par tour.

Pour fonctionner, il faut définir la machine d'état représentant le cycle de vie d'un zombie. Chaque état possède un nombre de tours qui lui est propre. Cela permet de représenter le temps que met le zombie pour accomplir une tâche. Ce temps peut être fixe ou déterminé à l'aide d'une fonction mathématique si ce dernier est variable. Cela peut être utile pour simuler la latence d'un réseau ou sa congestion progressive. Certains états sont obligatoires, comme par exemple, le scan et l'exploitation. L'état de scan permet de générer l'adresse IP à scanner en fonction de la stratégie employée. Une fois l'adresse générée, l'environnement de jeu va déterminer s'il correspond à une adresse vulnérable ou non. Il va ensuite transmettre le résultat au zombie qui pourra passer en état d'exploitation ou retourner en phase de scan. Cet automate d'état est représenté en figure 4.1.

Pour initialiser la simulation, on crée un nombre aléatoire de victimes (en utilisant une proportion

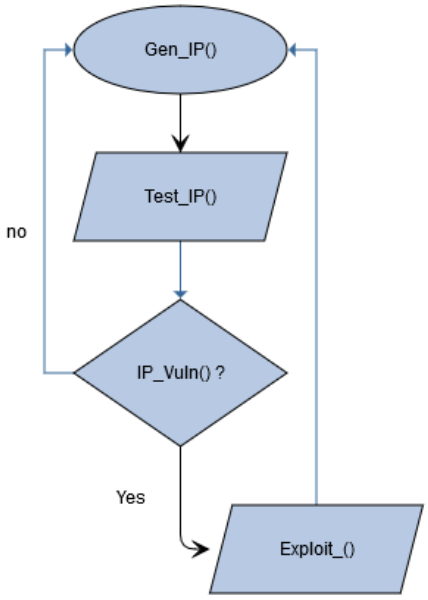

Figure 4.1 – Fonctionnement général d'un zombie lors du processus d'infection

donnée) vulnérable à chaque type de réseau de zombie. Ensuite, on va faire jouer chaque zombie dans un ordre aléatoire défini au début de chaque tour. Pour cela, on les fait avancer à l'état suivant. Si le zombie a généré une adresse IP, l'environnement va vérifier si cette dernière est vulnérable ou non. Si tel est le cas, on ajoute un zombie aux réseaux qui l'a infecté et les deux zombies passent en phase d'exploitation. L'état de la victime peut aussi être modifié si le programme malveillant corrige les failles qu'il a utilisées afin d'éviter que d'autres réseaux ne puissent acquérir la victime. Enfin, si un ver supprime les autres vers ayant déjà infecté la victime, on supprime un zombie dans chacun de ces réseaux. L'ensemble de cette phase de jeu est répété autant de fois que nécessaire. Un schéma récapitulatif est donné en figure 4.2. Sur cette figure, l'étiquette *GenIP()* correspond à la phase de génération de l'adresse IP qui sera attaquée, l'étiquette *TestIP()* correspond à la phase de détection de vulnérabilités exploitables chez la victime. Si une vulnérabilité est exploitable, alors on rentre dans la phase d'exploitation de la victime, étiquetée *Exploit()*.

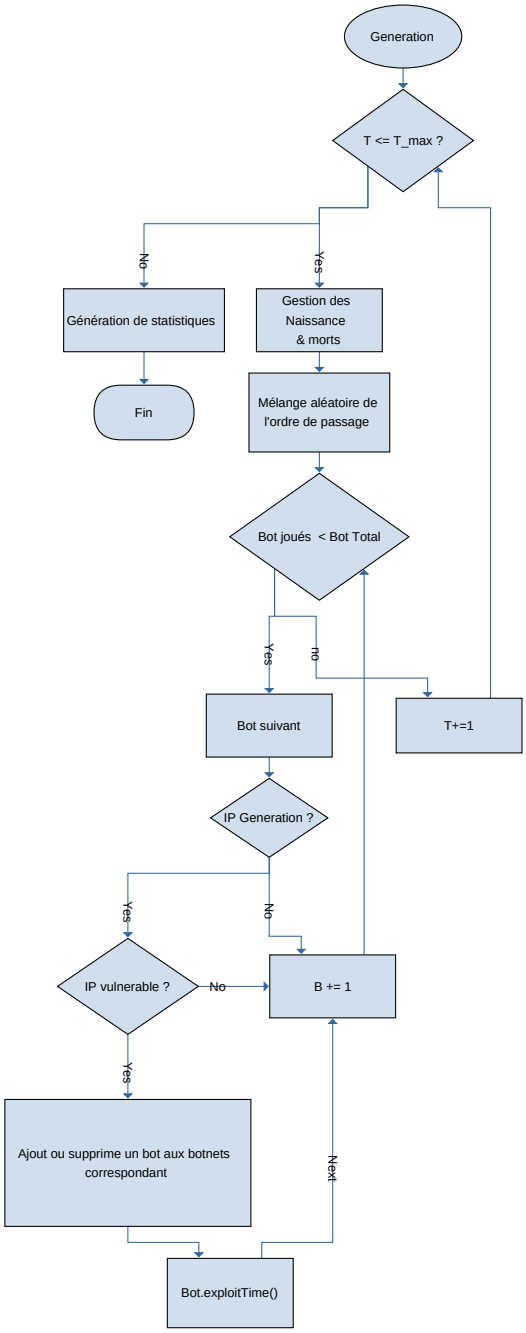

Figure 4.2 – Fonctionnement général du jeu

Ainsi, notre outil permet de très facilement observer les impacts que peuvent avoir chaque fonctionnalité sur l'efficacité du réseau de zombies. De plus, nous pouvons aisément simuler

une concurrence entre les réseaux de zombies par exemple entre Wifatch et ses prédécesseurs. La mort et la naissance de nouveaux appareils peuvent aussi être ajoutées en début ou en fin de tour. Pour ce faire, il suffit de déterminer un nombre aléatoire de machines à supprimer (et de les supprimer de leur réseau de zombies si elles ont été infectées) et à en générer de nouvelles. Là aussi, le nombre peut être fixe ou dépendre d'une fonction plus précise.

Le nombre d'états d'un zombie ainsi que les méthodes pour déterminer les temps pris par chaque phase ne sont pas limités. Ainsi, notre modèle ne fusionne pas divers comportement ou fonctionnalité sous un seuls paramètre, car on peut facilement modéliser chaque fonctionnalité et observer leur impact. Il est très extensible de par sa conception. Enfin, notre modèle simule la concurrence entre plusieurs réseaux de zombies (similaires ou différents). L'ensemble de cette partie est récapitulé dans le tableau 4.1. Pour rappel le critère de simplicité représente le fait que le modèle fusionne plusieurs comportements en un seul paramètre, l'extensibilité correspond à la capacité du modèle à pouvoir modéliser de nouvelles fonctionnalités, et la concurrence représente la capacité du modèle à décire la concurrence entre réseaux de zombies.

Modèle	Simplicité	Extensibilité	Concurrence	Type
S.I	✓	✗	✗	Déterministe
S.I.S	✓	✗	✗	Déterministe
Modèle Markovien	✓	✓	✗	Stochastique
AAWAP	✓	✓	✗	Stochastique
Notre modèle	✗	✓	✓	Stochastique

Tableau 4.1 – Comparaison des différents modèles

4.2 LES SIMULATIONS EFFECTUÉES AVEC NOTRE MODÈLE.

Dans cette section nous allons détailler les paramètres utilisés pour faire nos expériences. Nous allons ensuite exposer et analyser nos résultats expérimentaux.

Afin d'observer les différents impacts de quelques fonctionnalités, nous avons mis en place quelques simulations. Notre but est ici d'observer l'impacte de la stratégie de recherche de victimes (famille numéro 7 de notre taxonomie) sur la vitesse de propagation d'un réseau de zombies. Ensuite nous avons souhaité étudier l'impact des fonctionnalités de prévention des autres réseaux de zombies (famille numéro 11) sur l'efficacité d'un réseaux de zombies.

Pour chaque expérience, nous avons fait 105 simulations et avons tracé pour chaque réseau de zombies, sa population maximale, minimale, moyenne et médiane des simulations. Nous avons ensuite mis sur un même graphique les populations médianes de chaque réseau afin de les comparer plus facilement.

Les taux de mort et de naissance suivent une loi normale de moyenne 5 et d'écart-type 2. Le taux de mort a une fréquence d'apparition de 200 tours, tandis que le taux de naissance possède une fréquence d'apparition de 150 tours. Cela permet de modéliser simplement l'ajout et le retrait de nouveaux équipements connectés. Comme le nombre d'objets connectés est en hausse depuis plusieurs années, nous avons choisi de mettre en place une fréquence d'apparition plus élevée pour le taux de naissance que pour le taux de mort. Cependant, les valeurs sont ici arbitraires, car nous souhaitons observer les impacts des fonctionnalités et non modéliser les comportements réels d'un réseau de zombies particulier. Ces valeurs et ces fonctions peuvent être modifiées aisément si l'on souhaite reproduire la propagation d'une infection particulière.

Ici le taux de naissance et le taux de mort sont les mêmes pour toutes les expériences, afin qu'ils n'influencent pas nos expériences. Cette fonctionnalité sera utile pour de futures extensions

de ces travaux, où l'on pourrait souhaiter, par exemple, modéliser de manière très fidèle la propagation d'une infection sur un temps cours, tout en prenant en compte l'apparition et la disparition d'objets connectés du réseau Internet.

De même la notion de tour est une abstraction du temps, ainsi un tour représente une quantité atomique de temps. La correspondance en secondes, minutes etc, dépend ainsi de l'expérience et de ce que l'on souhaite observer. Pour nos expériences, le temps est abstrait complètement, nous souhaitons observer les conséquences des écarts de temps consommé par chaque fonctionnalité. Qu'une fonctionnalité A prenne 40ms de moins que la fonctionnalité B pour effectuer la même tache ne nous importe pas ici. Ce que nous souhaitons observer sont les conséquences d'un écart de temps, et comment cet écart va impacter l'efficacité générale du réseau de zombies. Peu importe que l'écart soit de 40ms ou 2min.

La première expérience a pour but d'observer la concurrence entre plusieurs réplicats d'un même programme malveillant. Cela se produit lorsque le code source d'un programme permettant de créer un réseau de zombies est vendu à plusieurs entités ou mis à disposition en ligne (comme pour Mirai par exemple). On a ainsi une compétition entre chaque réseau de zombies afin de capturer un maximum d'hôtes. Pour ce faire, nous avons simulé un programme ayant un scan aléatoire des victimes et s'immunisant à l'ensemble de ses réplicats. Cela veut dire que lorsqu'il infecte une victime, le programme va corriger la faille lui ayant permis d'entrer. Un programme comme Mirai fait cela en fermant les ports Telnet et SSH.

Notre simulation se fait sur 1500 tours, avec une population totale de 100000 individus dont 30% sont vulnérables aux vers. Nous avons fait deux expériences avec deux et cinq réplicats. Nous avons ensuite fait une simulation où chaque réseau commence à un temps différent. L'ensemble des paramètres de chaque réseau est donné dans le tableau 4.2. Nous avons choisi ces paramètres de manière arbitraire, concernant la durée de la simulation (1500 tour) de manière a pouvoir observer la totalité de l'évolution du système et donc observer le moment où le système devient

stable. Nous avons remarqué que dans la majorité des expériences, au bout de 800 tours, le système était stable. Ainsi, 1500 tours nous a paru un bon compromis, permettant de nous assurer de la stabilité du systèmes. Cependant, nous avons quand même fait varier ces paramètres pour certaines expériences (sans varier les autres) afin de montrer la pertinence de ces choix.

Notre deuxième expérience compare les différentes stratégies de scan. Nous avons simulé deux réseaux de zombie ayant des caractéristiques identiques, sauf pour la méthode de génération d'adresses à tester. Le premier les génère de manière aléatoire, tandis que le second les génère de manière séquentielle. Cela correspond aux fonctionnalités 7.2 et 7.3 de notre taxonomie. Il commence donc par la 1^{re} adresse, puis la 2^{me}, etc. De plus, chaque zombie reprend le scan depuis le début. Ce comportement peut être observé sur le tout premier programme malveillant pour objet connecté Hydra. En effet, la méthode de scan n'était pas encore automatisée et chaque zombie scannait l'ensemble des adresses IP en partant du début.

Cette dernière méthode étant particulièrement inefficace, nous avons fait une simulation sur 1000 tours, puis une sur 5000 tours afin d'observer la totalité de la compétition entre les deux vers. Au départ nous souhaitions effectuer une simulation sur 100000 tours pour laisser le premier réseau scanner l'ensemble des adresses existantes. Cependant, au bout de 750 tours, nous avions remarqué que le système était stable et que les populations des réseaux n'évoluaient que très peu. Ici, les victimes étaient vulnérables aux deux réseaux. Il y avait 30% des individus de vulnérables aux deux réseaux. Nous avons ensuite refait l'expérience, en attribuant des temps d'action relativement plus faibles pour le réseau de zombies utilisant la méthode de scan séquentielle que pour le réseau utilisant le scan aléatoire. L'ensemble des paramètres de chaque réseau est donné dans les tableaux 4.3 et 4.4.

Enfin, nous avons voulu observer le phénomène de suppression des concurrents. Cela s'est vu avec Wifatch, qui supprimait l'ensemble des programmes malveillants qu'il trouvait chez ses victimes et les immunisait ensuite. Cela correspond aux fonctionnalité 11.1 et 11.2 de notre

taxonomie. Ici, nous avons fait une simulation avec des réseaux identiques, à la seule différence que le premier n'immunisait pas ses victimes, là où le second supprimait les infections du premier et immunisait ses victimes. Nous avons ensuite fait de même, mais en modifiant la stratégie de scan du premier réseau afin qu'il utilise une génération d'adresse séquentielle. Dans tous les cas, nous avons fait une simulation avec les deux réseaux actifs en même temps et une simulation avec le second réseau (celui supprimant les infections de l'autre) démarrant avec un retard de 200 tours de jeu. Ici, les simulations se faisaient sur 1500 tours avec 30% des individus vulnérables. L'ensemble des paramètres de chaque réseau est donné dans les tableaux 4.5 et 4.6. Dans ces tableaux, A signifie *Aléatoire* et correspond à la fonctionnalité 11.2 de notre taxonomie ; S signifie *Séquentiel* et correspond à la fonctionnalité 11.1.

Réseau de zombies	Méthode de scan	Temps de génération IP	Temps de test	Temps d'exploit	Suppression	Immunité	Départ (t)
botnet #1	A	3	5	4	∅	Tous	0
botnet #2	A	3	5	4	∅	Tous	100
botnet #3	A	3	5	4	∅	Tous	150
botnet #4	A	3	5	4	∅	Tous	300
botnet #5	A	3	5	4	∅	Tous	1000

Tableau 4.2 – Expérience 1 : concurrence entre réseau du même type

Réseau de zombies	Méthode de scan	Temps de génération IP	Temps de test	Temps d'exploit	Suppression	Immunité	Départ (t)
botnet #1	S	1	5	4	∅	#1 #2	0
botnet #2	A	3	5	4	∅	#1 #2	0

Tableau 4.3 – Expérience 2a : concurrence entre stratégie de scan aléatoire et séquentielle

Réseau de zombies	Méthode de scan	Temps de génération IP	Temps de test	Temps d'exploit	Suppression	Immunité	Départ (t)
botnet #1	S	1	1	1	∅	#1 #2	0
botnet #2	A	3	5	4	∅	#1 #2	0

Tableau 4.4 – Expérience 2b : concurrence entre stratégie de scan aléatoire et séquentielle

Réseau de zombies	Méthode de scan	Temps de génération IP	Temps de test	Temps d'exploit	Suppression	Immunité	Départ (t)
botnet #1	A	3	5	4	∅	∅	0
botnet #2	S	3	5	4	#1	#1 #2	200

Tableau 4.5 – Expérience 3a : concurrence, stratégie identique avec surpression

Réseau de zombies	Méthode de scan	Temps de génération IP	Temps de test	Temps d'exploit	Suppression	Immunité	Départ (t)
botnet #1	S	1	1	1	∅	∅	0
botnet #2	A	3	5	4	#1	#1 #2	200

Tableau 4.6 – Expérience 3b : concurrence, stratégie différente avec surpression

4.3 LES RÉSULTATS DES SIMULATIONS

Dans cette section nous allons présenter les résultats de nos expériences. Pour certaines d'entre elles, nous n'avons pas affiché tous les graphiques. Cependant nous avons mis à disposition un dépôt Github [1] contenant notre code source ainsi que la totalité des graphiques. Ici, nous avons affiché l'évolution de la population pour quelques-uns des réseaux de zombies. Pour chacune, nous affichons sa courbe d'infection minimale, maximale, moyenne et médiane de l'ensemble des simulations de l'expérience.

4.3.1 EXPÉRIENCE 1A

Cette première expérience a pour but d'observer la concurrence entre divers réplicats d'un même réseau de zombies. Ici, nous considérons un cas où tous les réplicats commencent à chercher des victimes en même temps. Notre objectif est de trouver le temps à partir duquel le système devient stable et que les populations n'évoluent plus de manière significative. Ce cas de figure représente la phase primaire d'infection, nous ne montrons pas la phase de vie, où les divers réseaux vont grappiller les nouveaux hôtes ou ceux ayant subi un redémarrage provoquant la disparition de l'infection.

La Figure 4.3 représente l'évolution du premier ver (appelé ici Mirai0, car adoptant la même stratégie de scan) dans la configuration où cinq réplicats se disputent les victimes. La Figure **??** représente l'évolution du troisième réplicat (appelé ici Mirai2) dans la configuration où cinq réplicats se disputent les victimes.

1. `https://github.com/bvignau/The_Botnet_Game`

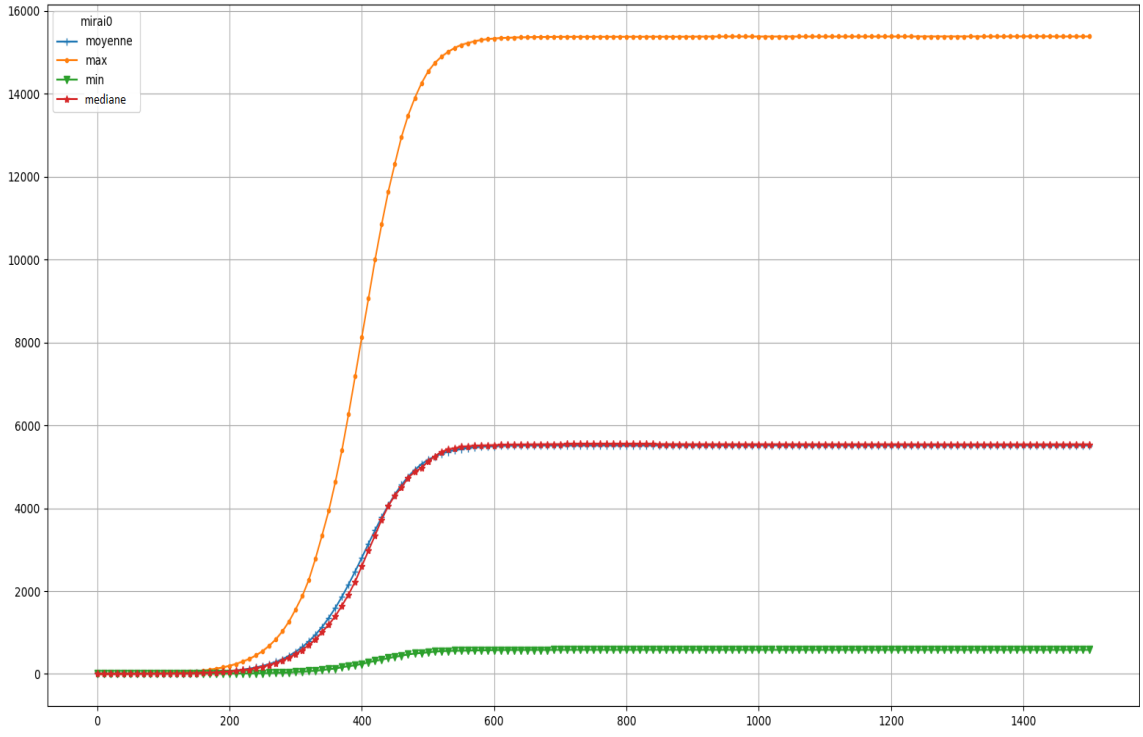

Figure 4.3 – Évolution de la population du réseau #1 sur 1 500 tours (configuration avec 5 vers)

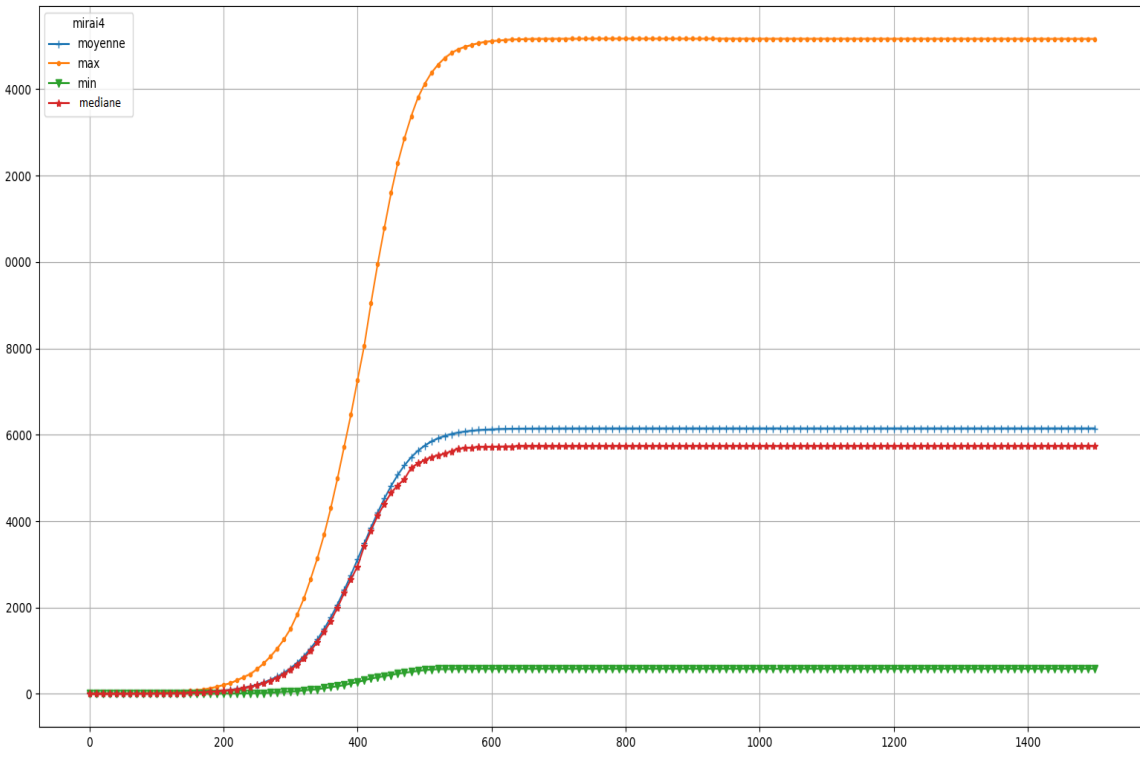

Figure 4.4 – Évolution de la population du botnet #5 sur 1500 tours (configuration avec 5 vers)

La Figure 4.4 représente l'évolution du cinquième réplicat (appelé ici Mirai4) dans la configuration où cinq réplicats se disputent les victimes. Ici, nous ne montrons qu'un ver sur deux car les résultats sont très similaires. Cependant, l'ensemble des figures des populations de chaque réplicat est disponible sur Github.

Pour cette expérience, on observe une grande disparité entre les populations de zombies. Les différences entre maximaux et minimaux sont importantes. On peut observer sur les graphiques représentant les populations individuelles (Figure 4.3 **??**), que leur minimum est d'environ 750 individus et leur maximum à plus de 15 000. Cette grande variation s'explique par le côté aléatoire de la stratégie de scan. On observe donc qu'une technique de scan aléatoire possède une efficacité très variable en milieu compétitif.

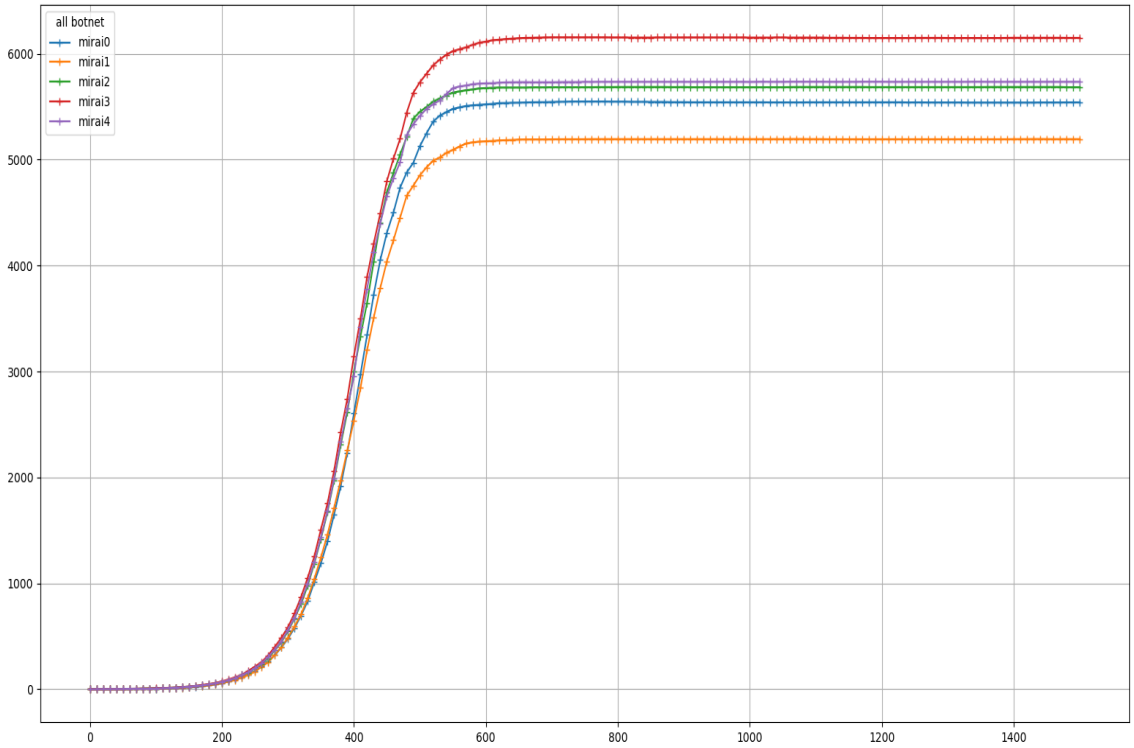

Figure 4.5 – Évolution de la population des 5 botnets sur 1500 tours (configuration avec 5 vers)

Sur la figure 4.5 nous affichons les courbes d'infections médianes de l'ensemble des réseaux de

zombies. Nous pouvons observer que ces dernières sont plutôt homogènes. L'écart de population entre la plus faible et la plus forte est aux environs de 1 000 individus. De plus, comme tous les réseaux vont attaquer la même population, on observe une division des populations de chaque réseau. Chaque réseau possède entre 16% et 21% du total victimes.

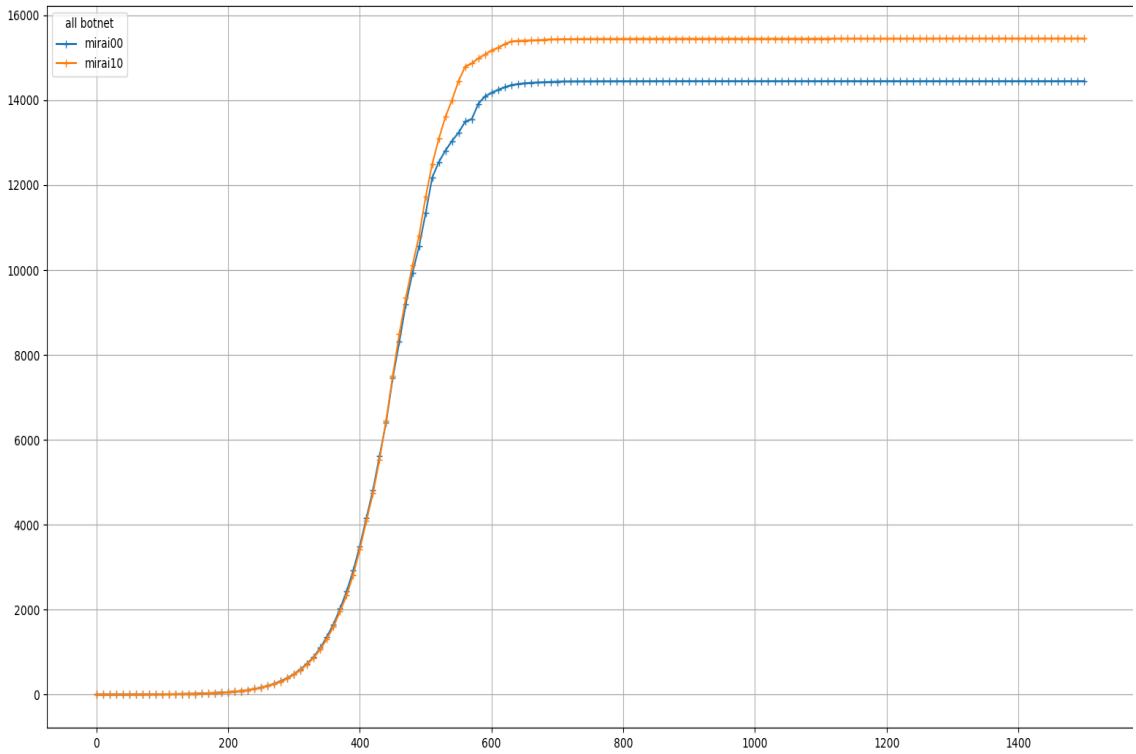

Figure 4.6 – Évolution de la population des 2 botnets sur 1500 tours (configuration avec 2 vers)

Sur la figure 4.6 nous observons les populations médianes des réseaux de zombies dans une configuration à deux réseaux. Ici, nous observons que les deux populations médianes sont proches et se retrouvent aux alentours des 15 000 individus. Cela correspond à environ 50% de la totalité de la population vulnérable.

On observe donc que plus il y a de réseaux, plus la puissance de chaque réseau sera diminuée, de manière quasi proportionnelle aux nombres de compétiteurs. Ainsi, une première solution pour réduire la puissance d'un réseau de zombies serait de lancer plusieurs réseaux du même type

dans un temps proche. Cependant, cette solution requiert une analyse et une modification très rapide d'un programme malveillant. En pratique, il y aura forcément un délai entre le moment où le programme originel commencera son attaque et le moment où une équipe pourra déployer un concurrent à ce réseau.

4.3.2 EXPÉRIENCE 1B

Cette seconde version de la première expérience vise à évaluer l'impact d'un retard au départ dans la capacité d'infection d'un réseau de zombies. Ainsi, nous avons lancé les mêmes réseaux de zombies, mais avec des temps de départs différents. Cela peux représenter, par exemple, le fait qu'une personne créé un réseaux de zombie avec un code, et que plus tard, une autre personne utilise le même code pour créer une instance différente et concurrente du réseaux de zombie initial.

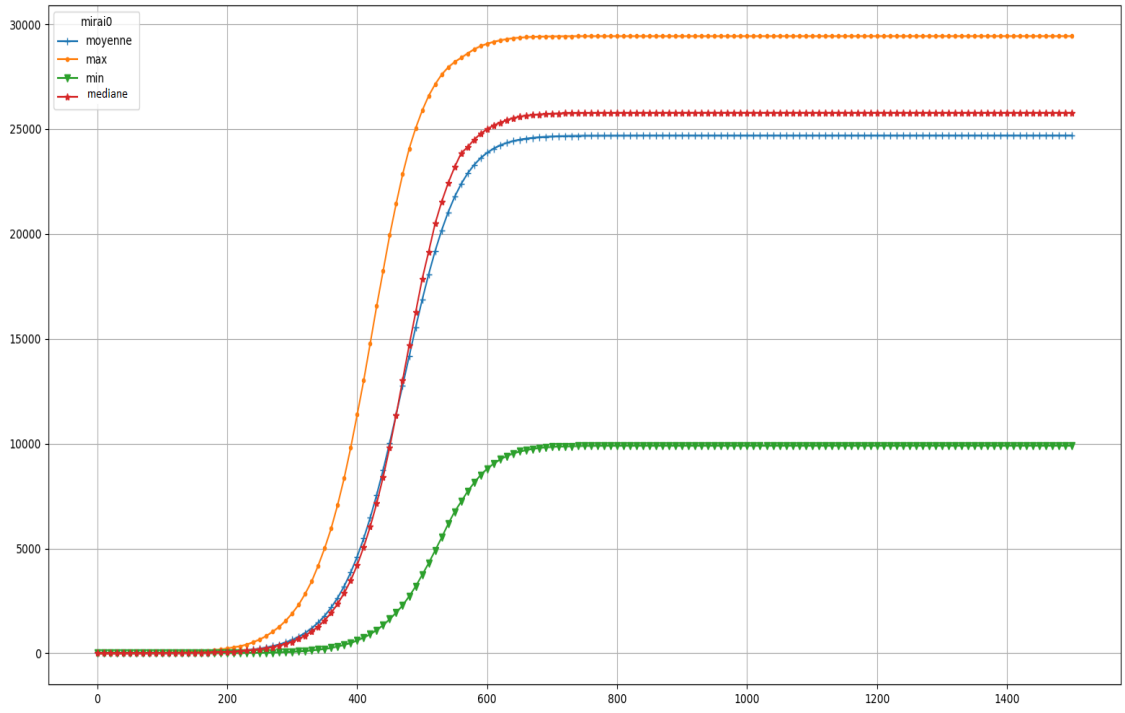

Figure 4.7 – Évolution de la population du botnet #1 sur 1500 tours

Sur la figure 4.7, nous pouvons observer l'évolution de la population du premier réseau de zombie. On constate toujours un grand écart entre le maximum et le minimum mais ce dernier est bien moins grand que pour l'expérience précédente. Ici, le minimum est aux alentours de 10000 individus et le maximum vers 30000, soit la totalité de la population vulnérable. Cependant, on observe que la médiane se situe un peut au dessus des 25000 individus, montrant ainsi une efficacité très importante dans une moitié des cas. Comme pour la première expérience, nous ne montrons que les résultats d'un réseau sur deux. L'ensemble des résultats est sur notre Github.

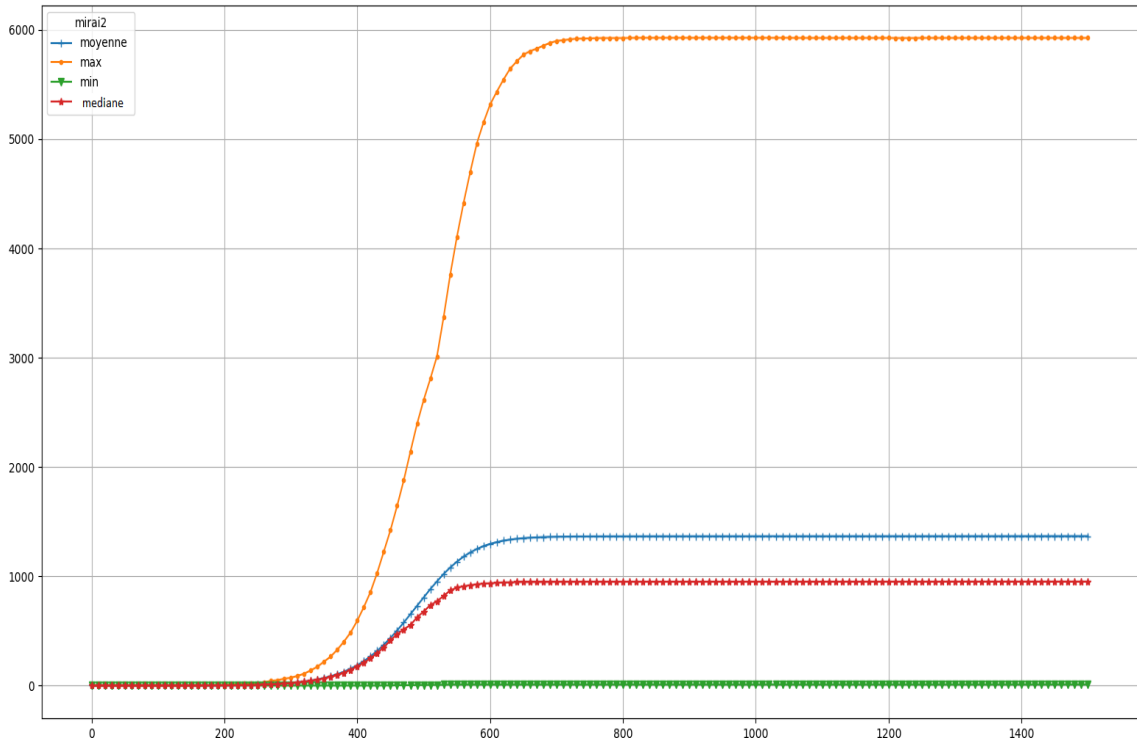

Figure 4.8 – Évolution de la population du botnet #3 sur 1500 tours

Sur la figure 4.8 nous observons l'évolution du troisième réseau de zombies, parti avec 150 tours de retard. Ici aussi, l'écart entre le maximum et le minimum est très élevé, environ 6 000 individus au maximum et presque 200 au minimum. Encore une fois, cela montre la grande variabilité de l'efficacité de cette technique. De plus, contrairement au réseau précédent, la population médiane et la population moyenne sont faibles, aux alentours de 1 000 individus.

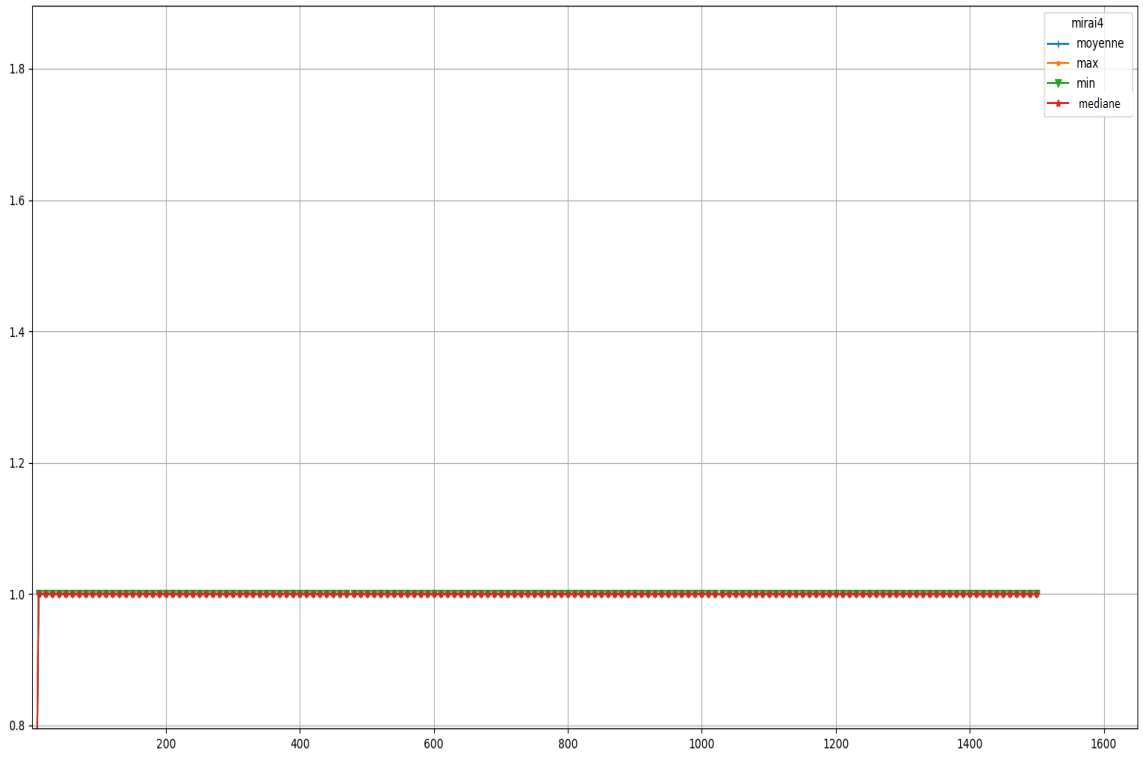

Figure 4.9 – Évolution de la population du botnet #5 sur 1500 tours

Pour le dernier réseau de zombies, démarré avec 1000 tours de retard, il n'a pas été capable d'infecter une seule victime. En effet, on observe sur l'expérience précédente que le point d'équilibre est atteint entre 600 et 800 tours. Or ici, en partant après ce point d'équilibre, le réseau ne peut pas infecter de nouvelles victimes.

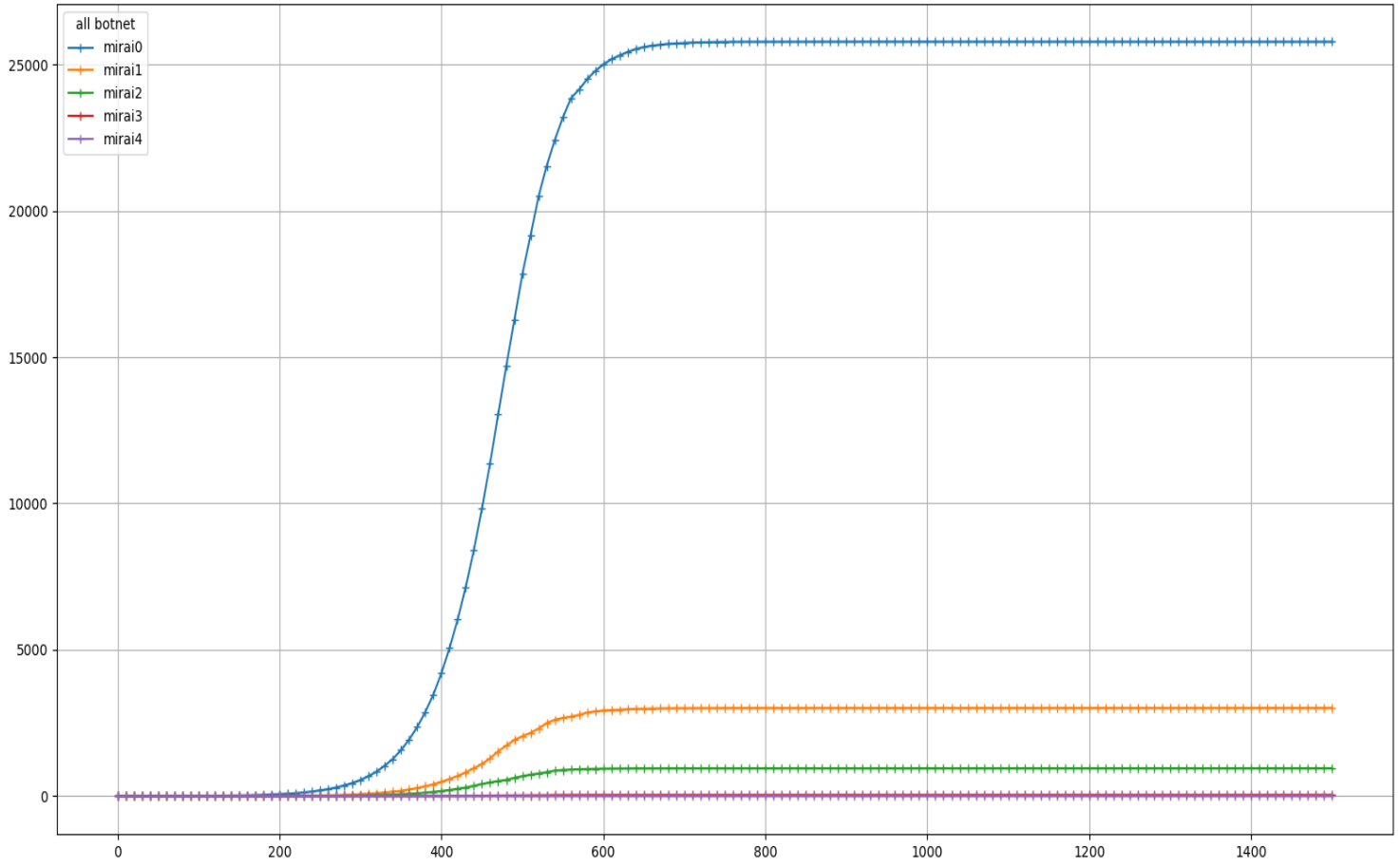

Figure 4.10 – Évolution de la population des 5 botnets sur 1500 tours

Finalement, en observant les populations médianes de chaque réseau, on observe que l'efficacité de chaque réseau décroît avec le temps de retard. Ceci est parfaitement logique étant donné que chaque réseau va infecter des victimes que les autres réseaux ne pourront récupérer. De plus, cette stratégie d'infection possède un point à partir duquel l'efficacité est exponentielle. Ainsi, avoir un départ retardé influence énormément le nombre de zombies que peut recruter un réseau. 100 tours de retards entraînent une grande différence entre les réseaux de zombies.

Le premier obtient plus de 25 000 individus représentant ici plus de 80% de la population vulnérable. Le second réseau n'en obtient qu'environ 4 000. Ainsi, on observe que par rapport à la première expérience où tous les réseaux démarrent simultanément, le premier réseau de zombies a fortement gagné en efficacité, là où les autres ont perdu significativement.

4.3.3 EXPÉRIENCE 2A

Pour cette expérience, nous avons souhaité observer la compétition entre deux modes d'infection. Le premier est le scan séquentiel et le second le scan aléatoire comme utilisés précédemment. Ici, nous supposons que le scan séquentiel sera moins performant que le scan aléatoire du fait que seul le premier zombie pourra ajouter de nouvelles recrues. Ici aussi, notre but est d'observer l'évolution des systèmes jusqu'à leur point d'équilibre. Nous avons fait les expériences sur 1000 et 5000 tours.

Les figures 4.11 et 4.12 représentent le réseau 1 (appelé ici Psybot), utilisant le scan séquentiel. La première figure représente l'évolution de la population sur 1000 tours et la seconde sur 5000tours. Les figures 4.13 et 4.14 représentent le réseau 2 (appelé ici Mirai), utilisant le scan aléatoire. La première figure représente l'évolution de la population sur 1 000 tours et la seconde sur 5 000 tours.

On peut d'ores et déjà observer que le point d'équilibre du système est atteint vers 800 tours et qu'à partir de là le système n'évolue plus. Dans cette expérience, on observe que le réseau 1 démontre une faible efficacité, variant entre une dizaine et une vingtaine d'individus. Cette variation s'explique par la distribution aléatoire des victimes, changeant d'une simulation à une autre. À l'opposé, le réseau deux réussis à accaparer la majorité de la population. Cependant, contrairement à la première expérience, on observe que la population totale termine toujours aux alentours de 30000 individus. La différence entre chaque simulation se fait au niveau du

temps d'apparition du point d'équilibre. Ainsi, dans le meilleur des cas, on observe aux figures 4.13 et 4.14 un point d'équilibre peut après le tour 600. Ce point arrive autour du tour 800 dans le pire des cas.

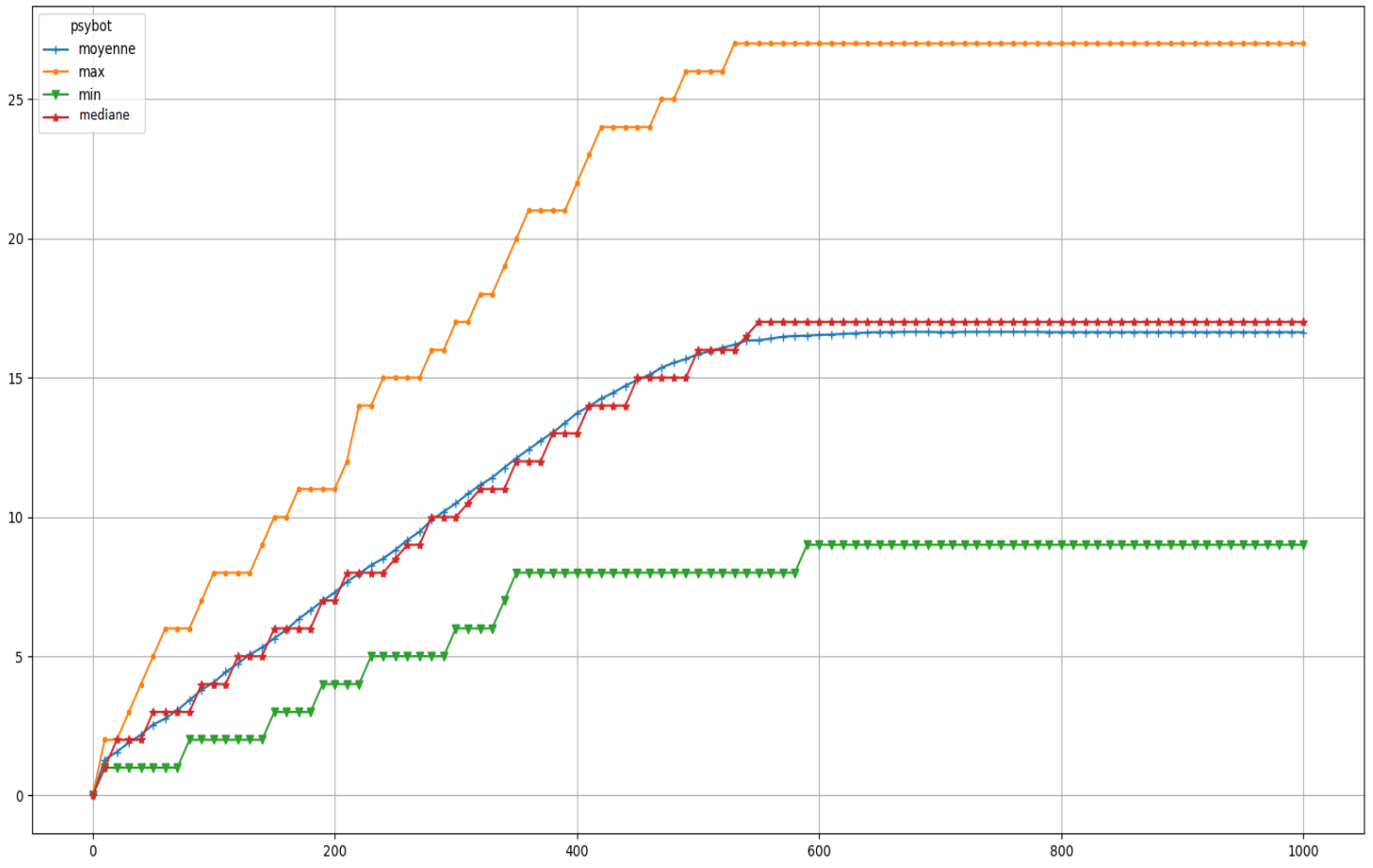

Figure 4.11 – Évolution de la population du botnet #1 sur 1000 tours

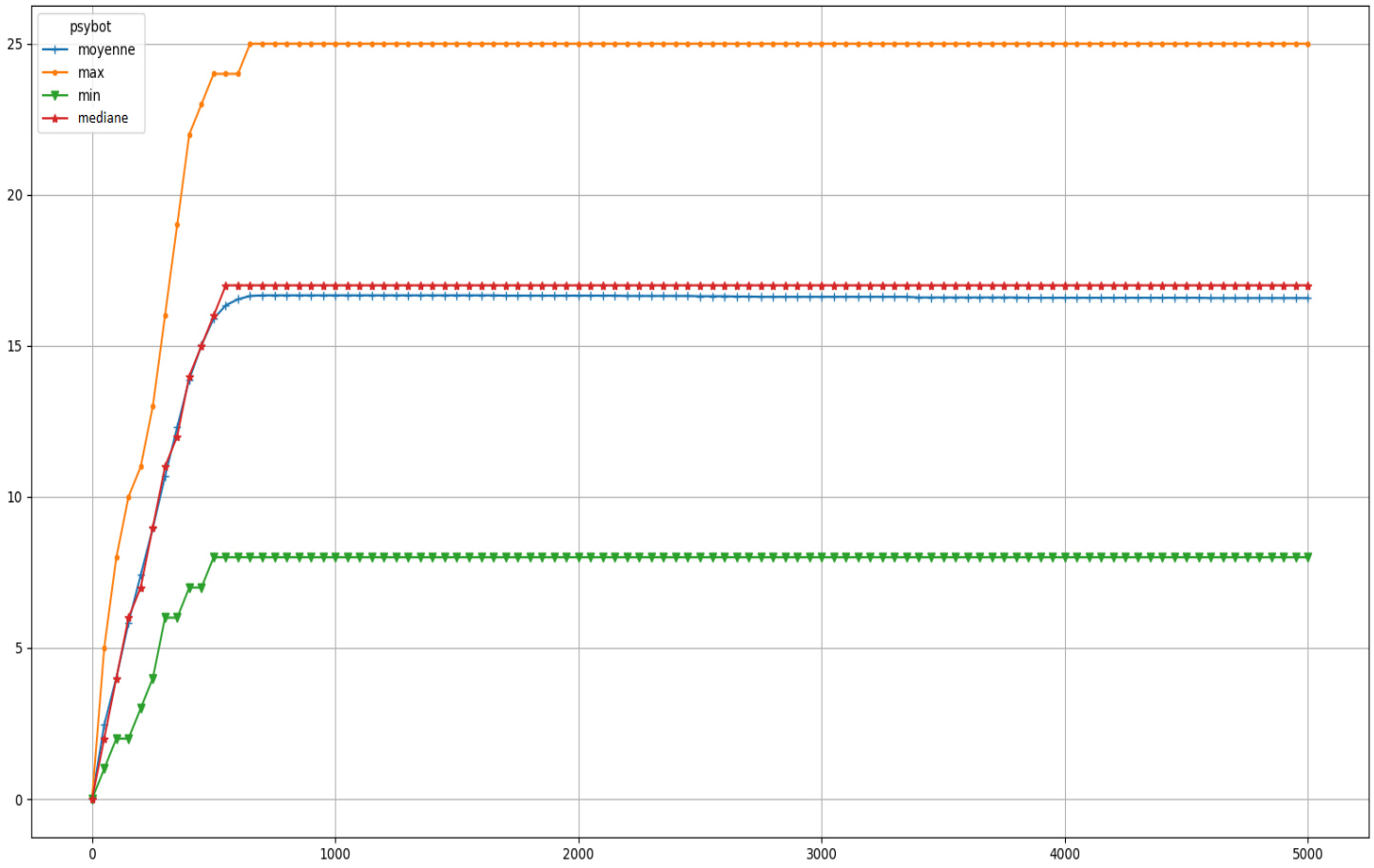

Figure 4.12 – Évolution de la population du botnet #1 sur 5 000 tours

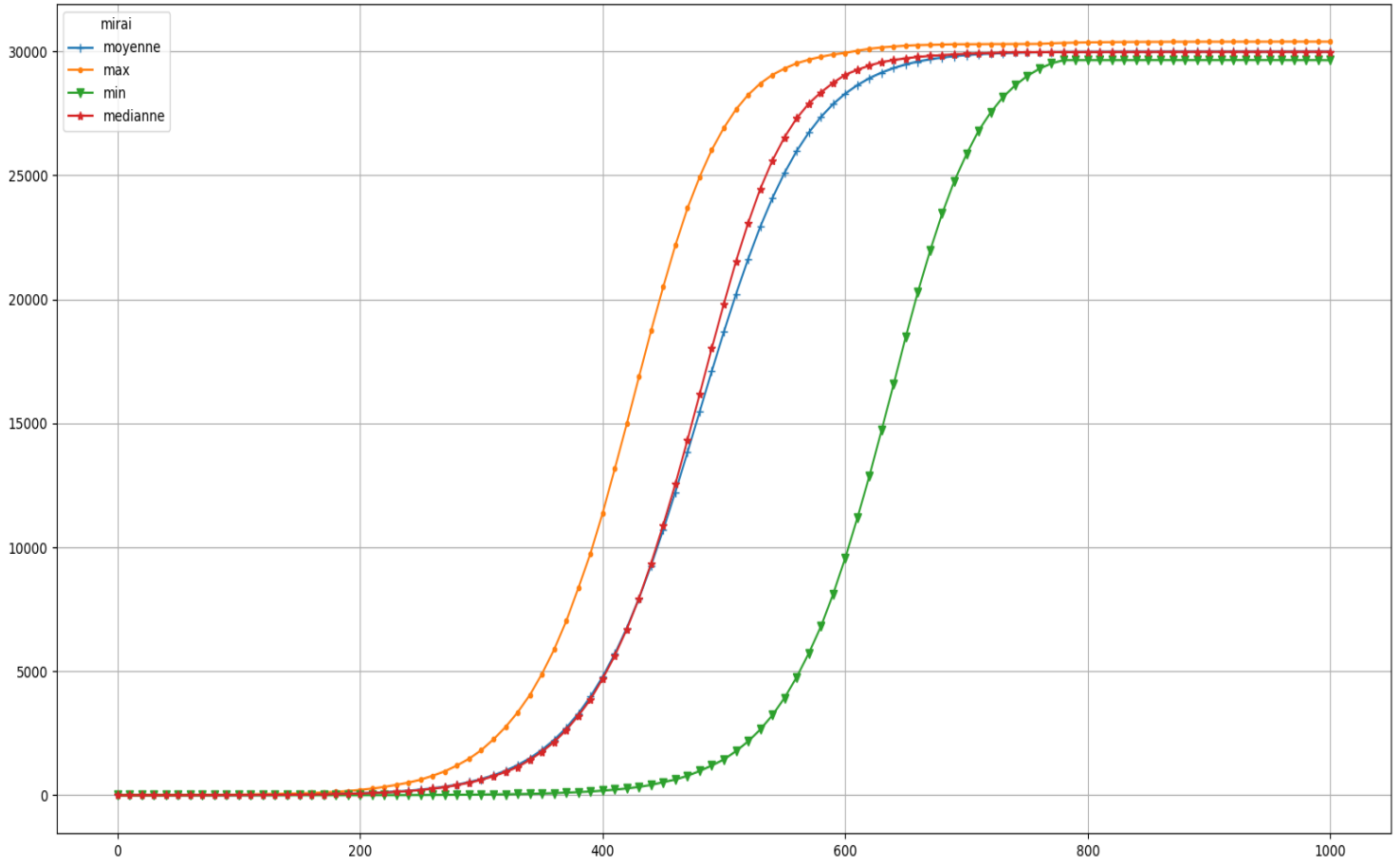

Figure 4.13 – Évolution de la population du botnet #2 sur 1 000 tours

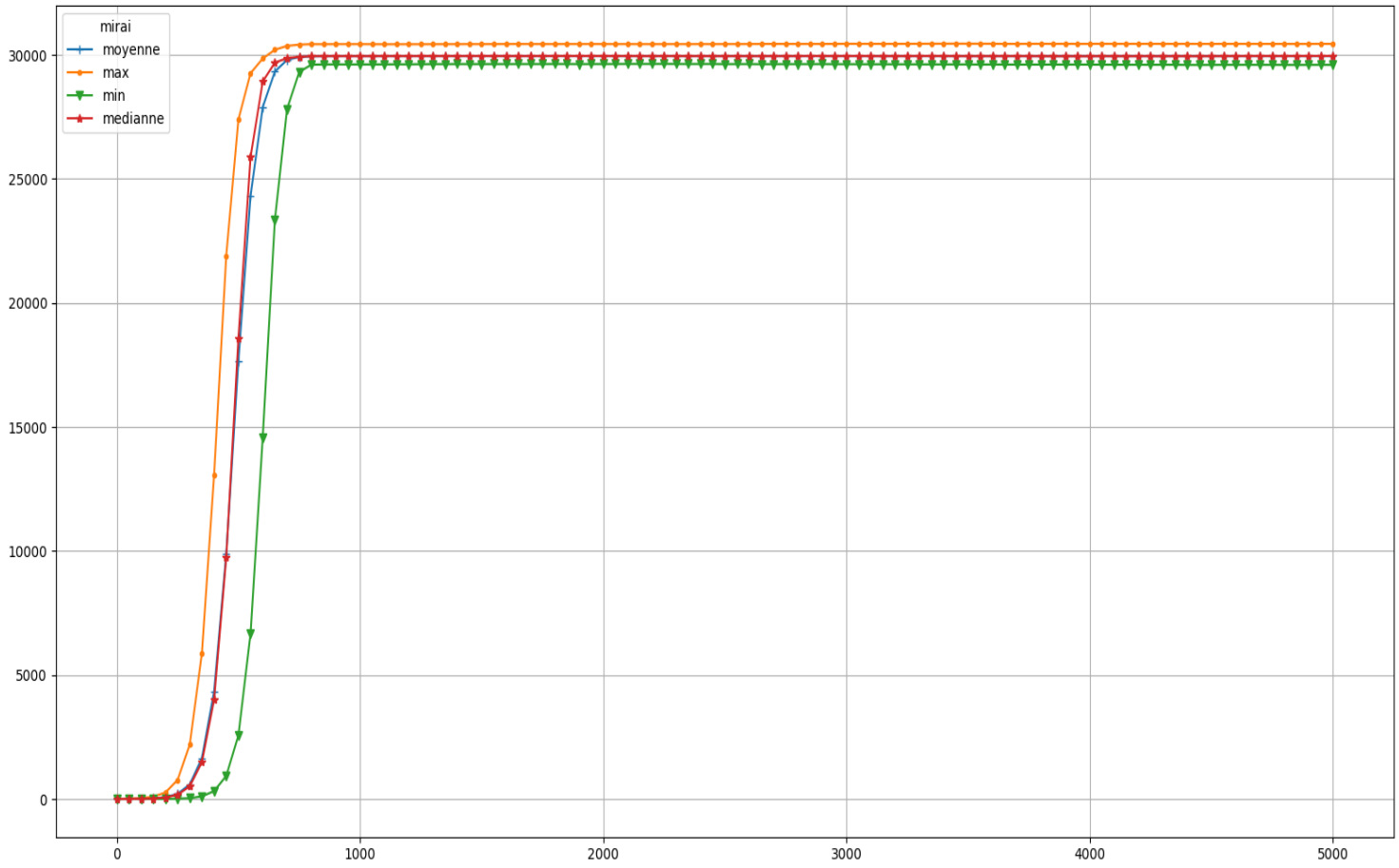

Figure 4.14 – Évolution de la population du botnet #2 sur 5 000 tours

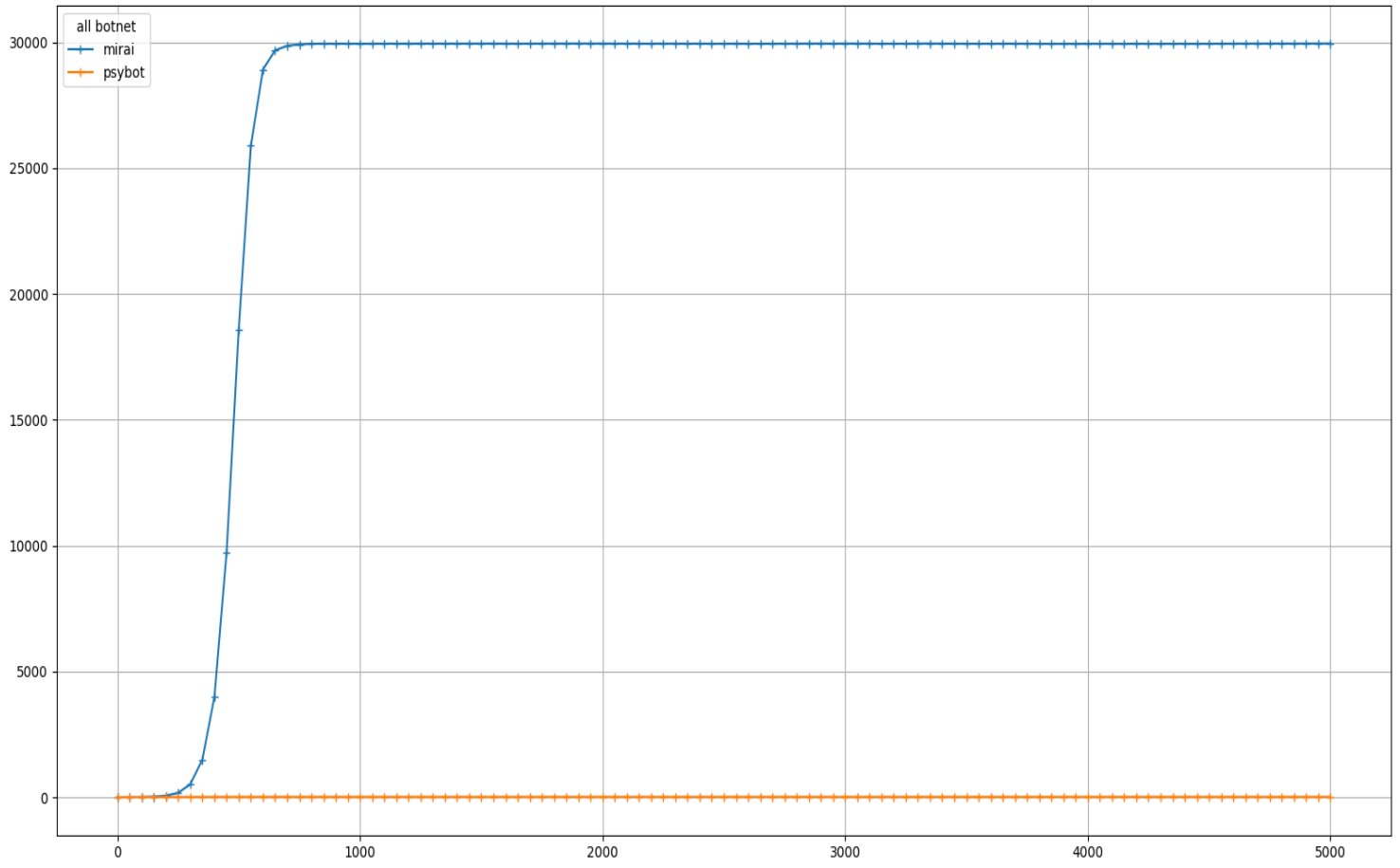

Figure 4.15 – Évolution de la population des deux botnet sur 5 000 tours

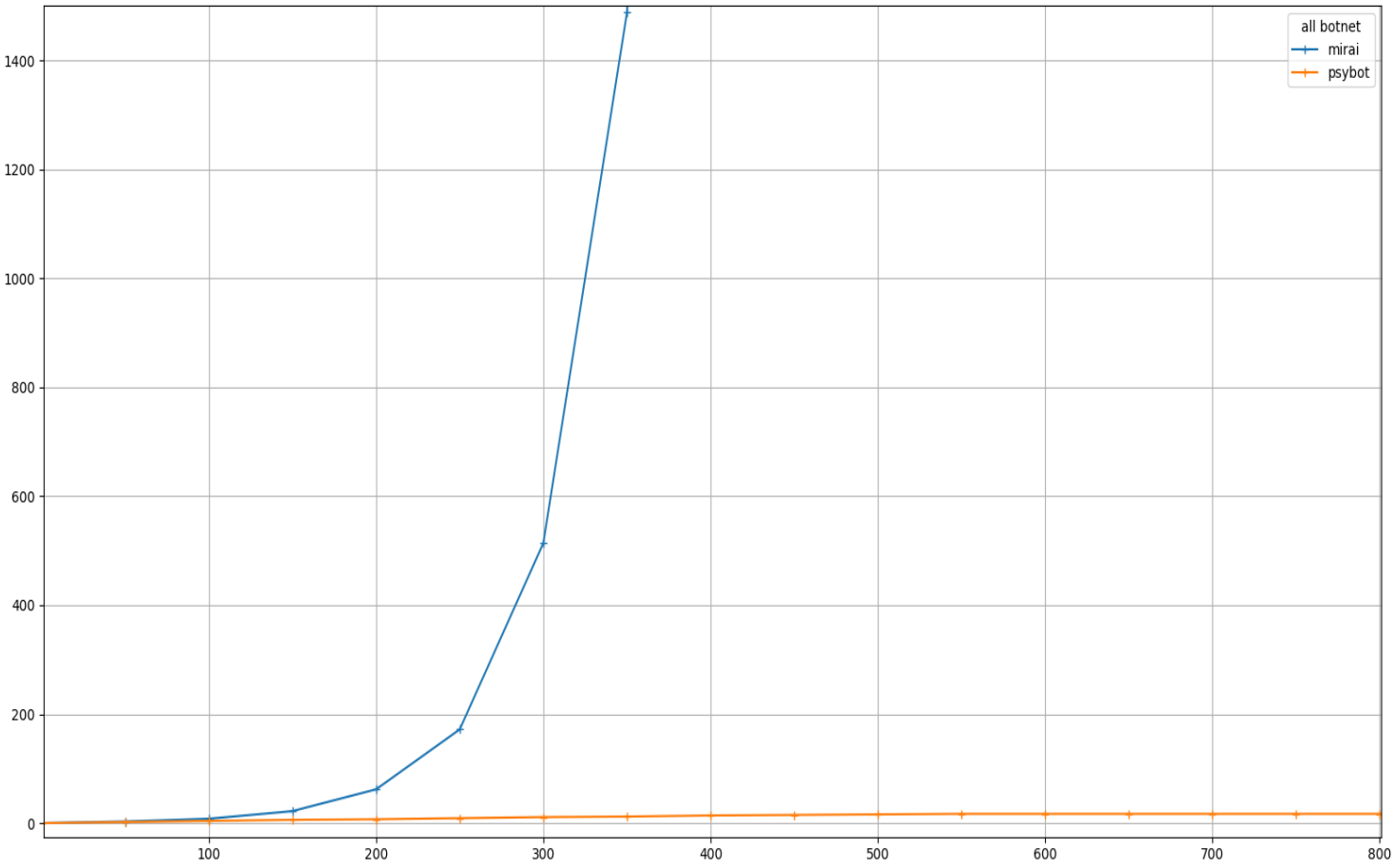

Figure 4.16 – Zoom de l'évolution de la population des deux botnet sur 5 000 tours

Les figures 4.15 et 4.16 représentent l'évolution de la population médiane des deux réseaux. On y observe que le réseau 2 est beaucoup plus efficace que le réseau 1 et le surpasse largement à partir de 150 tours. En effet, l'évolution du réseau utilisant le scan séquentiel est très lente et régulière, alors que le scan aléatoire possède une évolution exponentielle à partir d'un certain nombre d'appareils infectés. Ici, ce point arrive entre 150 et 200 tours.

4.3.4 EXPÉRIENCE 2B

Pour cette expérience, nous avons modifié les temps que prend chaque action pour le réseau 1, qui utilise le scan séquentiel. En effet, nous souhaitions observer l'impact du temps de processus sur cette stratégie et voir si cela permettait d'améliorer significativement son efficacité. Ainsi, nous avons divisé par 5 et par 4 les temps des phases de scans et d'infections. À noter que le temps de génération d'une adresse IP était 3 fois moins important pour le réseau 1 que pour le second.

Les figures 4.17 et 4.18 représentent le réseau 1 (appelé ici Psybot), utilisant le scan séquentiel. La première figure représente l'évolution de la population sur 1 000 tours et la seconde sur 5 000 tours. La figure 4.19 est un agrandissement de la figure 4.18. Les figures 4.20 et 4.21 représentent le réseau 2 (appelé ici Mirai), utilisant le scan aléatoire. La première figure représente l'évolution de la population sur 1 000 tours et la seconde sur 5 000 tours. La figure 4.22 est un agrandissement de la figure 4.21. Ici, les résultats sont proches de l'expérience 2A. L'efficacité du réseau 1 est meilleure que dans l'expérience précédente, mais reste extrêmement faible par rapport au réseau 2 utilisant le scan aléatoire. Son efficacité a plus que doublé, en passant en moyenne de 16 à 35 individus. Cela reste cependant négligeable par rapport aux 30 000 individus vulnérables.

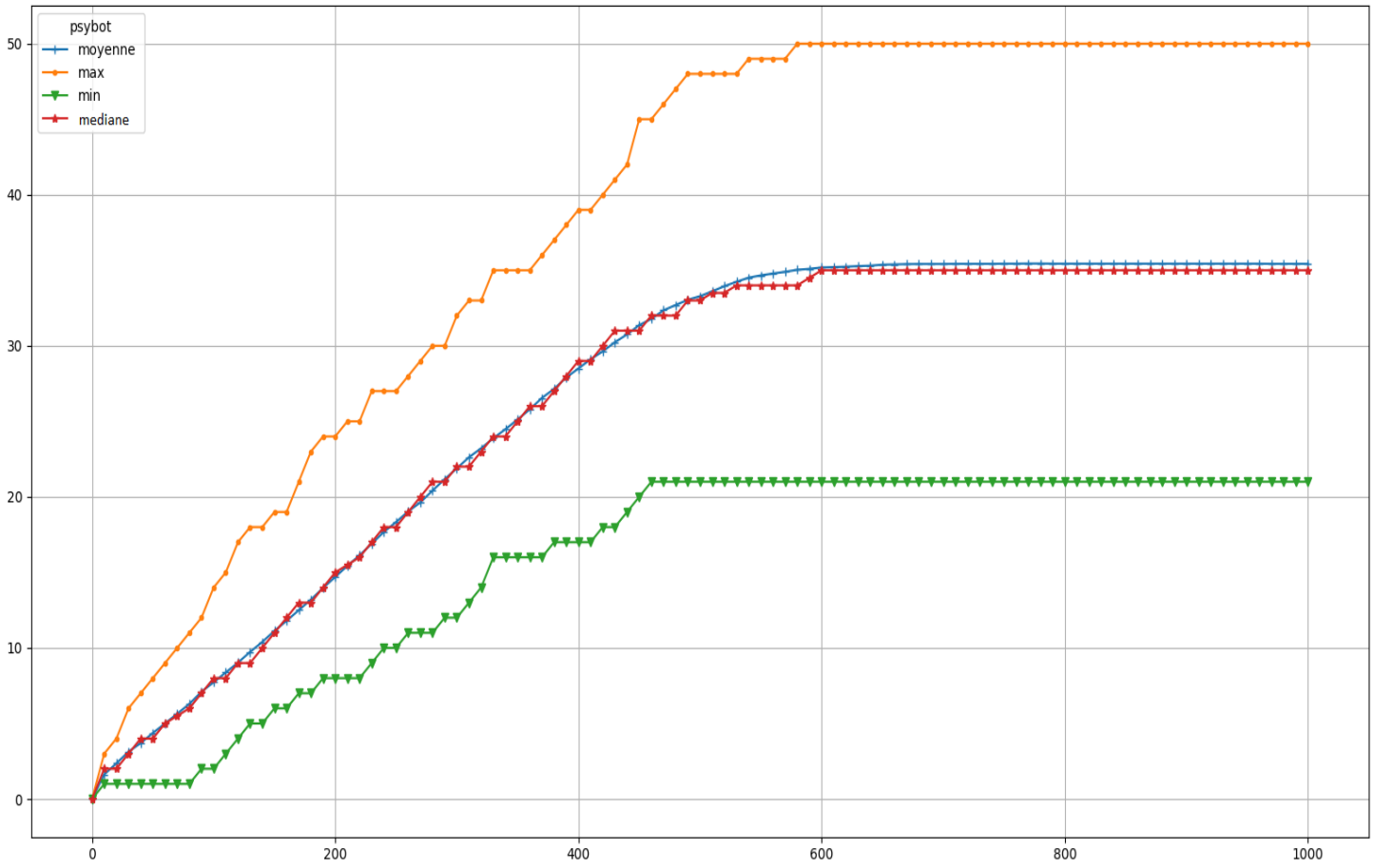

Figure 4.17 – Évolution de la population du botnet #1 sur 1000 tours

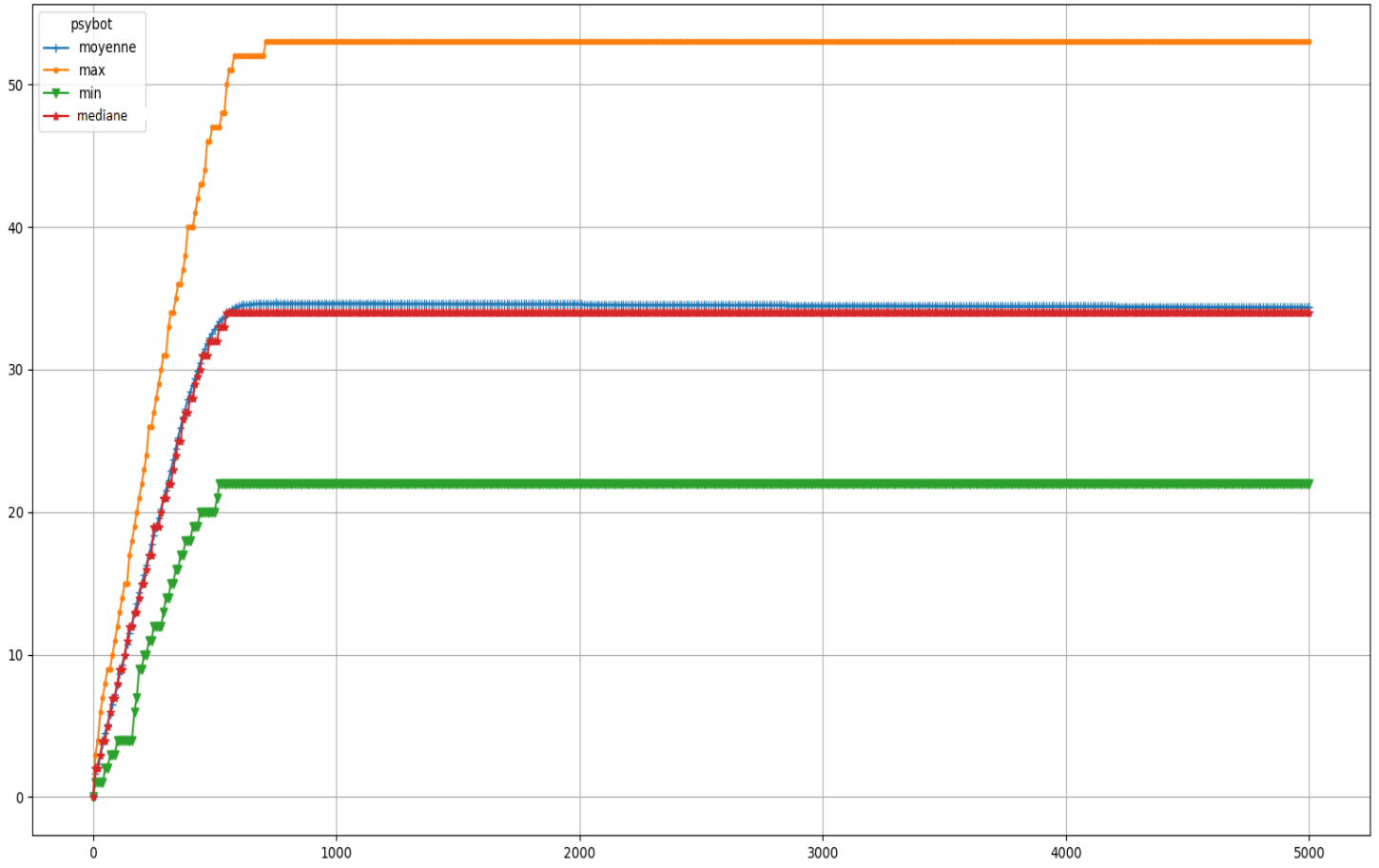

Figure 4.18 – Évolution de la population du botnet #1 sur 5000 tours

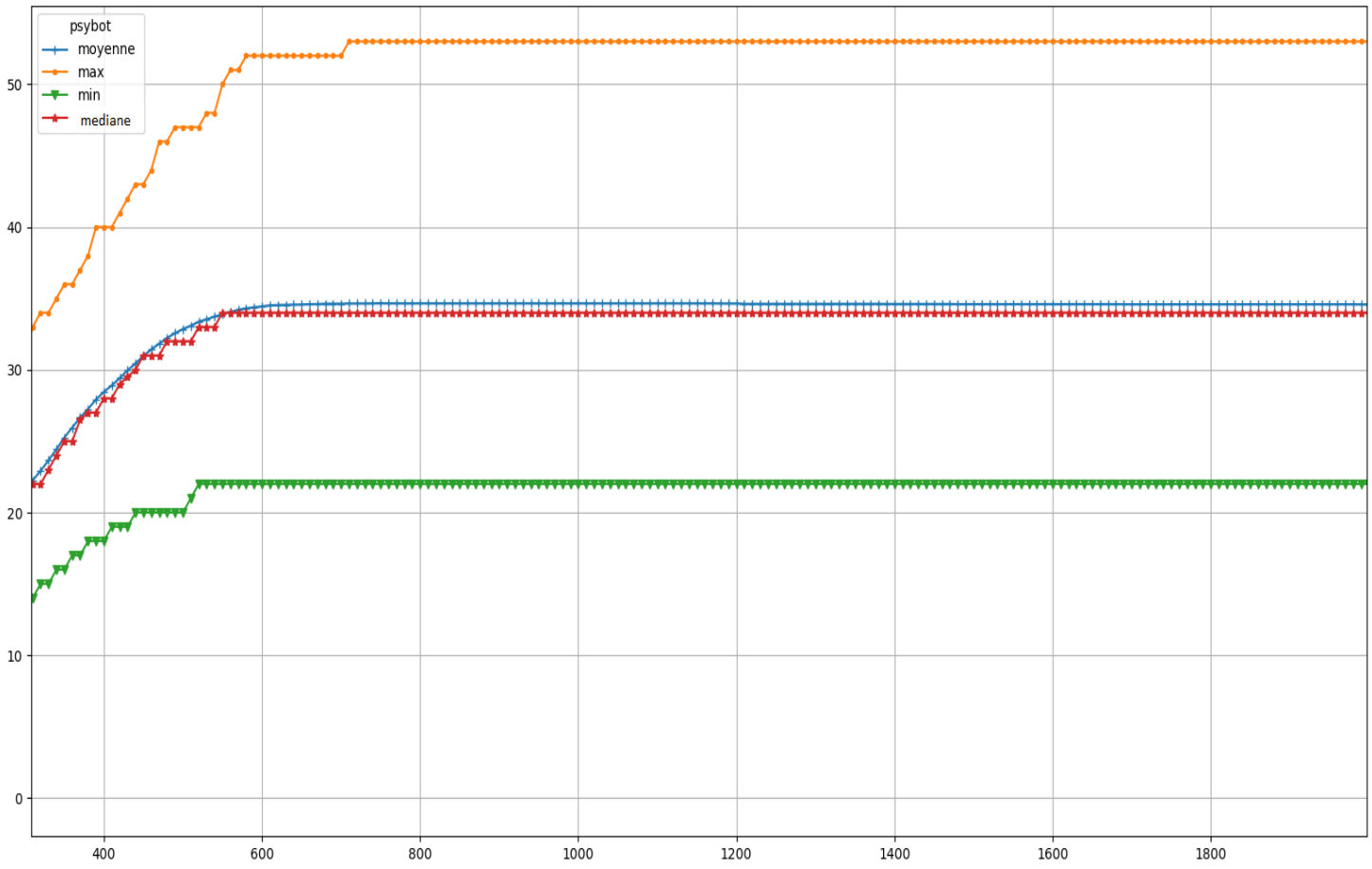

Figure 4.19 – Zoom de l'évolution de la population du botnet #1 sur 5000 tours

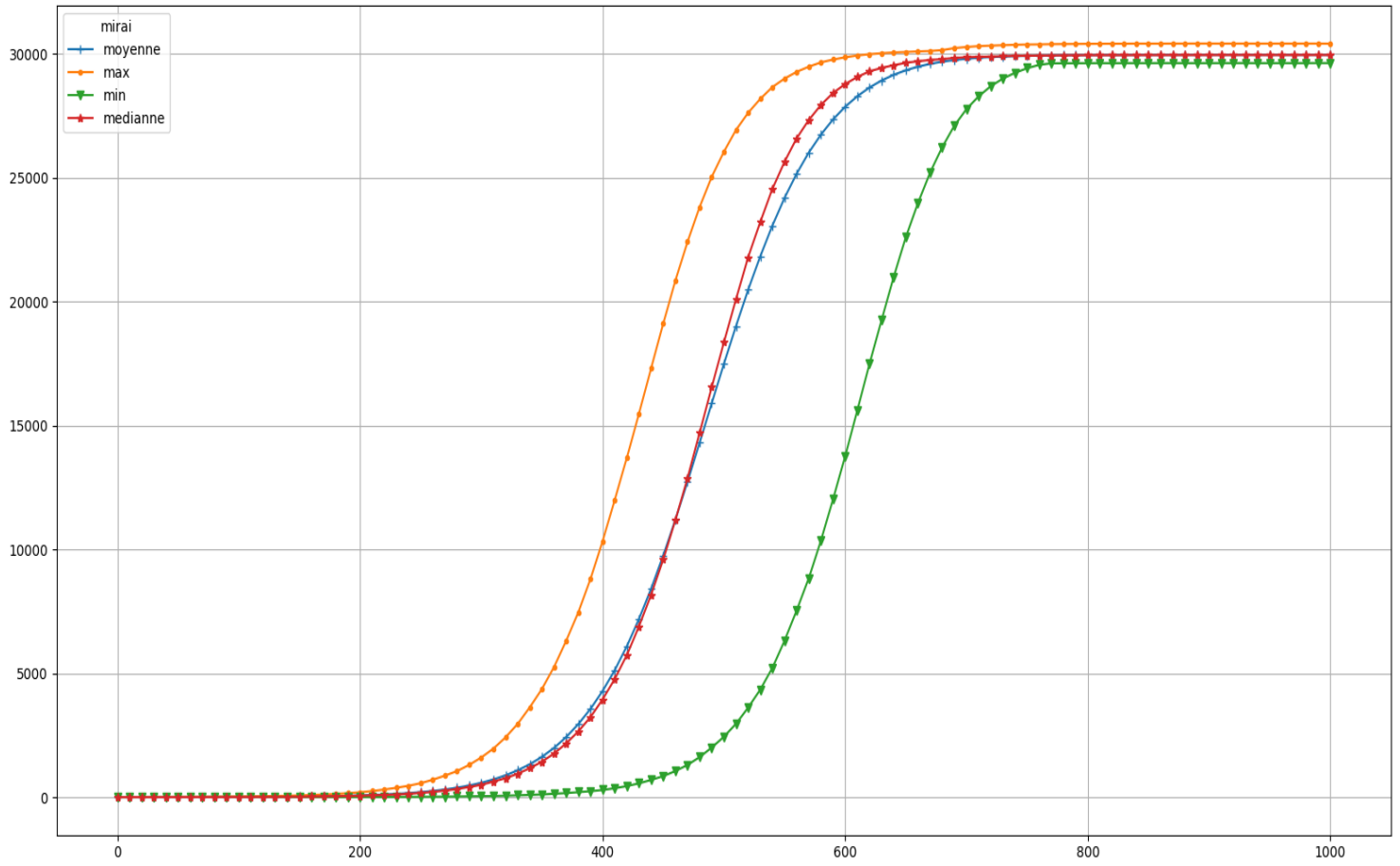

Figure 4.20 – Évolution de la population du botnet #2 sur 1000 tours

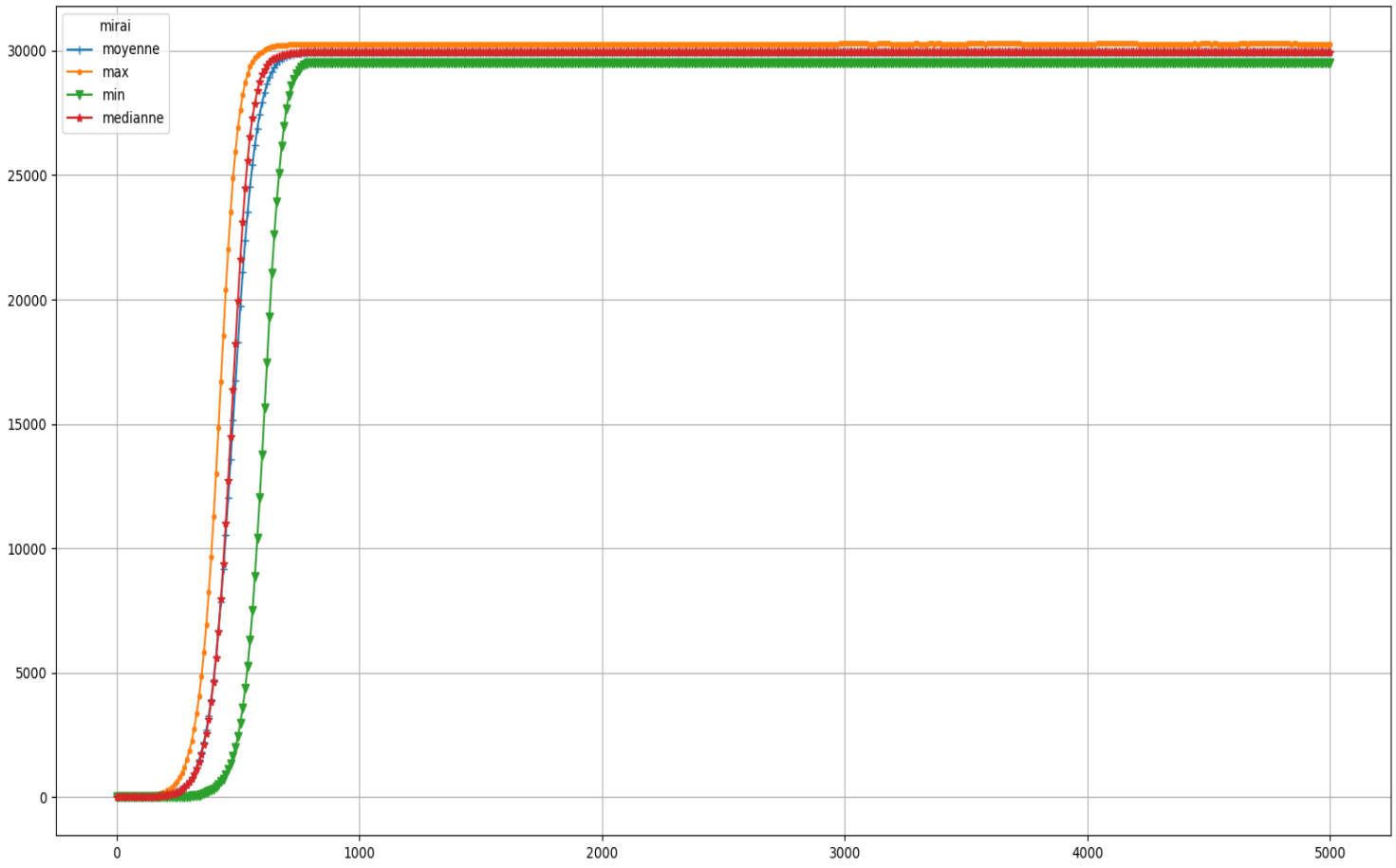

Figure 4.21 – Évolution de la population du botnet #2 sur 5000 tours

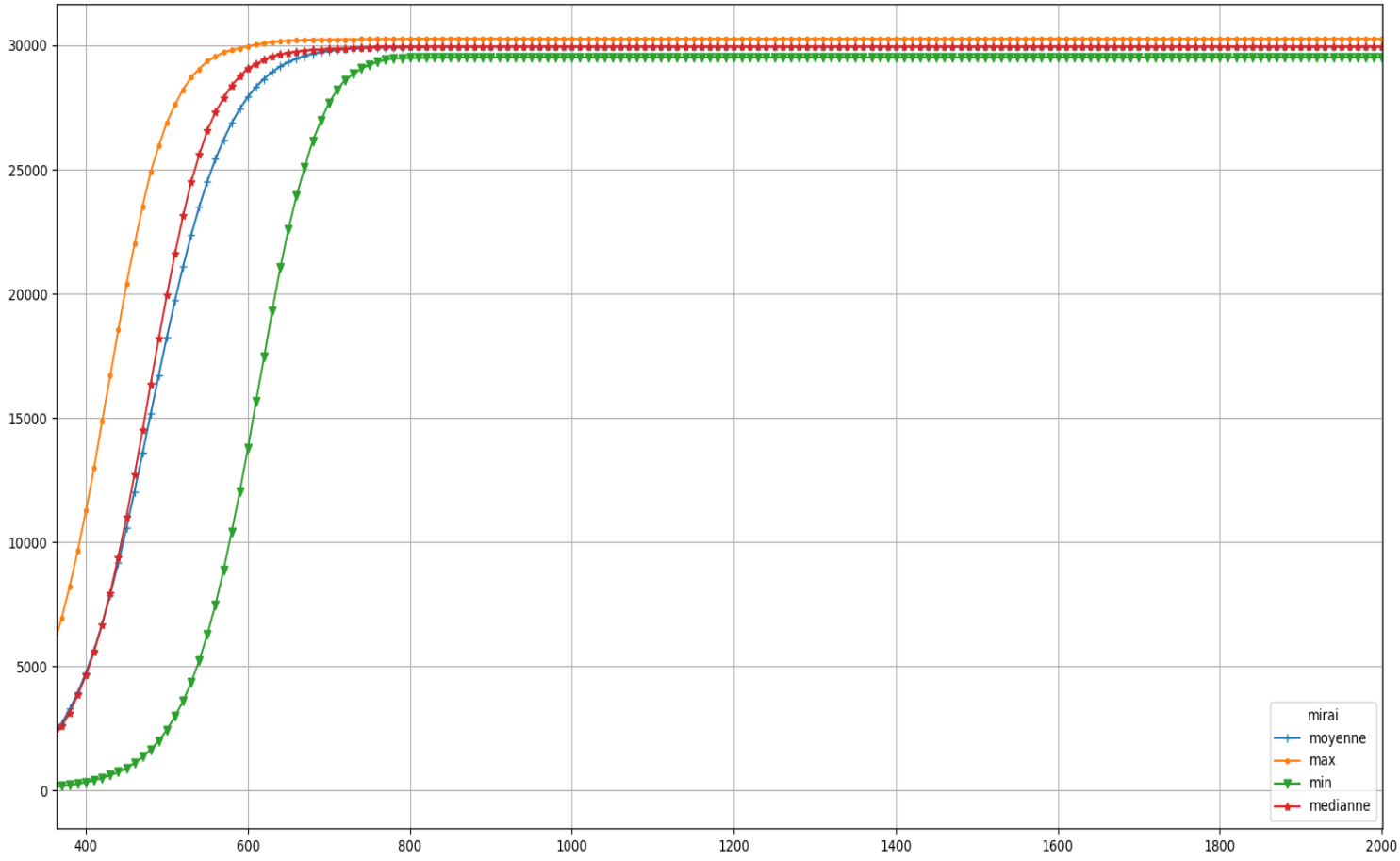

Figure 4.22 – Zoom de l'évolution de la population du botnet #2 sur 5000 tours

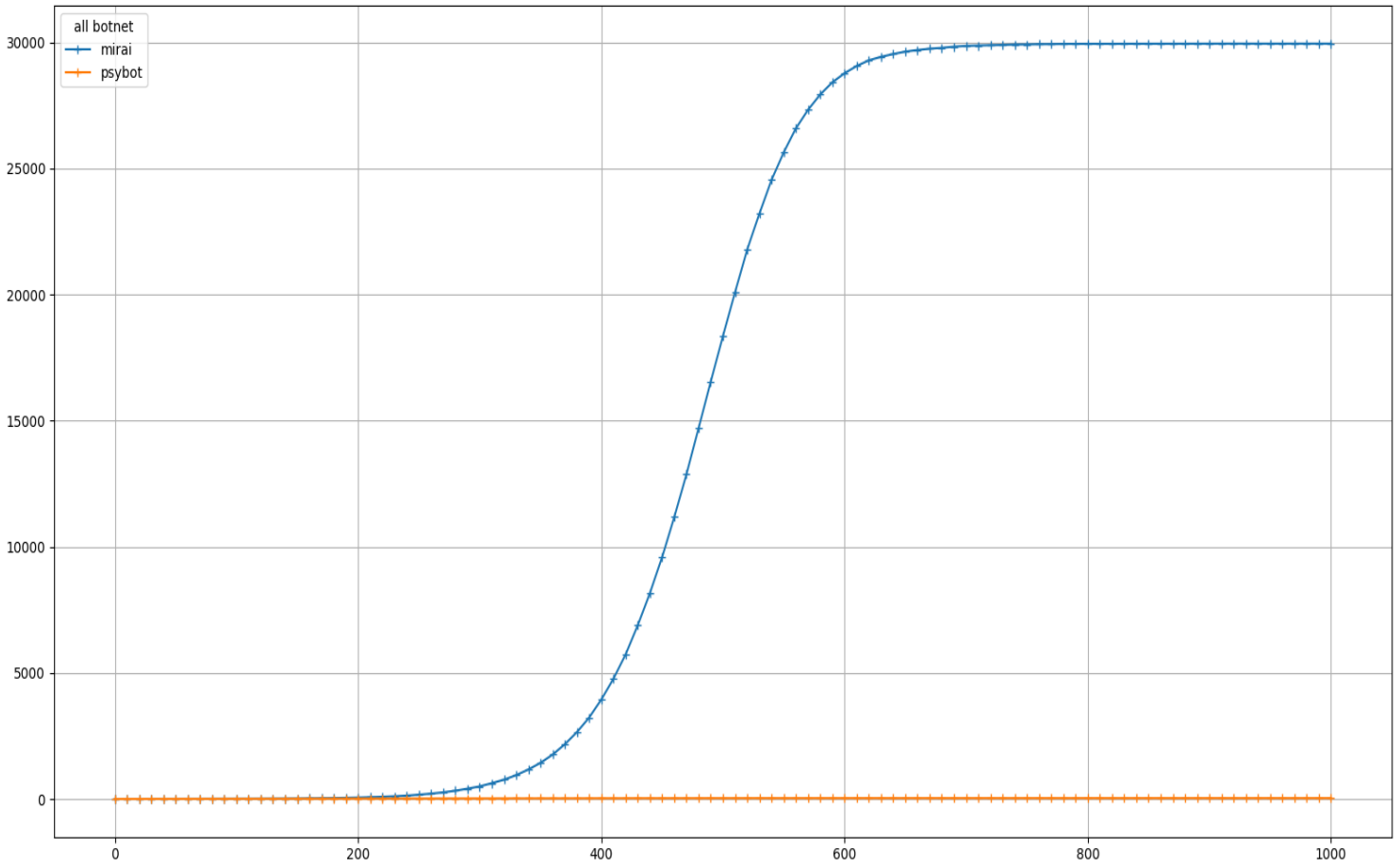

Figure 4.23 – Évolution de la population des deux botnets sur 1000 tours

4.3.5 EXPÉRIENCE 3A

Pour cette expérience, nous souhaitions observer l'impact d'un ver capable d'infecter les objets déjà infectés par d'autres programme malveillants et de supprimer l'infection, tout en immunisant la victime. Ce cas de figure est apparu ces dernières années. En effet, comme le précise Radaware dans un de ses articles de blogs (Daniel, 2018), la libération du code source

de Mirai a entraîné une saturation de réseaux de zombies. Ainsi, chaque réseau a dû innover, en incluant des systèmes de correctif pour supprimer les autres réseaux de zombies. Ils ont aussi commencé à exploiter différentes failles afin d'avoir de nouvelles méthodes pour recruter des zombies.

Pour cette première version, nous avons utilisé deux réseaux de zombies identiques, utilisant le scan aléatoire et le second immunisant ses victimes contre les deux réseaux. Le second est aussi capable de supprimer l'infection du premier réseau. De plus, le second réseau part avec un retard de deux cents tours, soit suffisamment pour que le premier réseau puisse atteindre une croissance exponentielle.

La figure 4.24 représente l'évolution de la population du premier réseau de zombies, n'immunisant pas ses victimes. La figure 4.25 représente l'évolution de la population du second réseau de zombies, capable d'immuniser ses victimes et de les désinfecter du premier réseau. La figure 4.26 représente l'évolution de la population médiane des deux réseaux.

On peut ainsi observer sur la figure 4.25 que la population du premier réseau est capable de grandir rapidement, pouvant parfois infecter la quasi-totalité de la population vulnérable. Cependant, aux alentours du tour 600, la population chute très rapidement, pour atteindre 0 individu entre le tour 800 et 1 000. Au contraire, la population du second réseau augmente normalement et finit par atteindre l'équilibre vers le tour 1 000, soit avec 200 tours de retard par rapport au cas classique.

Sur la figure 4.26 on observe que le point d'inversion de courbe du premier réseau, correspond au point d'inflexion de la courbe du second réseau. Ainsi, au moment où le second réseau entame sa phase de croissance exponentielle, il va « voler » des victimes au premier réseau.

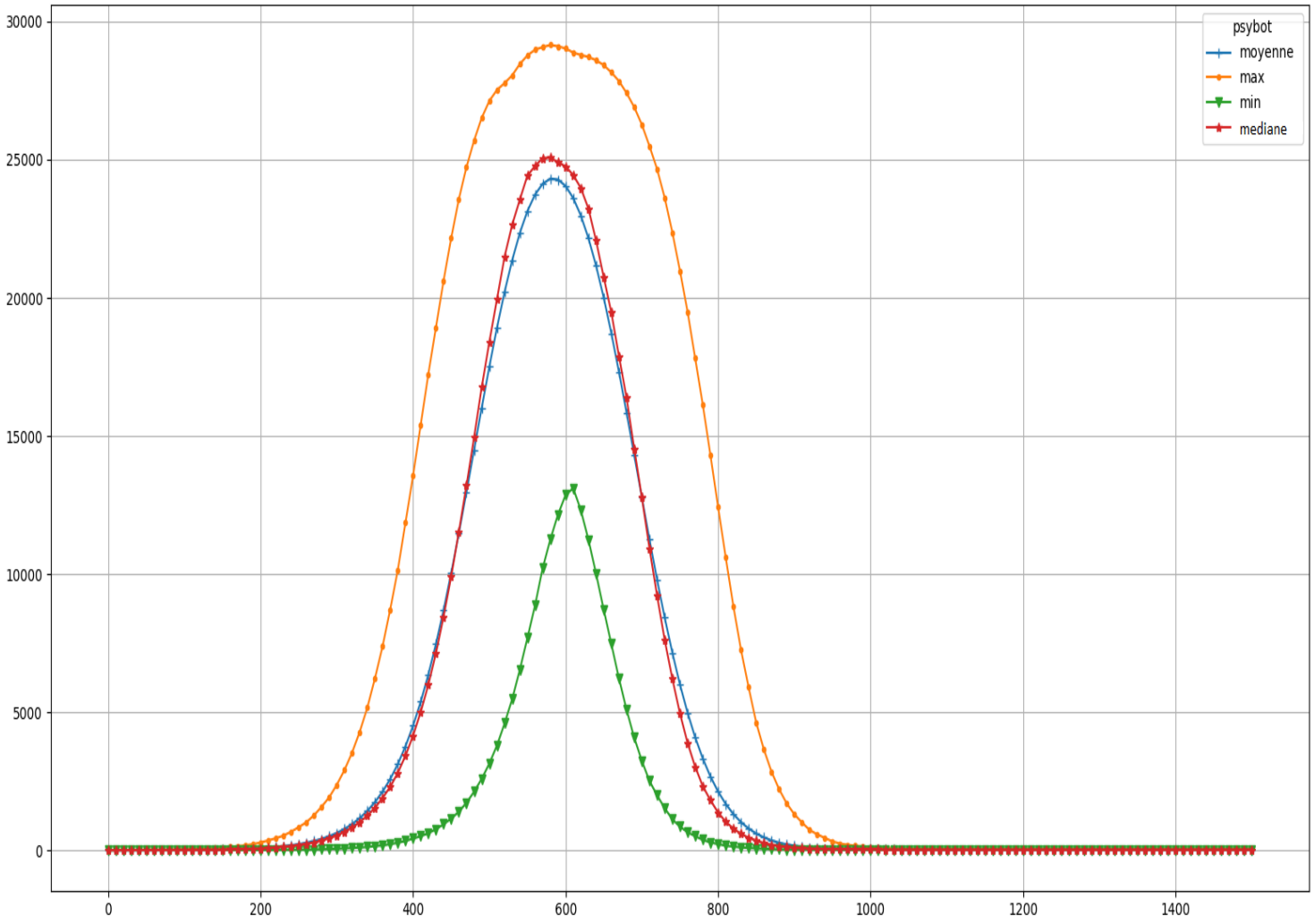

Figure 4.24 – Évolution de la population du botnet #1 sur 1 500 tours

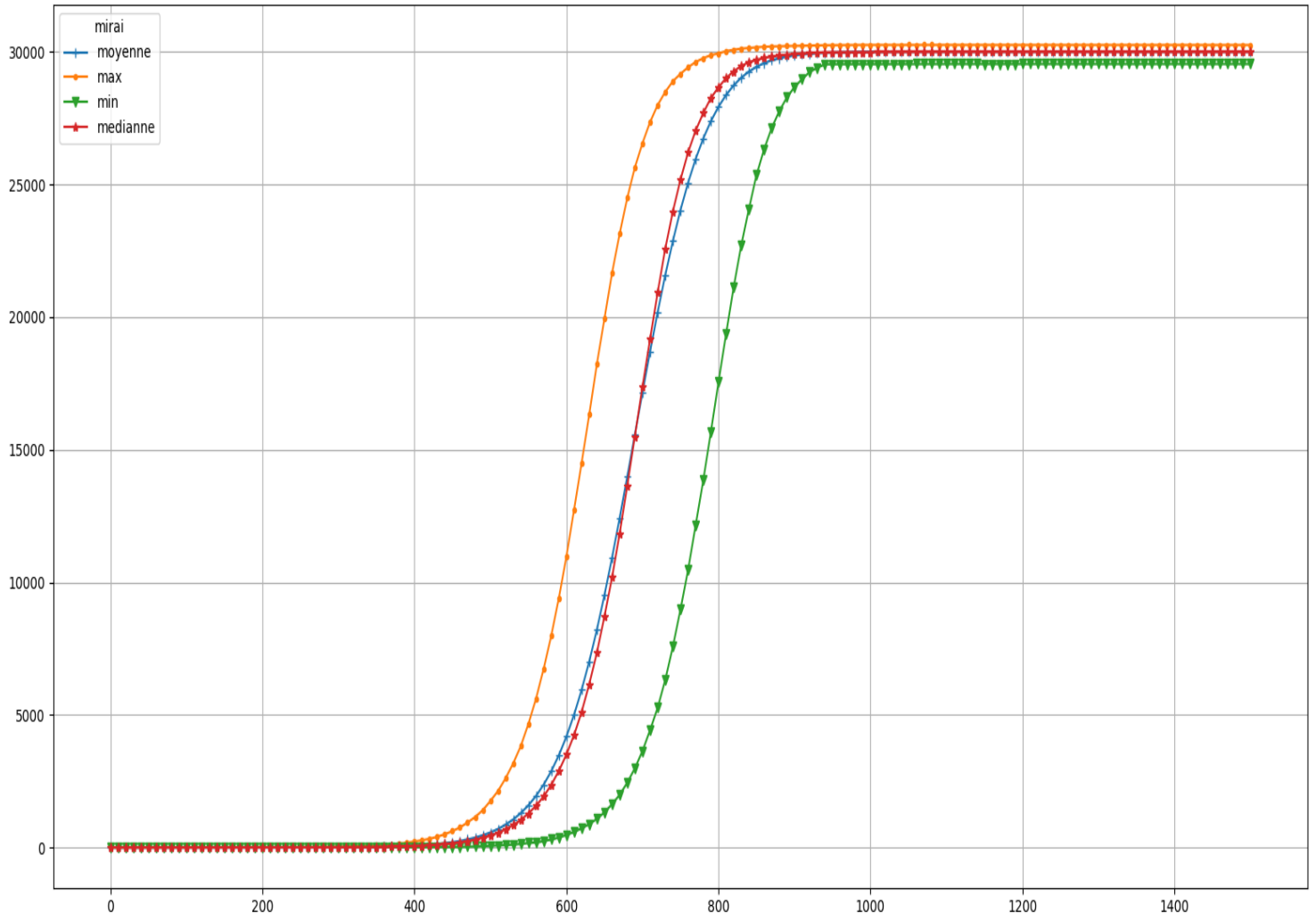

Figure 4.25 – Évolution de la population du botnet #2 sur 1 500 tours

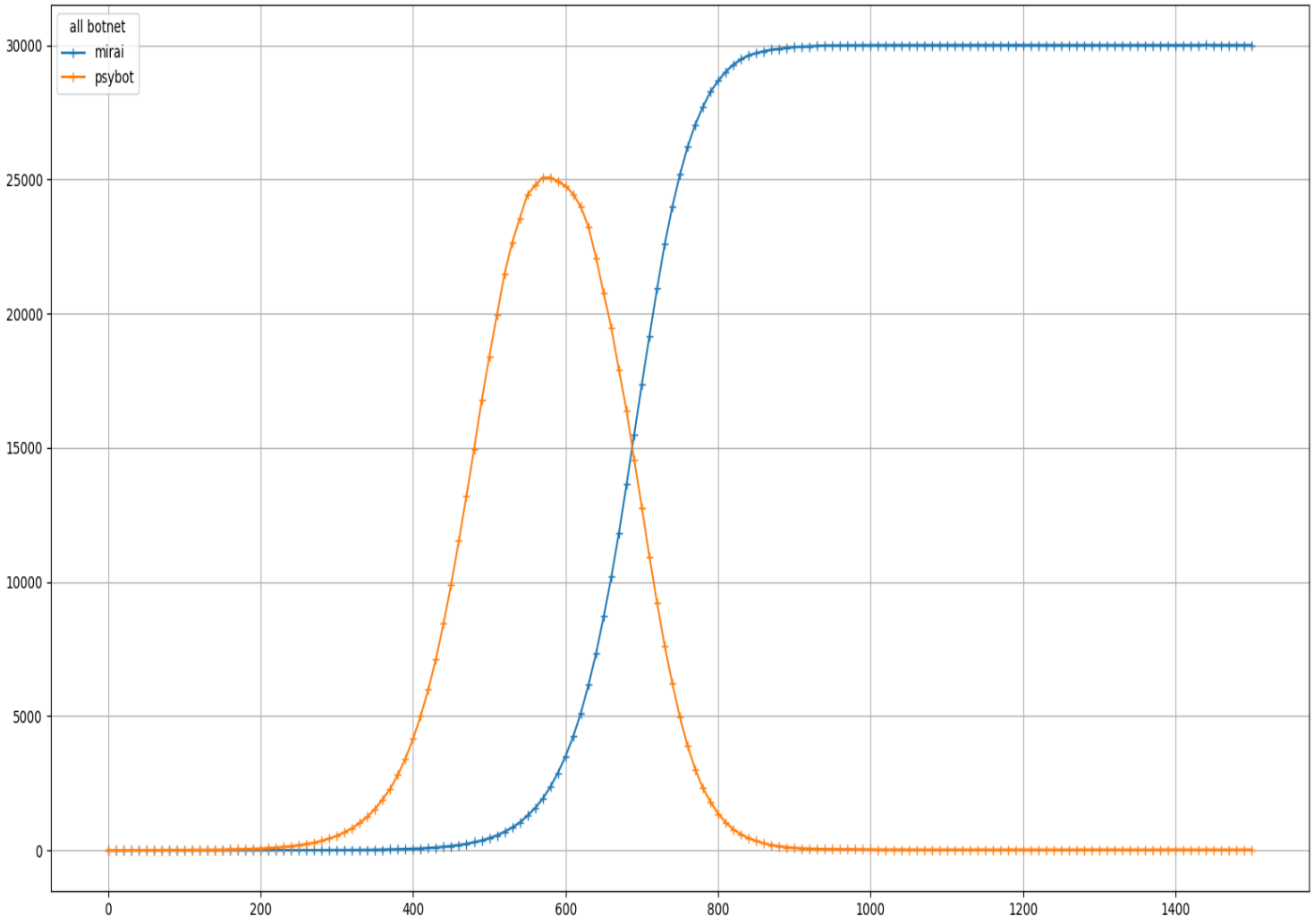

Figure 4.26 – Évolution de la population des deux botnet sur 1 500 tours

4.3.6 EXPÉRIENCE 3B

Cette expérience est la même que la précédente, à la différence que le premier réseau utilise un scan séquentiel tel que défini plus tôt. La figure 4.27 représente l'évolution de la population du

premier réseau. La figure 4.28 représente l'évolution de la population du second réseau. Enfin, la figure 4.29 représente l'évolution des populations médianes des deux réseaux. Les résultats sont similaires à la première expérience, à la différence que le premier réseau atteint au maximum 55 individus et non 30 000. L'évolution du second réseau est ici identique à son homologue de l'expérience 3A. A la figure 4.29 on observe bien que le premier réseau ne parvient à recruter que peu victimes et que ses dernières finissent par se faire recruté par le second réseau. A la fin on observe bien la mort du premier réseau terminant avec 1 seul individus, le serveur originel.

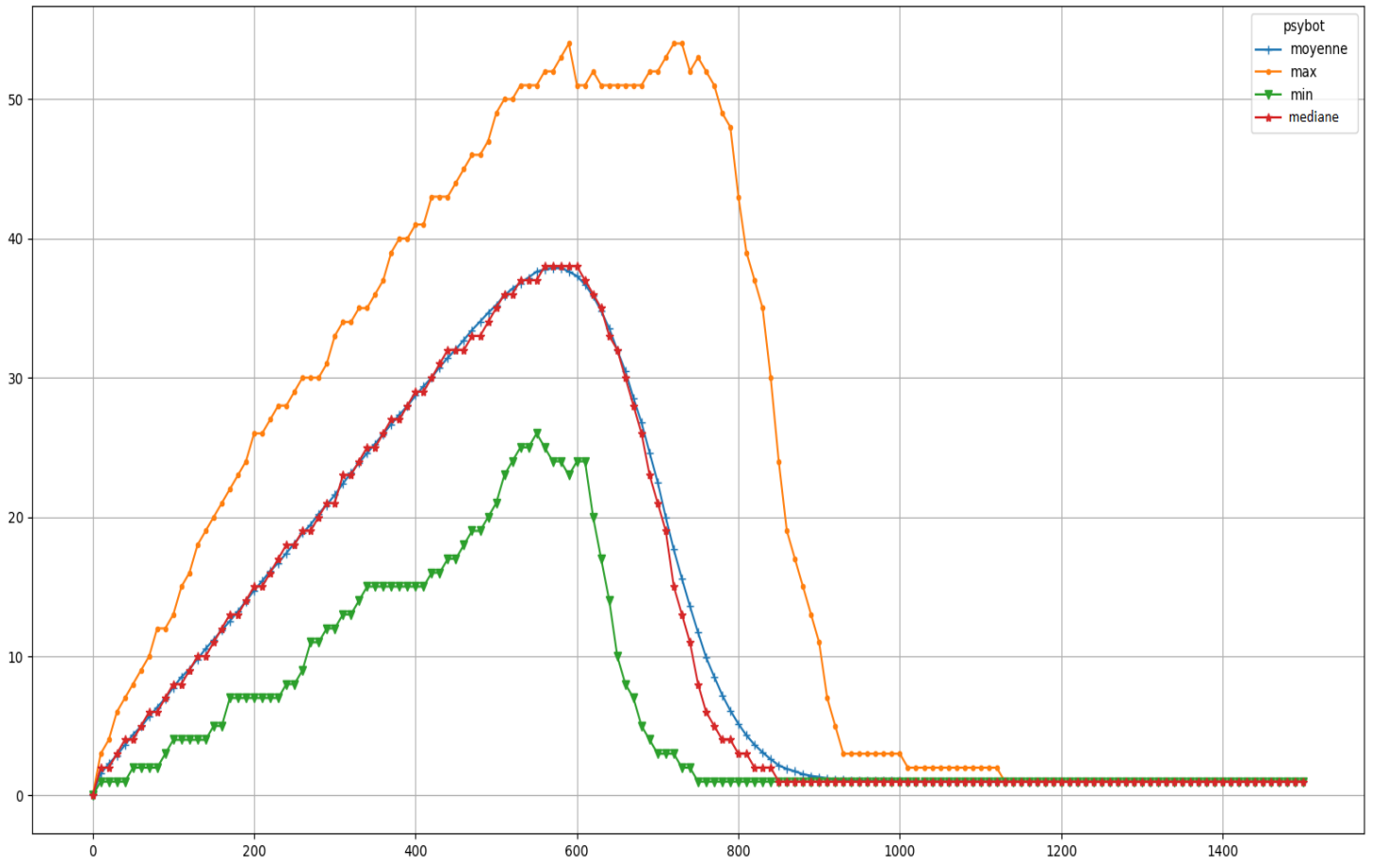

Figure 4.27 – Évolution de la population du botnet #1 sur 1500 tours

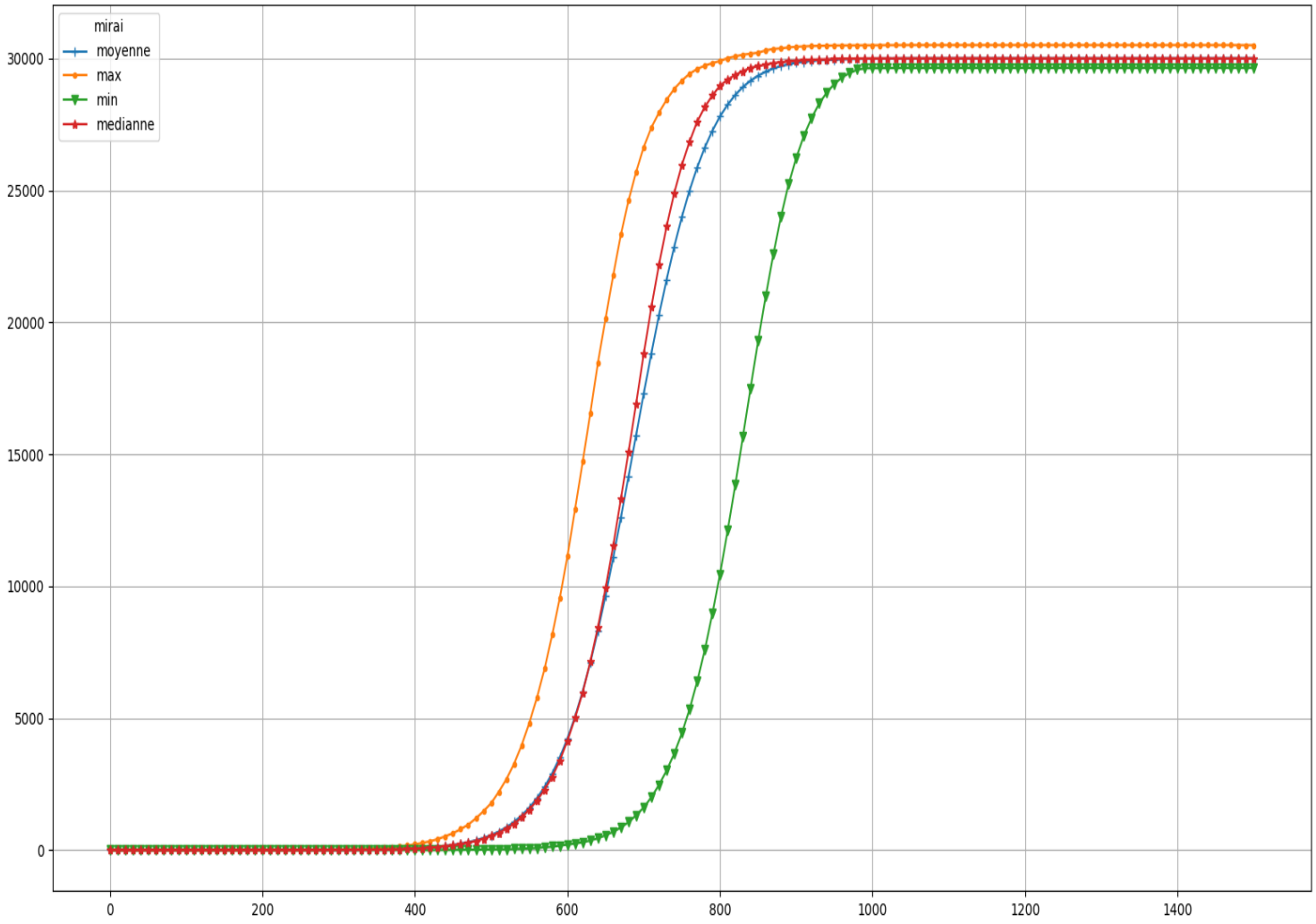

Figure 4.28 – Évolution de la population du botnet #2 sur 1500 tours

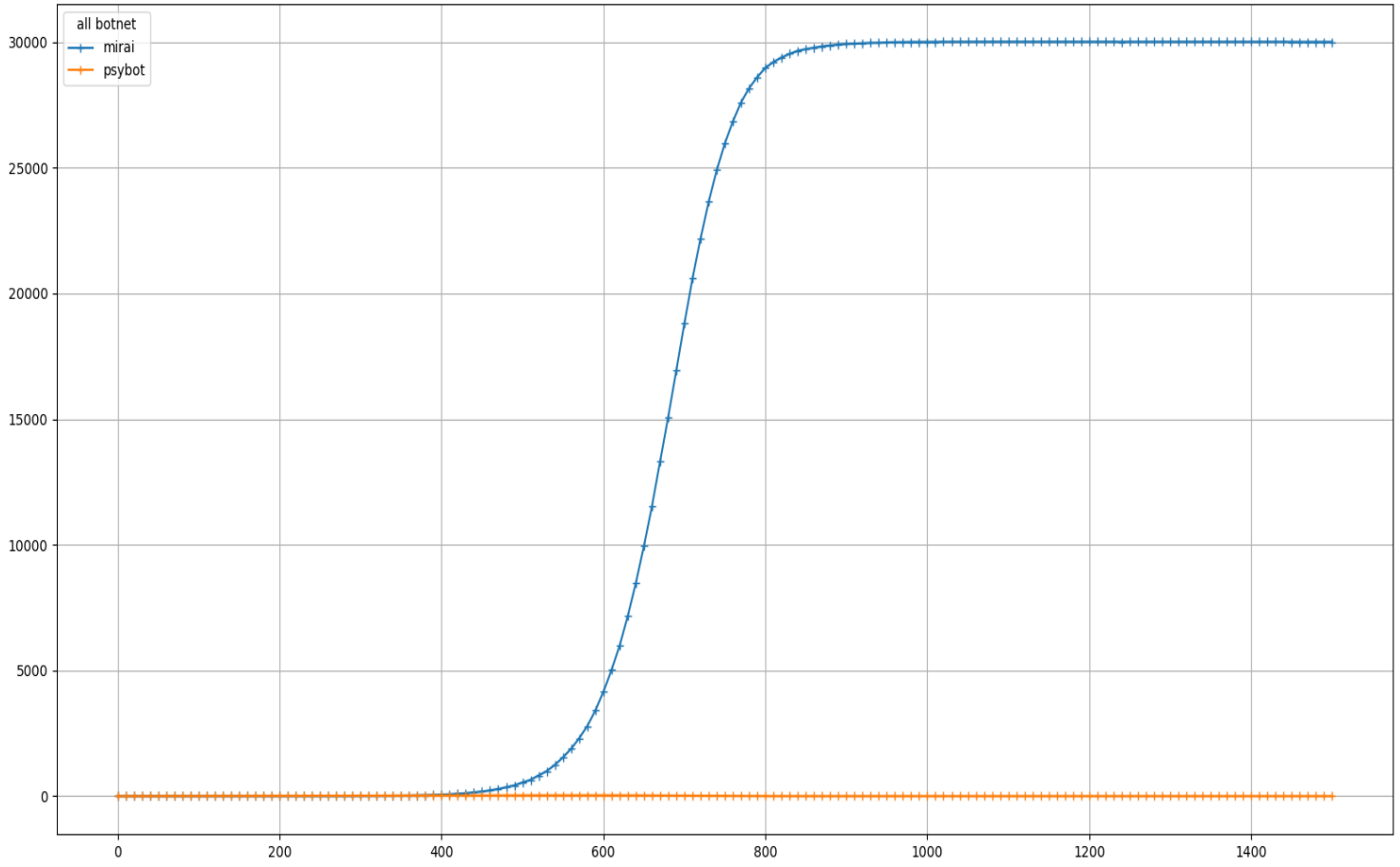

Figure 4.29 – Évolution de la population des deux botnet sur 1500 tours

L'ensemble de ces expériences à pu nous donner beaucoup de résultats pertinents, qui nous permettrons de répondre à plusieurs de nos interrogations concernant l'impact des fonctionnalités implémentées par des réseaux de zombies. Nous interpréterons et critiquerons ces résultats dans le prochain chapitre.