目次

第 1 部

SQL インジェクション超入門

NaruseJun　著

第 1 章

初めに

本書では、SQL に触れたことがない方でも SQL インジェクションについて理解できるように、SQL とは何かというところから、実際に SQL を用いたアプリを組みながら SQL インジェクションの仕組みについて見ていきます。基本的な SQL インジェクション脆弱性から、それを応用した高度な攻撃方法の紹介とその対策方法についても紹介します。

本書の目的は SQL インジェクションについての理解ですので、攻撃とは関係のない SQL や RDBMS の機能については紹介しません。また、紹介する Web アプリは攻撃の解説のために**脆弱な (≒間違った) コード**を含みますのでご留意ください。

本書で紹介する手法を用いて、実際に稼働しているサービスに対して攻撃を試みると罪に問われる可能性があります。後ほど紹介する演習環境や、CTF など攻撃が許された場所以外では絶対に行わないでください。

本書を読むにあたって、SQL についての知識は必要ありませんが、

- 基礎的な PHP コード
- 基礎的な HTML コード
- HTTP リクエストによってどのようにデータが渡されるか

が理解できていることを前提としています。

1.1 「PHP がわからないのですが」

PHP は、Java や C に似た文法を持ちます。

PHP がこれらの言語と異なるのは、

- PHP コードは`<?php ... ?>`で囲まれ、HTML 中に埋め込まれること
- 変数名の頭には必ず`$`がつくこと (例:`$var`)

くらいなので、Java か C の知識があれば PHP コードの意味を理解することはそれほど難しくありません。

1.2 「HTML がわからないのですが」

本書で重要なのは、HTTP でサーバにデータを送るために用いる`<form>`と、その部品である`<input>`のみです。

`<form method="POST">`は、POST メソッドによってデータを送信することを意味しており、PHP では連想配列`$_POST` を使ってこの値を受け取ります。

`<input name="hoge">`は、画面上に入力欄を表示することを意味し、このとき送信されるパラメータ名は *hoge* です。PHP から受け取る際は、`$_POST["hoge"]` としてアクセスします。

第 2 章

基礎知識

2.1 SQL

SQL とは、**リレーショナルデータベースマネジメントシステム**（RDBMS）上でデータの定義、書き換え、問い合わせなどを行う際に用いる言語です。いわゆるプログラミング言語とはちょっと違っていて、この SQL だけで何かを作ることは（できなくはないのですが）難しいです。

→ コラム： SQL はチューリング完全？

この SQL はしばしば、他のプログラミング言語と RDBMS 間の橋渡し的役割を担います。プログラミング言語からデータベース上のデータにアクセスする手段として、SQL は用いられます。もちろん、人間が直接データベースに手を加える際に用いることもあります。

RDBMS

リレーショナルデータベースマネジメントシステムとは、関係モデルに基づくデータベースを管理するシステムです。ざっくりといえば「データやデータ間の関係を表として保持しておく」システムです。RDBMS を用いることで、データの検索が容易になります。

広く用いられている RDBMS としては、

- Microsoft SQL Server
- Oracle Database
- MariaDB(MySQL)
- PostgreSQL
- SQLite

等が挙げられます。

2.2 SQL インジェクション

SQL インジェクションとは、プログラムの不備を突いて意図していない SQL 文を実行する攻撃方法です。また、こうした攻撃を許すような脆弱性のことをいいます。

第 2 部

ドキドキ GLSL ★ Shader 工房

そばや　著

第 1 章

そもそもの話

そもそもシェーダとは何のためにあるのか？ といえば、実はこの記事でやるような綺麗な画像を生成するものじゃありません。シェーダとは、3D の CG プログラミングをするとき、ポリゴンの表面をちょっとかっこよくするためのものです。普通ならのっぺりとしたポリゴンに光沢をつけてみたり、鏡っぽくして反射させてみたり、良くてももともとある画像を加工するというくらいのもので、ここで紹介するように画面全体のすべてのピクセルの色をシェーダだけで決めるものでは断じてありません。が、これはこれで面白いのでまぁいいです。

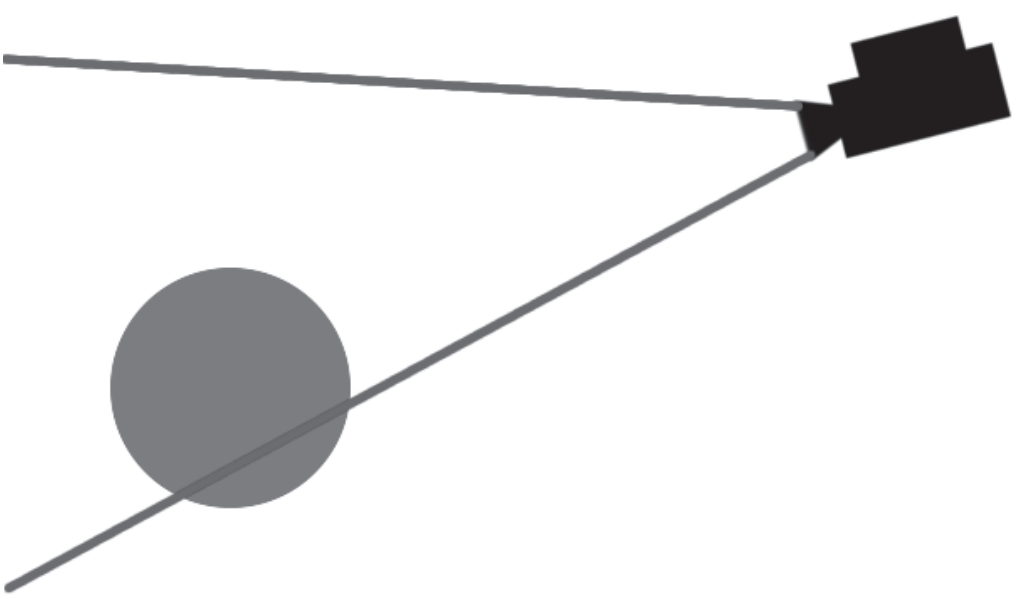
さて長い前置きはこのくらいにしまして、レイマーチングとは何かを説明していきます。

第 2 章

レイトレーシングの話

レイマーチングとは、レイトレーシングと呼ばれる 3DCG の技法の 1 つです。レイトレーシング法とは、カメラから**レイ**（光線）を仮想的に飛ばし、光線にあたったものの色をディスプレイに塗るという技法です。

例えば 3D 空間上に、カメラがあったとします。



カメラは空間のどこかを向いており、カメラに写った 3D 空間がディスプレイに映しだされます。

第 3 部

セールスマンが巡回したがってるんだ。

nari　著

第 1 章

セールスマンが巡回したがってるんだ。

こんにちは。nari です。今回は巡回セールスマン問題 (TSP) について紹介します。

1.1 コンピュータの限界

突然ですが、コンピュータにはどう頑張っても解けない問題があるのはご存知ですか？ 有名なのはプログラムの停止性問題です。これはプログラムが無限ループにハマるなどして停止しなくなるかどうかを判定する、というプログラムが存在するか？ という問題です。停止性問題については「存在する」と仮定するとどのようにしても矛盾してしまうため、そのようなプログラムは存在しません。つまり、「プログラムが停止するか？」という問題については無限時間かけても解けません。

では今度は、無限時間では無いものの、非常に長い時間かかってしまう問題はどうでしょう。例えば今世界で生存している全ての人間の髪の毛の本数を数える、なんてのはどうでしょうか。途方もない時間な気がしますが、現在世界人口は 70 億人 ($= 7 \times 10^9$) 、一人あたりおよそ 10 万本 ($= 1 \times 10^5$) の髪の毛が生えていると言われていますから、大体 700 兆本 ($= 7 \times 10^{14}$) ぐらいになると考えられます。確かに人間であれば絶対に一生の内で 700 兆も数えることは出来ないとは思いますが、スーパーコンピュータの中には 1 秒間に 1000 兆回 ($= 1 \times 10^{15}$) の演算が行えるものもあり、この程度であれば現実的な時間だといえます。

ということは、スーパーコンピュータさえあればどんな問題も一瞬で計算できるんでしょうか？ 答えは NO、と言われています。(少し曖昧なのは、もしかしたら今後の技術の発展でどんな (解けない問題でない) 問題も一瞬で計算できるコンピュータが登場するかもしれないし、問題の解決の糸口がつかめるかもしれないからです。) さて、ではどんな問題がこれにあたるのでしょうか。これは NP 問題と呼ばれるクラスに所属した問題が挙げられます。

1.2 計算量の表記

ここで軽く計算量の表記について解説します。髪の毛を数える問題の計算量を考えます。人間の数を n 、一人あたりの髪の毛の本数を m とすれば、計算回数は nm です。この時、計算に必要なプロセスの数が nm の値に比例する場合、 $O(nm)$ と表記します。これをランダウの記号と

第 4 部

普通の大学生が【LLVM】やってみた

long_long_float 著

第 1 章

普通の大学生が【LLVM】やってみた

以前から LLVM に興味があり、機会[*1]が出来たので LLVM をやってみた (というより使ってみた) 記録です。

自分はコンパイラ・言語処理系に関してはほぼ素人でありここに書いてあることが間違っているかもしれないのでツッコんでいただけるとありがたいです。

また、他の IR(中間言語) を知らないので言語仕様的に.NET や JVM との比較はせずあくまで自分が LLVM を見た時に感じたことを書いていきます。

1.1 対象読者

- 言語処理系の基礎は知っている
  - プログラムがどうコンパイル、実行されるか
- アセンブリ言語の基礎を知っている
  - 例えば for 文がどうやってアセンブリ言語で実現されているか
- C++ が読める
  - そんなに高度な機能は使っていない気がする (筆者自身 C++ ﾁｮｯﾄﾃﾞｷﾙ程度) ので知らなくてもなんとかなると思います

[*1] http://connpass.com/event/38487/

第 2 章

LLVM とは

公式ページを見る限りだと「モジュール化されており再利用可能なコンパイラ、及びツールチェイン技術」と言ったところでしょうか。これだとざっくりしているので具体的に書くと

- C, C++, Objective-C 等のフロントエンド
- LLVM 命令の実装 (アセンブリとバイトコードのリーダ、ライタ、ベリファイア)
- コードジェネレータ (アセンブラ?)
- JIT コンパイラ
- LLVM コンポーネントを作成するための API

こんな感じでしょうか。これらのツールを使ってソースコードを以下のように実行形式に変換していきます (以下は Clang の例です)。

第 5 部

ディープラーニングへの第 0.5 歩

とーふとふ　著

第 1 章

ディープラーニングへの第 0.5 歩

1.1　この記事でわかること

- Windows10 で Tensorflow を始める方法
- サンプルプログラム Cifar10 を動かしてみる方法
- Cifar10 を改造して自分用の画像分類機を作る方法

1.2　この記事を書いた人

とーふとふ

大学一年生で機械学習は 11 月に入ってから初めて触りました。プログラミング自体も本格的に始めたのは大学に入ってからです。

1.3　この記事の目標

Tensorflow のチュートリアルの一つである cifar10 を改造して、自分で用意した画像セットで訓練し、画像分類機を作ることです。機械学習を始めるモチベーションとしてよくあるものが、画像を分類したい！ ということだと思います。そこで色々ググってみたのですが、なかなかこのチュートリアルを実行する第一歩から自分でモデル[*1]を構築する独り立ちまでの間を詳しく書いた記事があまりなかったので、ここに自分の奮闘をまとめます。少しでも皆さんが機械学習に挑戦するときの手助けになりたいと思っています。

[*1] 学習するプログラムの構造のようなもの

第 2 章

環境

今回は多くの人が使っているであろう Windows10 で、Bash on Ubuntu を用いて Tensorflow の環境を構築していきます。さらに、簡単のために CPU 版の Tensorflow をインストールします。もし、GPU 版をインストールしたい場合はこちらなどを参考にすると良いと思います。

- http://qiita.com/qooa/items/c516001c07a768c6b51b#4-tensorflow%E3%81%AE%E3%82%A4%E3%83%B3%E3%82%B9%E3%83%88%E3%83%BC%E3%83%AB
- http://ill-identified.hatenablog.com/entry/2016/08/11/204205

Bash on Ubuntu で GPU の利用は難しいようです。Linux をインストールしてください。

Tensorflow のインストールができれば Mac や Linux でも Tensorflow のコードは同じように動くので、参考になると思います。それでは早速はじめていきましょう

◆ Tensorflow のバージョンアップ

この記事を書いたのは 11 月の半ばなのですが、タイミングがいいのか悪いのか Tensorflow のバージョンアップが来ました (r0.11 -> r0.12)。今回変更するコードには支障はありませんが、ダウンロードする URL などは適宜読み替えてください！！ しかも Windows で Tensorflow がサポートされたらしいですね！！！ なんと間の悪い！！！

2.1 Bash on Ubuntu の設定

Windows10 では Bash on Ubuntu という機能を利用することで、Windows 上で Ubuntu のコンソールを実行することができます。起動するために多少設定が必要です。スタートボタンを右クリックして、プログラムと機能をクリックします。