Vol.2015-DPS-162 No.4
Vol.2015-CSEC-68 No.4
2015/3/5

公衆回線を利用した リモート装置の同時複数操作手法の提案

本間将紘[†1]　金井敦[†1]　川上公一郎[†2]　中尾貢[†2]

近年業務の効率化を図るために様々な分野に IT が取り入れられるようになった。多数のビルの状態等を管理するシステムにおいてはクライアントに設置された装置の保守や診断に ICT が取り入れられている。しかし、安価なシステムでは廉価な公衆回線を使わざる終えないため双方向通信が実現できず、システム障害発生の際に作業員が実際に現場に出向き対応しなければならない。専用の回線を用いてクライアントシステムを新たに開発すれば問題は解決できるが、廉価にはシステムを構築できない。そこで、本研究では、廉価な公衆回線を利用し業務の効率化を図るため作業員が現場に出向かなくても集中管理システムからの操作で問題を解決でき、さらに IRC(Internet Relay Chat)のコンセプトを利用しグループ通信を効率的に可能とする廉価に構築できるアーキテクチャを提案する。さらに、プロトタイプを実装し性能評価を行い実用性があることを示す。

Simultaneous multiple operation method of remote terminal using public line

MASAHIRO HOMMA[†1]　ATUSHI KANAI[†1]
KOICHIRO KAWAKAMI[†2]　MITSUGU NAKAO[†2]

Recently, information technology has been exploited in various fields to improve the efficiency of operations. However, in the case of building information gathering system, Maintenance personnel has to go to the building when system trouble occurs because of using a low cost consumer public communication line. It is possible to solve the above problems by using an expensive dedicated line to communicate between control server and clients (building monitoring equipment). However it costs too much. In this paper, worker don't need to go to site because we solve the problem from remote location. We provide the system at a relatively low cost. In this paper, we don't change the client's network to cut the cost. We propose system architecture to realize two way communication between control server and clients and group communication for maintenance using IRC (Internet Relay Chat) concept and low cost consumer public line, furthermore we evaluate the architecture by implementing the prototype system.

1. はじめに

近年の IT の進歩には目覚ましいものがあり、様々な技術が登場、向上していく一方で、価格は急速に低下している。IT の進歩にともない業務の効率化を図り様々な分野に IT が取り入れられるようになった。

多数のビルの状態等を管理するシステムにおいてはクライアントに設置された装置の保守や診断に ICT が取り入れられている。以降ではクライアントに設置された装置をリモート装置と呼ぶ。通常、リモート装置にトラブルが発生した場合は、クライアントからベンダー事業者に連絡が入り、作業員が現場に出向きトラブルの解決を行う。しかし、上記の業務フローでは、作業員が現場に出向いてもトラブルの発生原因がすぐにはわからず、トラブルの解決に現地で多大な時間を消費する場合や、現場に着いても装置が稼働していて、作業は装置を停止させる事の出来る深夜にしか出来ない場合もある。また、装置の電源が入っていないなど、ごく単純な原因で装置が動作しない事をトラブルだと勘違いして連絡を受けた場合は簡単な操作でトラブルを解決できるので現場に出向く時間自体が無駄であり、充分に効率化が図られているとは言えない。上記のような作業はインターネットを通してリモート装置の監視/操作を行うことが可能になれば現地に出向く時間と費用を削減する事が出来、業務の効率化を図る事ができる。

リモート装置の保守/診断の例として NTT ファシリティーズの低価格での監視・保守サービス提供のために開発された監視システムを紹介する。システムの概要図を図 1 に示す。

†1 法政大学大学院
Hosei University Graduate School
†2 (株)NTT ファシリティーズ研究開発本部
Research and Development Headquarters, NTT FACILITIES, INC.

Vol.2015-DPS-162 No.4
Vol.2015-CSEC-68 No.4
2015/3/5

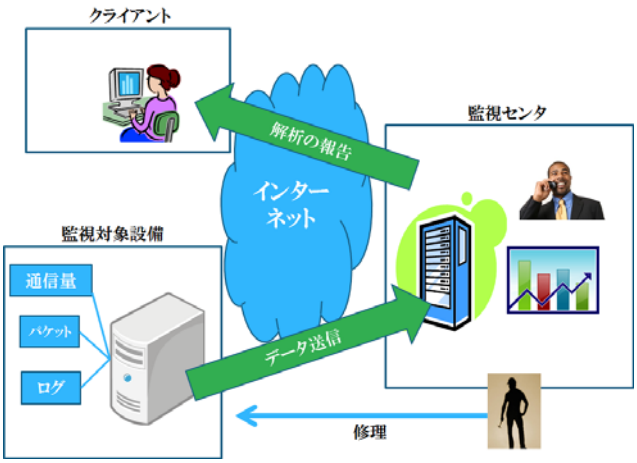

図 1 一般的なリモート端末の保守/診断

Figure 1 General of the remote terminal maintenance / diagnostic

このシステムではクライアントに「監視ユニット」と呼ばれる電源装置等を監視するリモート装置を設置し、電源装置にて障害を検知した場合、「センタシステム」と呼ばれる監視センタへ、アラームを随時送信することで、保守センタへの障害対応の駆けつけを手配する。このシステムでは、計測データを定期的に送信しているため現地でトラブルの原因解明に役立てる事が出来、時間の削減が出来る。しかし、監視ユニットのほとんどは、クライアントのローカルネットワーク上に設置されており、監視センタではデータを受信することは出来ても、監視ユニットへアクセスすることが出来ない。また、このシステムは監視センタへのデータの送信に HTTP(Hyper Text Transfer Protocol)を採用している。HTTP は、リクエスト/レスポンス方式でデータを送受信しているので、本システムにおける通信は、リモート装置からのアクセスを契機にしなければ通信を開始できない。また、レスポンスに関しても予め用意した内容を返すことしかできないので、リモート装置からのアクセスを利用した監視センタからの対話的な操作も行うことができない。よって、監視ユニットの操作を要するトラブルに直面した際は、最終的には作業員が現場に出向きトラブルを解決する必要がある。従って、NTT ファシリティーズのシステムのように低価格で提供されているシステムおいては、作業員が現場へ出向くための費用と時間を削減する事が重要な課題となっている。また、クライアントのネットワークに変更を加える手法やシステム用に専用線を引く手法を用いれば、監視センタ等の遠隔地からリモート装置を操作することが出来るので、作業員が直接現場に出向かずにトラブルを解決することが出来るが、初期構築時とランニングの費用が掛かるため導入されにくいという問題がある。そこで、本研究では作業員が現場へ出向く費用と時間の削減を図るために監視センタからもアクセスが可能であり、リモート装置を制御するためのコマンドを監視センタから送信し、リモート装置を操作することのできるシステムを低価格で提供できる手法を提案する。費用を削減するためにクライアントのネットワークに変更を加えず、公衆回線を利用する実装方法の調査と検討を行い、提案手法の有効な実装方法を示す。

このシステムを構築するためには次の 2 つの要素が必要である。1 つは、リモート装置と監視センタが対話的な通信を行うためのインフラである。もう 1 つは、監視センタからのコマンドをリモート装置で実行するためのインターフェースである。本稿では、インフラの構築について検討を行った。検討の際に参考にしたシステムは IRC(Internet Relay Chat)プロトコルを利用した BOT Net である。BOT Net とは BOT マスターからの指示を伝えるために BOT に感染した PC で構築されるネットワークのことである。BOT に感染した PC は特定の IRC サーバに接続し、特定のチャンネルに参加する。チャンネルに参加すると、PC は他の PC にユニキャスト、マルチキャストのどちらかでテキストベースのデータを送信することができるため、BOT マスターは同じチャンネルに参加し攻撃の指示をおこなう[1]。

IRC プロトコルとは、TCP/IP ネットワーク上でクライアントとクライアントがサーバを経由してテキストデータを交換して会話をするプロトコルである。典型的な構成としては、クライアントの接続点の中心にメッセージの配信、多重配信などの機能を持つ IRC サーバを設置し、メッセージの交換を行う。

以下本稿では、2 章では、リモート装置の同時複数操作システムについて述べる。3 章では、クライアントのネットワークに手を加えず、公衆回線を利用した提案手法の実装方法の調査と検討を述べる。4 章では、実装方法の優位性の評価を行う。5 章では、評価結果に対する考察を述べる。6 章では、本稿をまとめる。

2. 提案手法

現在リモート装置の保守/診断システムは、計測データやアラートなどはインターネットを通して通知するが、トラブルが発生した場合は作業員が直接現場に出向き作業を行う手法が主流である。中にはクライアントのネットワークに変更を加えたり、専用線を引いたりして遠隔地からトラブルを解決するシステムも存在するが、初期構築時とランニングの費用が掛かるため導入されにくいという問題がある。そこで、初期構築時とランニングの費用を抑えるために以下に示す 2 点を満たす手法を提案する。

(1) ネットワークに変更を加えず、公衆回線を使用する。
(2) サーバの構築に掛かる費用を抑える。

Vol.2015-DPS-162 No.4
Vol.2015-CSEC-68 No.4
2015/3/5

さらに、同種のリモート装置に対して定期的にメンテナンスを行う場合や同じトラブルに対する処理を 1 対 1 で全て行うのではなく、一斉に処理を行えるように、1 対 1 の通信だけでなく、グループ単位でのマルチキャストを可能にすることで業務の更なる効率化を図ることにする。

この章では、提案システムの概要について述べ、次章で、(1),(2)の条件を満たす 実装方法について述べる。

本システムでは、監視センタにデータ中継サーバとデータの送受信機能を持つコンピュータを設置し、クライアントに本システムでやり取りするデータの処理を行うインターフェースの付いた、テキストデータの送受信機能を持つリモート装置を設置する。以降ではそれぞれの装置を単に監視サーバ、管理端末、リモート端末と呼ぶ。

それぞれの装置が持つ機能について述べる。まず、監視サーバの持つ機能を以下に示す。

(1) テキストデータの配送、多重配送機能
(2) ID と PW によるクライアント認証機能
(3) SSL による中継データの盗聴・改竄防止機能
(4) SSL によるサーバ認証機能
(5) グループ管理機能

(1)の機能はリモート端末から送られる計測データ/アラートを管理端末に、管理端末から送られるコマンドをリモート端末へそれぞれ転送するための機能である。

(2),(3),(4)の機能は本システムが公衆回線を利用する事を前提にしているために必要な機能であり、(2)の機能は管理システムと関係ない端末の接続を防ぐための機能、(3)の機能はクライアント認証情報、計測データ、その他重要な情報を悪用されないようにするため機能、(4)の機能は第三者が設置した偽装サーバに接続する事を防止するための機能である。

(5)の機能は同種の機器、同じ地域にいるクライアントや同一クライアント内に複数のリモート端末を設置しているなど特定の条件下にある複数のリモート端末をグループで管理し、グループ単位でテキストデータを転送する機能である。

次に管理端末、リモート端末の持つ機能について以下に示す。

(1) テキストデータの送受信機能
(2) データ送受信用の GUI
(3) SSL による中継データの盗聴・改竄防止機能
(4) SSL によるサーバのなりすまし防止機能
(5) グループ管理機能

(1)の機能はリモート端末から送られる計測データ/アラート、管理端末から送られるコマンドを送受信するための機能である。

(2)の機能は人の手によりデータを送信するための機能と受信したデータを確認するための機能である。

(3),(4)の機能は監視サーバの(3),(4)の機能と同様の機能である。

(5)の機能は管理端末のみが持ち、リモート端末は持たない機能で、グループの追加、作成、削除が可能である。

本システムの構成図を図 2 に示し、1 台のリモート端末を監視する場合の稼動フローを以下に示す。

(1) リモート端末起動、監視サーバに接続
(2) 通常稼動
(3) トラブル発生時
(4) 通常稼動

(1)リモート端末は起動すると自動で監視サーバに接続を開始する。この時 SSL を使いサーバの正当性を確認してから暗号化通信を利用して ID と PW を送信し、クライアント認証を行う。クライアント認証後の通信は全て SSL を利用して暗号化して行う。

(2)クライアント認証を済ませた後は、リモート端末としての本来の動作を行い、必要に応じて計測データなど必要な情報を管理端末へ定期的に送信する。また、この時管理端末からのコマンドも受信可能になっているので、コマンドが送られてきた場合はコマンドに従った動作を行う。

(3)トラブルが発生した場合はリモート端末からアラートを送信し、管理端末に異常が発生した事を通知する。アラートを受けた管理端末はコマンドを送信してトラブルの解決を行う。

(4)トラブルが解決したら(2)の通常稼動に戻る。

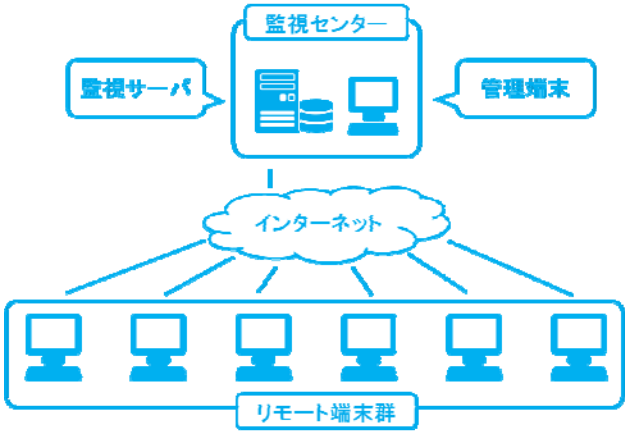

図 2 提案手法の概要図
Figure 2 Schematic diagram of the proposed method

次にグループを作成して、複数のリモート端末を監視する場合の稼動フローを以下に示す。

監視システムに参加しているクライアントの事業内容と事業所を表 1 に示す。事業所の地域を東北(T)、関東(K)、北海道(H)で表し、事業内容を金融系(F)、環境系(E)、福祉系(W)で表す。各企業の事業所には事業内容ごとのリモート端末がそれぞれ 1 台ずつ設置されているものとする。

表 1　各企業の事業所と事業内容

Table 1　Office and business of each company

企業名	A	B	C	D
事業所	T	K	T	T
	K	H	K	H
事業内容	F	F	E	F
	E	E	W	W
	W	―	―	―

表 1 のクライアントをグループで管理する場合のグループ例を企業別、地域別、事業内容別に表 2、表 3、表 4 に示す。

表 2　企業別のグループ例

Table 2　Examples of groups of each company

グループ	A	B	C	D
企業名、事業所、事業内容	A,T,F	B,KF	C,T,E	D,T,F
	A,T,E	B,K,E	C,T,W	D,T,W
	A,T,W	B,H,F	C,K,E	D,H,F
	A,K,F	B,H,E	C,K,W	D,H,W
	A,K,E	―	―	―
	A,K,W	―	―	―

表 3　地域別のグループ例

Table 3　Examples of groups of each region

グループ	H	K	T
企業名、事業所、事業内容	B,H,F	A,K,F	A,T,F
	B,H,E	A,K,E	A,T,E
	D,H,F	A,K,W	A,T,W
	D,H,W	B,K,F	C,T,E
	―	B,KE	C,T,W
	―	C,K,E	D,T,F
	―	C,KW	D,T,W

表 4　事業内容別のグループ

Table 4　Examples of groups of each business

グループ	F	E	W
企業名、事業所、事業内容	A,T,F	A,T,E	A,T,W
	A,K,F	A,K,E	A,K,W
	B,K,F	B,K,E	C,T,W
	B,H,F	B,H,E	C,K,W
	D,T,F	C,T,E	D,T,W
	D,H,F	C,K,E	D,H,W
	―	―	―

(1)　全てのリモート端末起動、監視サーバに接続
(2)　管理端末がグループを作成
(3)　通常稼動
(4)　金融系のリモート端末でトラブル発生
(5)　通常起動

(1)1 台のリモート端末を管理する場合と同様に SSL を使って全てのリモート端末の認証を行う。

(2)管理端末から表 2、表 3、表 4、に示すように企業別、事業所別、事業内容別のグループを作成し、ユーザの追加を行う。1 つのリモート端末は複数のグループに重複して参加させることが出来る。グループの作成、追加、削除は管理端末からのみ行える。また、管理端末からの意思のみでリモート端末の同意を必要としない。また、グループの作成、追加、削除は動的に行え、(3)の通常起動に状態が移った後にも行える。

(3)1 台のリモート端末を管理する場合と同様に計測データなど必要なデータを管理端末にのみ送信する。この時同じグループに所属しているリモート端末には計測データ等は中継されない。

(4)金融系のリモート端末に 1 台にトラブルが発生した場合、リモート端末からアラートがリモート端末に送信される。アラートを受けた管理端末はアラートを送信してきたリモート端末だけでなく同じグループに対して診断/操作を行う。全ての操作が完了したら(2)の通常稼動に戻る。

3. 実装方法

提案方式ではクライアントのネットワークに変更を加えず、公衆回線を利用することを前提としている。提案方式に必要な機能のうち上記の前提条件を守って実装する場合問題となる機能はリモート端末と監視サーバの双方向通信である。ほとんどの企業で外部との通信が許可されているプロトコルの中でテキストデータの送受信に向いているプロトコルとして HTTP や SMTP(Simple Mail Transfer Protocol)と POP(Post Office Protocol)のメールプロトコルが挙げられる。メールプロトコルの送信は個々の好きなタイ

ミングで送信可能であるが、受信は受信サーバにポーリングする事でメールが存在する場合ダウンロードを開始するので、リアルタイムの双方向通信としては使えない。HTTP は通常リクエスト/レスポンス方式でデータを送受信行っているためリアルタイム双方向通信に向いていない。しかし、Ajax(Asynchronous Javascript + XML)や Comet による実装方法をとれば、HTTP を利用して擬似的な双方向通信が可能となる。Ajax とは、リクエストした HTML のなかで Javascript のクラスである XMLHttpRequest[2]を利用して非同期に polling することで擬似的な双方向通信を可能にする pull 型の実装方法である。Comet とは、リクエストした HTML の中で XMLHttpRequest を利用して非同期に Long polling することで、サーバのタイミングで返答を返す擬似的な双方向通信を可能にする push 型の実装方法である。また、ポート 80 番を使うプロトコルに WebSocket というプロトコルがあり、WebSocket を利用することでリアルタイム双方向通信が可能となる。WebSocket とは HTML5 の仕様の一部として策定されていた Web サーバとクライアントの双方向通信用の規格であり、HTTP コネクションより軽量である[3]。サーバとクライアントが 1 度コネクションを確立すると明示的に切断しない限り持続的接続があり、どちら側からでも通信を開始できる。また、提案手法はサーバの構築に掛かる費用を抑える事を前提としている。サーバの構築費用を抑えるためには CPU の使用量を抑える実装方法をとる必要がある。上記の 3 つの手法についてサーバの構築費の指標としてリソースの使用量、双方向通信の性能の指標としてトラフィック量、リアルタイム性について比較を行う。リソースの使用量、トラフィック量、リアルタイム性について比較した表を表 5 に示す。表 5 より提案方式の実装は WebSocket を利用して実装することにする。

表 5　評価指標のまとめ
Table 5　Summary of evaluation index

実装方法	リソース	トラフィック	リアルタイム
Ajax	○	×	△
Comet	×	△	○
WebSocket	△	○	○

3.1.1 リソースの使用量

リソースの使用量は実装方法が双方向通信する際に占有する CPU 使用率のこととする。

Ajax はポーリングの際にコネクションを確立する時のみリソースを占有する。Comet はサーバでイベントが発生するまで polling の保留状態を維持する実装方法なので双方向通信を行っている間はリソースを占有する。Ajax と Comet がリソースを占有する場合は HTTP コネクションに必要な分だけリソースを占有する。WebSocket は双方向通信を行っている間 WebSocket に必要な分だけリソースを占有する。

3.1.2 トラフィック量

トラフィック量は実装方法が双方向通信する際にネットワークに流れるデータの量のこととする。

Ajax は polling の間隔によるが、polling のたびに HTTP のリクエストを送信する。Comet は Long polling の時リクエストとサーバでイベントが発生した時のみリクエストが返ってくる。Ajax と Comet で発生するトラフィックには HTTP ヘッダが付いている。WebSocket はそれぞれの通信の時のみトラフィックが発生する。

3.1.3 リアルタイム性

リアルタイム性は双方向通信をする際のサーバとクライアントの間の通信速度のこととする。

Ajax は pull 型の実装方法なので polling の間隔によってリアルタイム性が変化する。polling の間隔が短くなればリアルタイム性が上がるが、トラフィック量が増える。トラフィック量とリアルタイム性はトレードオフの関係にある。Comet はサーバからのレスポンスはサーバのイベントが発生したタイミングなのでリアルタイム性が高い。しかし、Long polling の時のリクエストを送っている間にサーバでイベントが発生するとラグが発生してしまう。WebSocket は双方向通信を目的に作られた規格なのでリアルタイム性が高い。

4. 検証

この章では実装方法で述べた実装方法のうち特に優れていると考えられる WebSocket と Comet のリソースの使用量、トラフィック量とリアルタイム性を簡易的な双方向通信システムを実装し、比較した。検証内容は以下に示す通りである。

(1) 100~1000 のクライアントをサーバに接続した時にサーバで使用される CPU の使用率を計測した。
(2) 100~1000 のクライアントから同時に通信を行った時にサーバで使用される CPU の使用率を計測した。
(3) 100~1000 のクライアントから同時に通信を行った時にサーバから送られるトラフィック量とその通信速度を計測した。

Comet の実装を行った環境について表 6 に WebSocket の実装を行った環境について表 7 に示す。リソースの使用量についての検証結果を表 8 にトラフィック量、リアルタイ

ム性についての検証結果を表 9 に示す。

表 8、表 9 に示した結果からわかるとおり、Comet による実装より WebSocket による実装の方がリソースの使用量、トラフィック量、リアルタイム性の 3 点では優れていることを確認することが出来た。

5. 考察

サーバのリソースの使用量が少ないほうがサーバの設置にかかる初期投資を抑えることが出来るので、結果としてサービスを安価で提供できると考えられる。トラフィック量についてはサーバから送られる通信量なので、クライアントに届くトラフィック量は接続数分の通信量となる。クライアントに届くトラフィックの量が少ないほうがクライアントの他のシステムに与える影響が少ないという点でも優れていると言える。

検証結果で述べた通り 100~1000 台の端末では提案手法を実装する方法として WebSocket が優れていることが確認できた。接続する台数を増やしていっても WebSocket による実装方法の優位性は変わらないと考えられるので、WebSocket による実装はスケーラビリティが高いと言える。

表 6　Comet の実装環境

Table 6　Comet implementation environment

役割	OS	CPU	メモリ	実行環境
サーバ	CentOS6.5	Intel Core 2Duo E8500 3.16GHz	4GB	Apache + PHP
クライアント	Windows7	Intel Core 2Duo E8400 3.00GHz	4GB	JMeter

表 7　WebSocket の実装環境

Table 7　WebSocket implementation environment

役割	OS	CPU	メモリ	実行環境
サーバ	CentOS6.5	Intel Core 2Duo E8500 3.16GHz	4GB	Node.js + WebSocket
クライアント	Windows7	Intel Core 2Duo E8400 3.00GHz	4GB	Node.js + 自作ツール

表 8　CPU の使用率

Table 8　CPU utilization

接続数	Comet		WebSocket	
	待機時	通信時	待機時	通信時
	CPU [%]	CPU [%]	CPU [%]	CPU [%]
100	1	7	1	3
200	1	10	2	6
300	2	12	2	10
400	2	13	3	14
500	2	17	3	16
600	3	18	3	17
700	3	20	3	17
800	3	20	3	18
900	3	21	3	18
1000	3	21	3	19

表 9

Table 9　Communication time and traffic volume

接続数	Comet		WebSocket	
	通信量 [kb]	通信速度 [ms]	通信量 [kb]	通信速度 [ms]
100	49.3	10	15.7	16
200	164.2	16	25.9	16
300	215.7	17	37.5	16
400	334.6	19	62.7	16
500	456.2	32	78.2	16
600	608.3	35	96.2	16
700	807.5	38	109.9	16
800	1028.4	42	125.3	16
900	1156.2	50	140.9	16
1000	1363.6	91	163.2	16

6. おわりに

本研究ではリモート装置の保守/診断の業務を遠隔地からコマンドでリモート装置の操作を可能にするという点と 1 対 1 の操作だけではなくグループ単位によるマルチキャストでコマンドを送信できるようにするという点で効率化を図る手法を提案した。また、提案手法は WebSocket を利用して実装する方法が有効であることを示した。

参考文献

1)　荒谷光,本間将紘,金井敦,斉藤典明:IRC プロトコルを利用した攻撃者と感染端末の探索手法, コンピュータセキュリティシンポジウム 2013 論文集,2013 巻,4 号,pp.139-146(2013)
2)　van Kesteren, A., Ed.: XMLHttpRequest, W3C Candidate Recommendation CR-XMLHttpRequest-20100803(2010)
3)　I.Fette,A.Melnikov:The WebSocket Protocol,IETF https://tools.ietf.org/html/rfc6455