同時セッション数制御機構における認証機能の開発と評価

18G470 柴原 涼 （最所研究室）

特定サービスに対して，同時セッション数を制御するシステムの認証サーバの実装及び評価について述べる．さらに，Soft Limitation と予約システムの実装及び評価についても述べる．

1 はじめに

Web サーバにアクセスが集中するとサーバの応答性は低下してしまう．特にインタラクティブな Web アプリケーションではただ利用できるだけなく安定した応答性が求められる．それに対応するために，ある特定のサービス (特定サービス) を安定的に提供することができるファイアウォールとユーザ毎の識別を行う機構を組み合わせた同時セッション数制御機構の開発を行っている．この機構により，特定サービスサーバへの同時セッション数を制限することができるだけなく，ファイアウォールにより許可したユーザ以外からのアクセスを防ぐことで DoS 攻撃に対応することができる．本稿では，同時セッション数制御機構の認証機能の開発及び評価について述べる．

2 同時セッション数制御機構

同時セッション数制御機構の構成を図 1 に示す．本機構は，ユーザ認証を行う認証サーバ (Auth サーバ)，IP アドレスによるフィルタリングを行うファイアウォールである IP フィルタリングサーバ (IPF サーバ)，ユーザを識別し許可されてないユーザからのアクセスを拒否するユーザ識別サーバ (UI サーバ) で構成されている．実際にサービスを提供している特定サービスサーバ (SS サーバ) にアクセスするためには，IPF サーバと UI サーバを必ず経由する．IPF サーバでユーザの IP アドレスが許可されているか確認し，許可されているならば通過させる．次に UI サーバでは，認証されたユーザからのアクセスであるか確認し，認証されている場合は SS サーバにアクセスさせ，認証されてない場合は認証を促す Web ページにアクセスさせる．SS サーバへのアクセスは以下の手順で行う．ユーザは SS サーバのアクセスに先立ち Auth サーバで認証を行う．Auth サーバは認証が通れば，現在の同時セッション数と上限を比較し，認証されたユーザからのアクセスの場合，SS サーバにアクセスし，その結果をユーザに返す．認証されていないユーザからのアクセスの場合，認証を促すページにアクセスさせる．同時に UI サーバに許可されたユーザの情報を通知する．これ以降，ユーザは SS サーバへ IPF サーバと UI サーバ経由でアクセスできる．

3 認証サーバ

3.1 設計

Auth サーバは，ユーザ認証の機能を提供しているサーバであり，ユーザ情報とセッション情報のデータベースを保持している．この時，Auth サーバに対して，「DoS 攻撃からの Auth サーバの保護」と「悪意のある攻撃からのデータベース保護」というセキュリティ問題が考えられる．1 つ目の問題では，複数台の Auth サーバを配置することが考えられるが，Auth サーバ同士で情報を同期する問題が発生する．これに対応するために，Auth サーバをファイアウォール内に配置し，Auth サーバへのリバースプロキシサーバを配置する案を考えた．これにより，リバースプロキシサーバで Auth サーバへのアクセスを制限して DoS 攻撃などから Auth サーバを保護することができる上，悪意のある攻撃からデータベースを保護することができる．

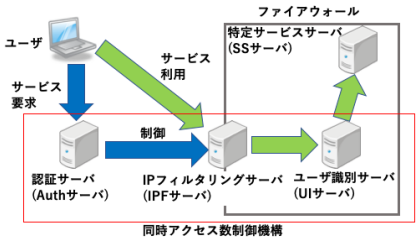

図 1: 同時セッション数制御機構の構成図

3.2 サーバ構成と認証手順

実装した同時セッション数制御機構のサーバ構成と認証手順を図 2 に示す．認証の手順は以下に示す通りである．ユーザは，いずれかのリバースプロキシサーバにアクセスする (1)(2)．この時，上限を超えている場合は上限を超えていることをユーザに通知する．そうでなければ，Auth サーバでログイン処理を行う (3)．Auth サーバは，セッション DB を参照して現在の同時セッション数が設定した上限に達しているか確認する (4)．上限に達していない場合，Auth サーバはセッション DB にセッション情報を登録し (5)，リバースプロキシサーバに許可がでたことを通知する．リバースプロキシサーバは 2 秒待機し，SS サーバの URL をユーザに通知する (8)．IPF サーバは，1 秒毎に Auth サーバで許可されたユーザ情報を問い合わせフィルタリングルールを更新し (6)，同様に Auth サーバも 1 秒毎にセッション情報を UI サーバに通知している (7)．フィルタリングルールの更新とセッション情報の通知をアクセスごとに行うことは，処理の待ち時間が多く発生してしまう．そのため，1 秒ごとにまとめで処理することで処理回数を減らしている．フィルタリングルールの更新とセッション情報の通知の完了が確実に終了することを待つために，リバースプロキシサーバで 2 秒待機している．

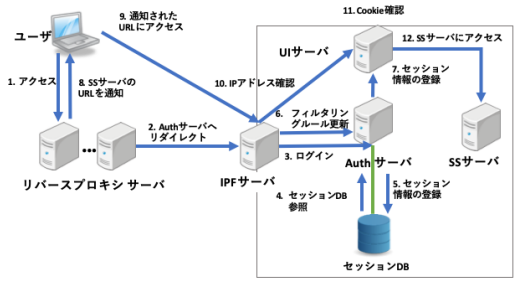

図 2: サーバ構成と認証手順

4 性能評価実験

秒間リクエスト数を 10 秒おきに 1 増やし 1 から 100 まで変化させて実験を行った．Auth サーバの WorkerProcess 数は 32 に，同時セッション数の上限は上限に達しない十分な数を設定している．実験開始から 450 秒 (秒間リクエスト数は 45) までのレスポンスタイムを図 3 に示す．結果から WorkerProcess 数より多い秒間リクエスト数でも応答時間が 1 秒以下であることから十分な応答性があると判断した．

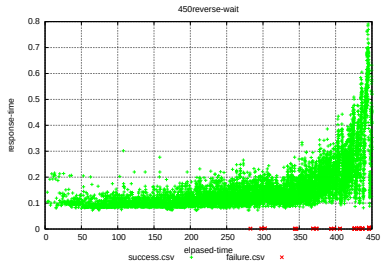

図 3: 450 秒までのレスポンスタイム

5 Hard Limitation と Soft Limitation

現在の同時セッション数が上限付近である時，ユーザ認証に成功するが SS サーバへのアクセスが拒否される問題が発生した．先着順のサービスであれば拒否するべきであるが，応答性を維持することが目的である場合は，多少上限を超えても応答性には大きくは影響しないと考えられ，Auth サーバにログインできたユーザは，SS サーバへのアクセスを許可した方が良いと考えた．そこで，指定した数まで超えていても許可する仕組み (Soft Limitation) を導入することにした．従来の方式を Hard Limitation とここでは呼ぶ．上限を 200 をとして秒間リクエスト 4 で 6 分間実験した．Hard Limitation の結果を図 4 に，上限を超えてアクセスできる人数を上限の 1 割 (20) とした Soft Limitation の結果を図 5 に示す．グラフの緑が成功したレスポンス，赤が失敗したレスポンス，青がリバースプロキシサーバでアクセスを拒否されたレスポンスを示している．Hard Limitation では，Auth サーバにアクセスできユーザ認証に成功したが SS サーバへのアクセスが許可されなかったことが確認できた．Soft Limitation では，失敗アクセスはなく上限を超えていても，若干応答時間が伸びてはいるがアクセスが許可されていることが確認できた．秒間リクエストを 7.5 倍の 30 に変更して行った実験の結果を図 6 に示す．初期アクセスに失敗し，さらに，Soft Limitation の上限である 220 を超えた 249 の許可を出していた．

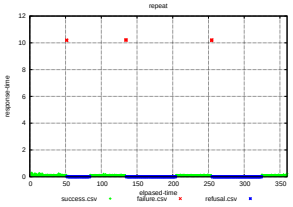

図 4: Hard Limitation のレスポンスタイム

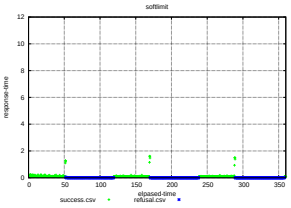

図 5: Soft Limitation のレスポンスタイム

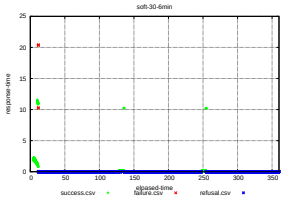

図 6: Soft Limitation のレスポンスタイム

6 予約システム

本機構では，サービスを利用したい時に利用できない状態が継続する事態が発生する．そこで，予約機能を導入することにした．予約処理のタイミングチャートを図 7 に示す．ユーザは利用したい時間を指定して予約を行う (reserve created)．予約した時間を過ぎてもサービスを利用しない可能性があるため，自動的にキャンセル時間を設定する．予約した時間 (reserve start) から自動キャンセルの時間 (reverse end) までの期間を予約期間と呼び，この期間内にアクセスすることで，上限に達していてもアクセスが許可される．予約時間よりも前にアクセスした場合は，可能であればアクセスが許可され，予約は取り消される．しかし，同じ時間帯に多くの予約が重なる場合，それら全てを予約することができない．そのため，指定数の予約までを受け付けるようにする．現在，予約期間内に到着したアクセスを受け付ける機能を実装し，その機能テストを行い，設計通りに動作することを確認した．

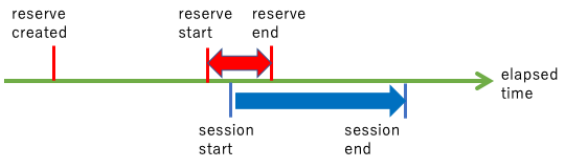

図 7: 予約処理のタイミングチャート

7 まとめ

本稿では，同時セッション数制御機構の認証機能の開発及び評価について述べた．評価によって，本機構の認証機能に十分な応答性があると判断した．さらに，Soft Limitation の提案，実装及び評価についても述べた．また，予約システムを提案し，一部の機能を実装した．今後は，予約機能とキャンセル機能について実装する必要がある．