## Le bridge

A l’origine, un root Bridge (pont) comportait 2 ports pour relier deux pc.

L’objectif de ce graphe est d’obtenir un seul chemin unique dans le réseau.

Le pont, est un élément actif dans un réseau électrique. Il contient de l’électronique et doit analyser les trames ethernet pour savoir s’il doit retransmettre ces trames sur un autre bus.

Il permettait de segmenter le réseau et de le rendre plus rapide.

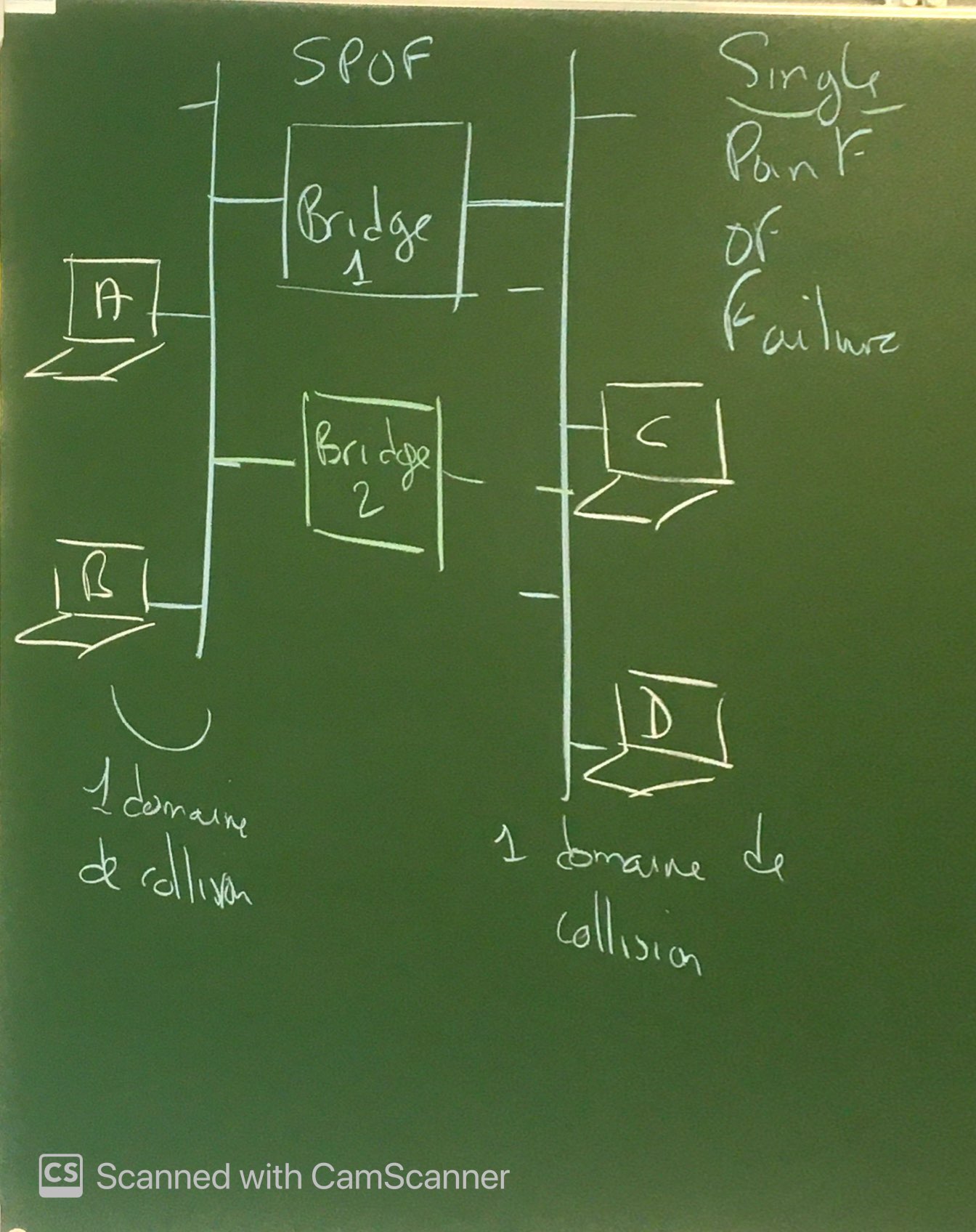
**« Spof » = single point of failure**

Le pont est un spof car il est susceptible de tomber en panne et une architecture en bus, rend le réseau plus vulnérable.

La solution pour éviter ce spof, c’est d’ajouter au moins un autre bus. Cependant, cette architecture va anéantir le réseau car les deux vont retransmettre les trames de broadcast en même temps, ce qui va faire des trames en doublons, entraînant une tempête de broadcast.

La solution à ce problème c’est de supprimer l’accès logique du bridge 2 au réseau et non physiquement car cela permet de garder une redondance au cas où le Bridge 1 tombe.

La solution logicielle permettant d’assurer cette redondance (l’algorithme) doit proposer une solution sans ambiguïté.



## L’algorithme de Spanning Tree

Différentes étapes

1. **Bloquer les ports sur les pc pour empêcher une boucle pour les data, pas pour les BPDU**

Seuls les switches peuvent communiquer sur BPDU. On enlève toute possibilité qu’une boucle puisse avoir lieu.

1. **Le root bridge élection (élection du pont racine)**

Déterminer le chemin unique. Chaque switch envoi un BPDU avec son bridge ID.

* Quelle méthode de propagation ?
* Véhicule : Les trames traitées par cet algorithme sont les trames BPDU (Bridge Protocol data unit). Elles sont le véhicule.
* Méthode de propagation : chaque switch copie les BPDU à ses voisins.
* L’idée est de choisir le pont racine à partir duquel faire le calcul. Comment choisir le pont ?

Chaque pont possède un Bridge ID. Il est composé de deux parties :

* Bridge Priority -> permet de configurer le bridge de manière à ce que l’algorithme converge dessus (entier).
* Mac Address -> garantit la convergence, car permet d’identifier de façon unique (48 bits).

Le meilleur bridge est le plus petit Bridge ID. « Out of the box », un bridge possède un Bridge ID medium. Il est possible ensuite de modifier cette priorité par configuration.

* L’élection

Tout switch qui reçoit une BPDU meilleure que la sienne arrête d’envoyer la sienne. Il ne reste plus que celui qui a la meilleure qui cause. Tout le monde a reçu la BPDU du meilleur switch.

Toutes les deux secondes, il émet sa BPDU comme message de survie avec un timer.

1. **Déterminer mes « route port » et « designated port »**

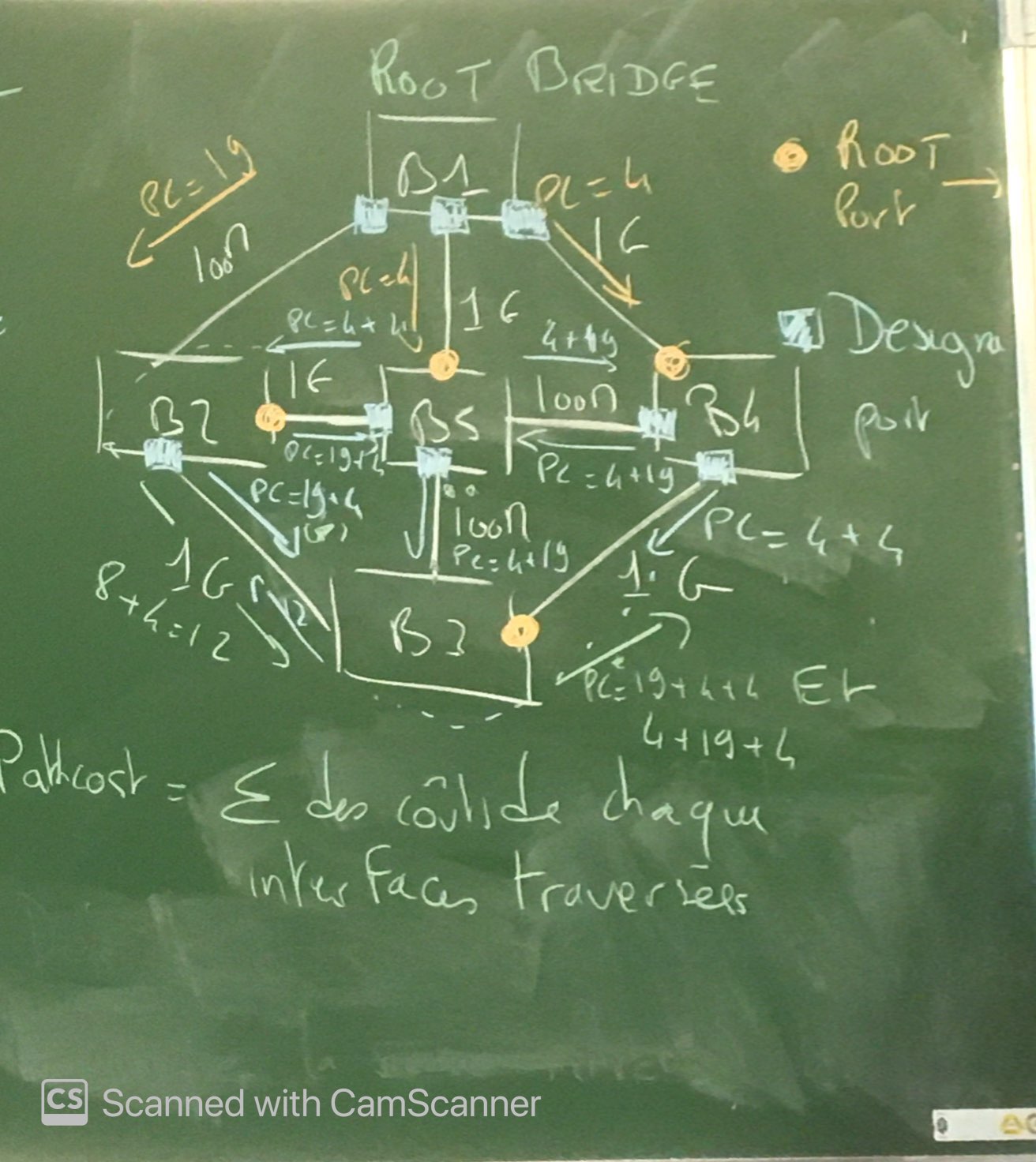
**« Route port »** -> celui par lequel je reçois mes BPDU, celui qui a le plus de bande passante pour aller vers le route bridge.

**«** **Designated port »** -> ports qui vont vers d’autres segments. Ils transmettent la meilleure BPDU.

On va choisir ceux qui sont les plus rapides.

Il faut déterminer le « pathcost ». Il s’agit de la somme des coûts de chaque interface utilisée.

Le root port c’est le meilleure chemin pour aller au root bridge.



Orange = première émission, bleu, 2e émission, vert 3e émission.

Le route port est celui qui reçoit la meilleure BPDU (ronds sur schéma) / path cost. Ainsi, chaque switch a trouvé son meilleur port.

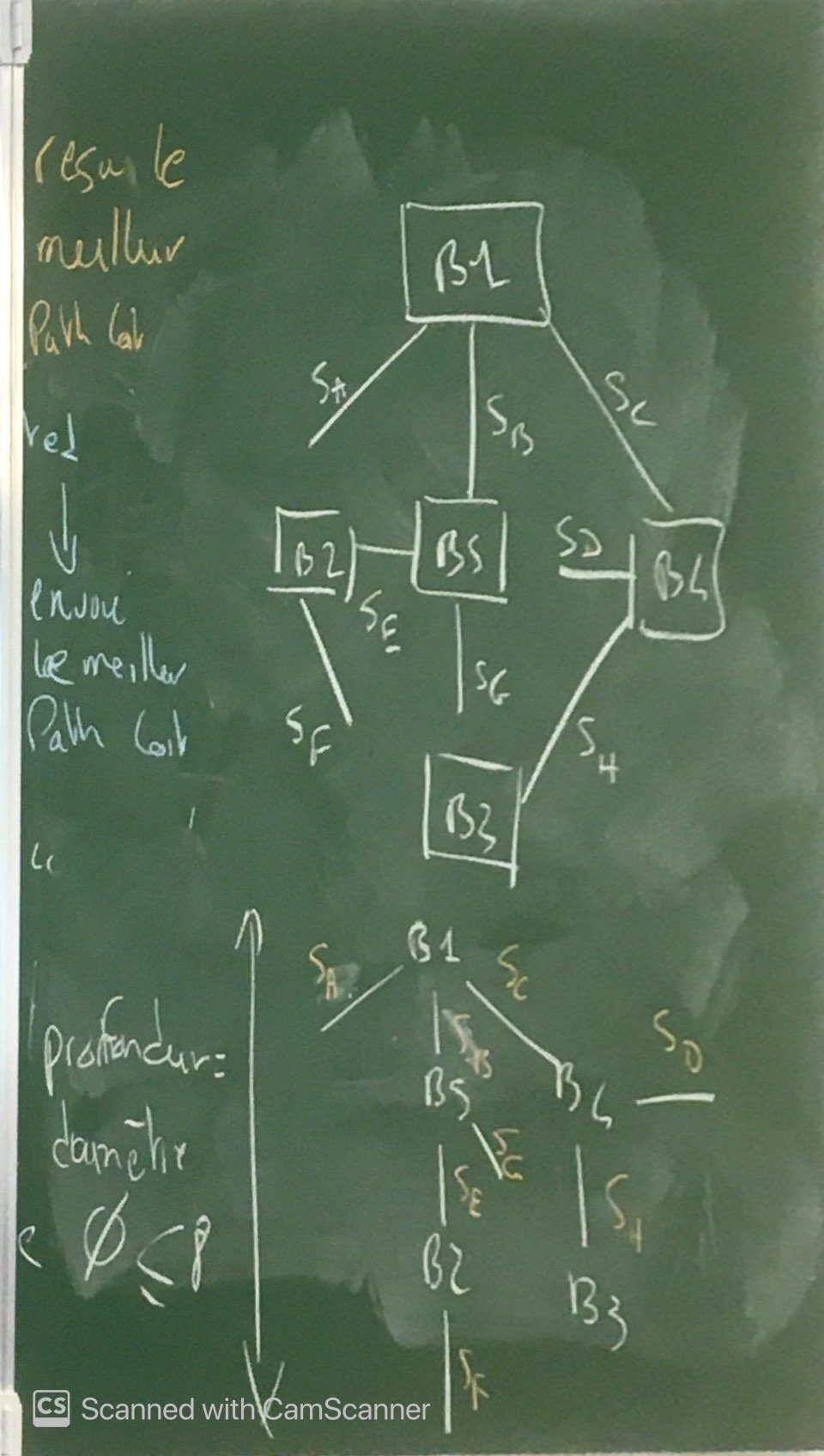
Mais ils doivent encore désigner ceux qui renvoient le meilleur path cost. Ce sont celles qui partent du root bridge (B1).

Au premier envoi de protocole broadcast. Chaque interface utilisée possède un pathcost différent. (PC = 19 ..) Si deux ports émettent un BPDU identique vers un même switch, on va utiliser le numéro du port le plus petit pour faire le choix (ex sur B3).

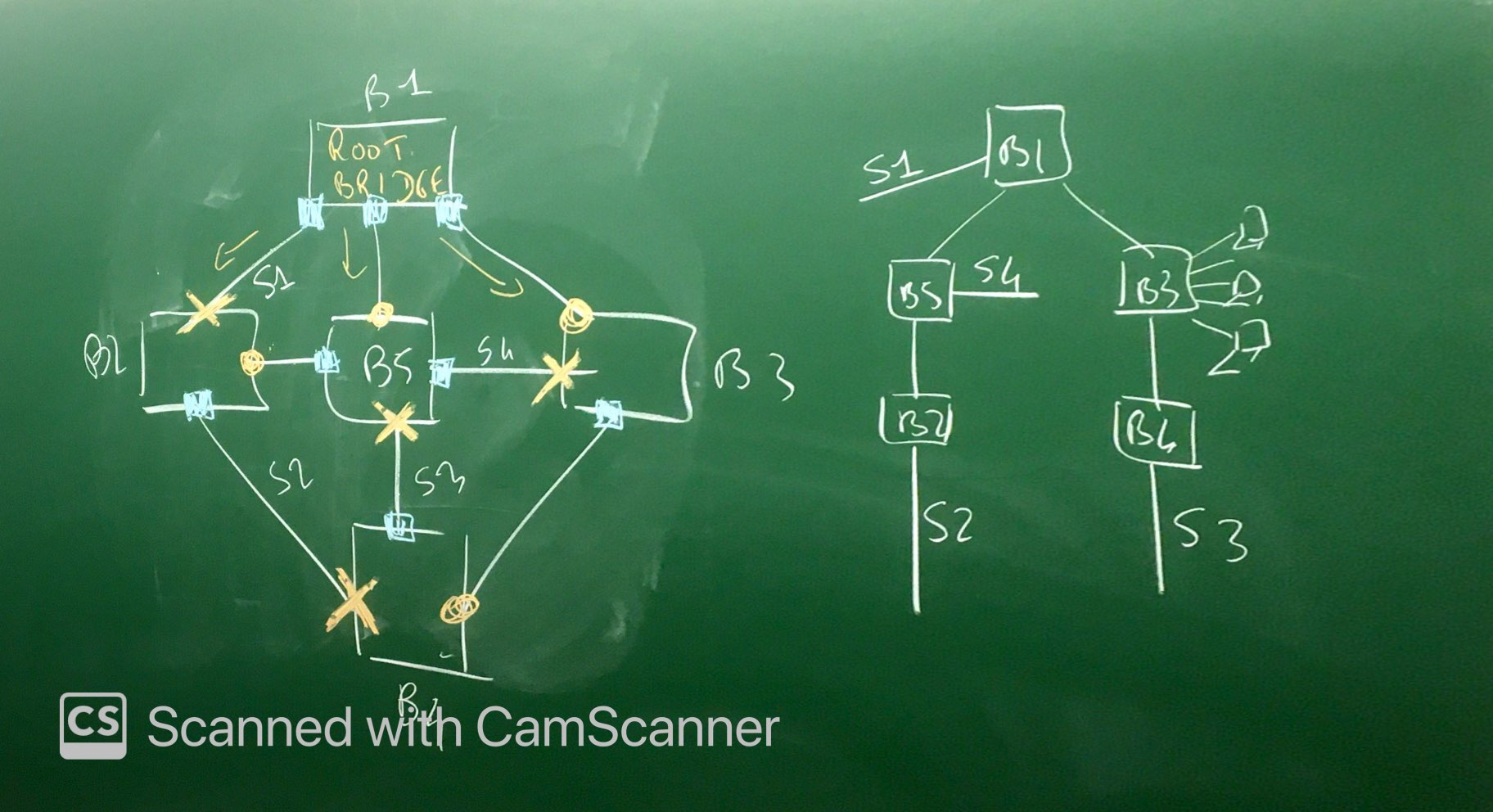
A la fin, tous les ports non choisis (comme étant meilleurs pour la réception et l’émission) sont bloqués.

Au final, on se retrouve avec ce schéma simplifié

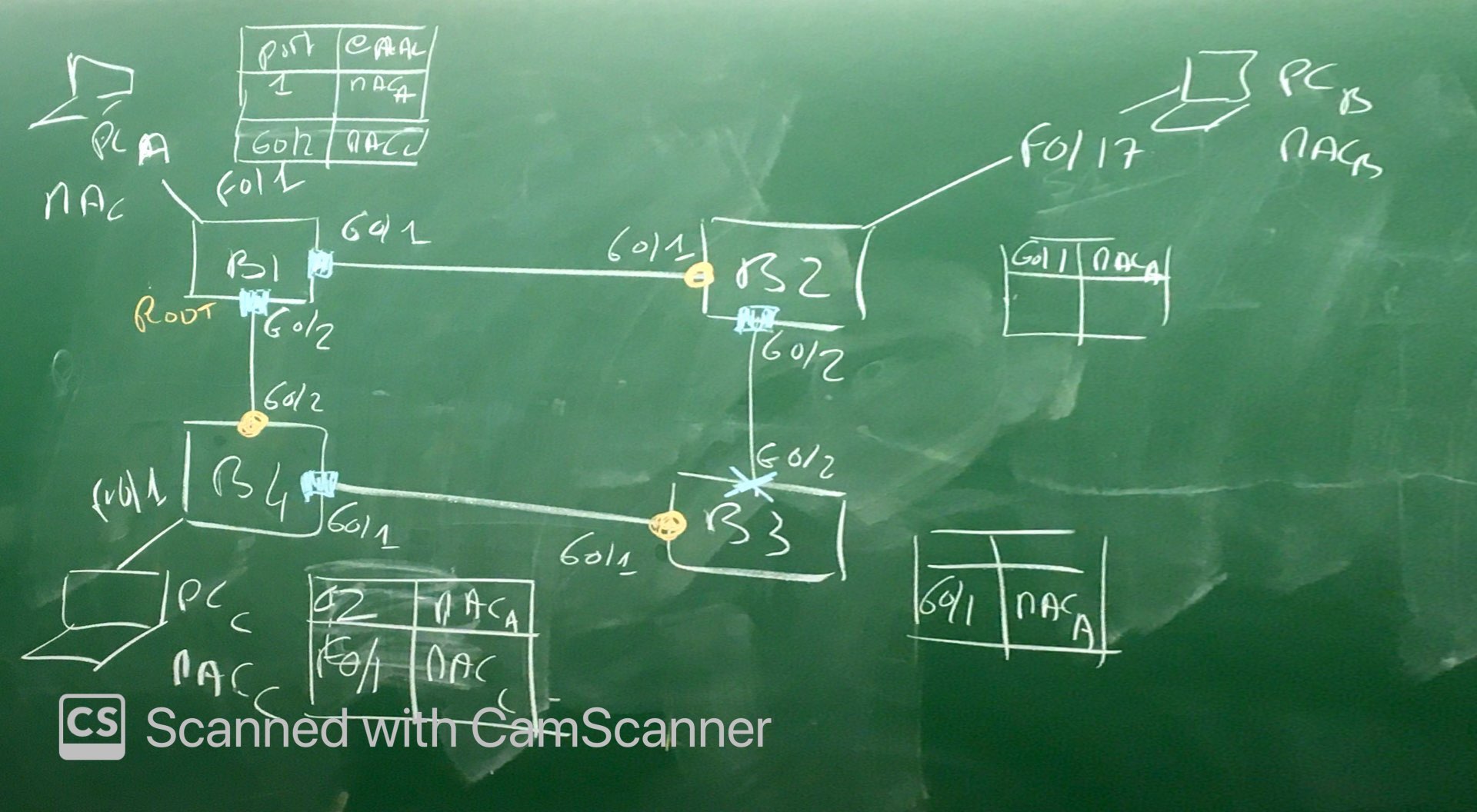
Toutes les machines pourront à nouveau rediscuter.

Si le lien entre B2 et B5 est arrêté, il bloque tous les ports (au sens du BPDU) et recalcule. L’idée initiale étant d’avoir de la redondance.

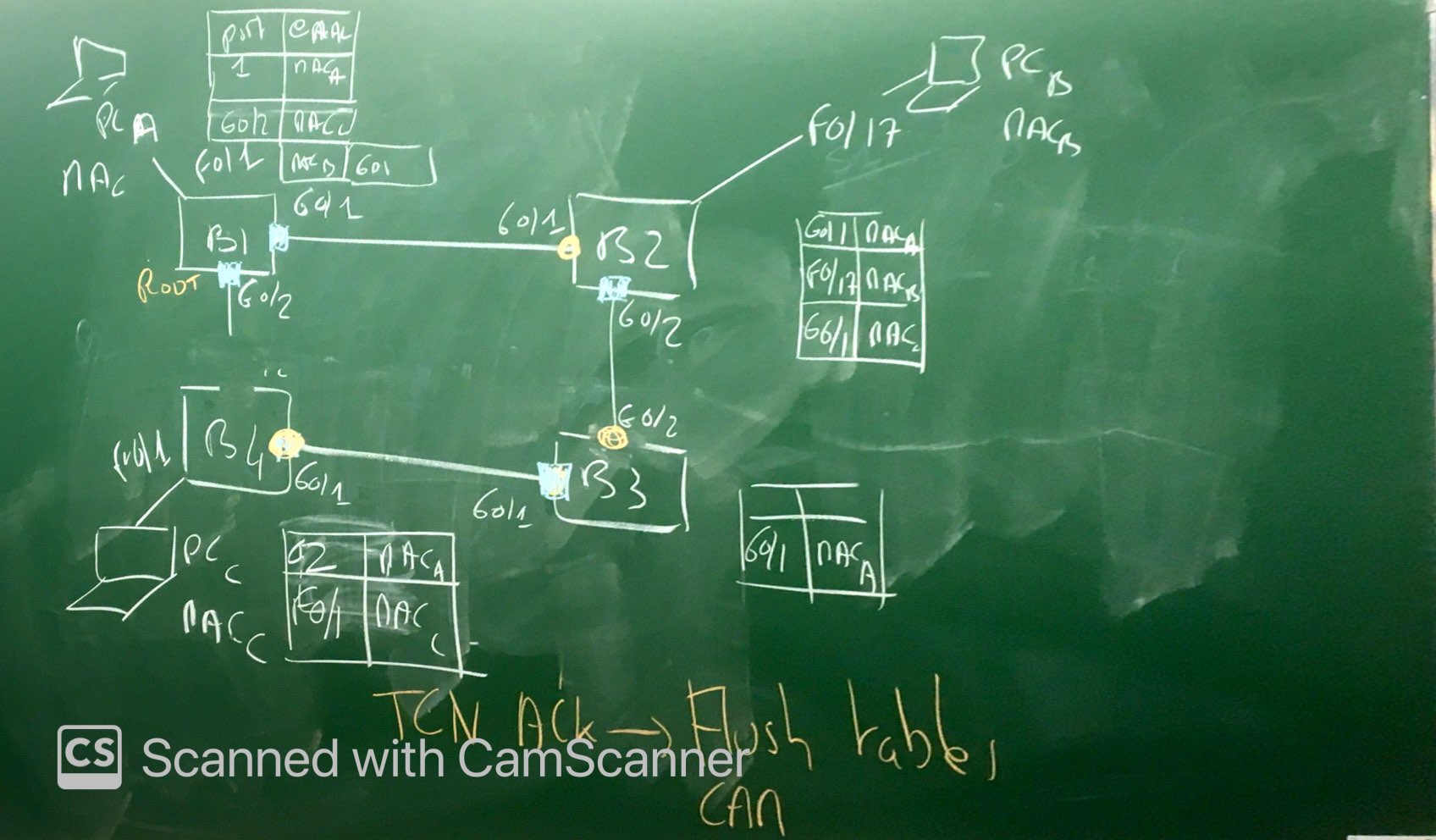
**Rappel de fonctionnement**



**Autre exemple de Spanning tree**



**Changement de structure suite à cassure du lien B4 – B1**



**TCN (Topology change notification)** : protocole envoyé, lorsqu’un lien se casse, au root bridge. Le root bridge renvoie un TCN ACK à tous les switchs. Chaque switch fait un « flush » de sa table et efface toutes les adresses. Tous les chemins sont reconstruits. Il font donc que le spanning tree prévienne tous les switchs que l’ancien chemin ne fonctionne plus. Il sert à synchroniser l’état sur Spanning tree et le réseau physique.

<https://mrproof.blogspot.com/2010/12/architecture-des-reseaux-locaux.html>

* Timer des trois phases du spanning tree ?

**Hello time** entre chaque BPDU, envoyé sur un port, message de survie : 2 secondes par défaut

**Forward delay timer** : intervalle entre l’écoute et l’état d’apprentissage : 15 secondes, ajustable entre 4 et 30 secondes.

* Quels inconvénients du Spanning Tree ?
* Le temps de redondance (forward delay timer) est de 20 secondes environ. En QoS, ce n’est pas terrible. -> Cisco a apporté un spanning tree portfast.

Solution : on a aussi inventé le **backbone fast** qui opère la bascule sur un port de secours dont la priorité est la plus grande sans recalculer. L’administrateur doit le définir. L’**uplinkfast** existe aussi (faire recherche). Mais ces solutions ne sont pas activées par défaut.

* Toutes les machines partagent la même connexion, ça ne pourra jamais dépasser le GigaBit.
* Pas de redondance sauf avec anneau (boucle)
* Avantages du Spanning Tree ?
* Simplicité
* Economie de coûts sur les liens
* Pas de boucles
* Performance : sur 4 switch, on perds en bande passante.

## L’architecture en étoile

La réponse à l’architecture en spanning tree est l’architecture en étoile.

L’avantage principal est la performance mais aussi l’extensibilité du réseau. On ajoute un pc, il a 100% de la bande passante.

Les inconvénients principaux sont le SPOF (single point of failure) dû à la dépendance au switch. Le prix est plus élevé, et plus de complexité.

* Trouver quelles architectures peuvent se débarasser du SPOF en gardant la redondance.