Vol.2022-IOT-56 No.52
2022/3/8

# IoTデバイス群を対象とした P2P型ネットワークブート機構の検討

森川 太斗[1,a)]　松原 克弥[1,b)]

**概要**：実世界の様々な情報をセンシングするために導入される IoT デバイスは，管理者の目がとどきにくい設置場所や設置数に比例した管理コスト増大などが原因となり，情報漏えいや不正利用などの攻撃に晒されるリスクが高い．特に，近年の高機能・高性能化した IoT デバイスは，PC と同様に，OS やライブラリなど多数のソフトウェアコンポーネントで構成されており，IoT デバイスの管理においても，脆弱性対策としてのソフトウェア更新管理が重要となっている．PC 管理で広く活用されているネットワークブートは，サーバからネットワークを通じて配信されるソフトウェア環境を使って OS やアプリケーションを起動することで，起動性能とソフトウェア環境管理コストの両方を最適化できるプロビジョニング手法である．しかし，IoT デバイスのような膨大な数のデバイス群を対象としたネットワークブートでは，サーバやサーバにつながるネットワークがボトルネックとなって，起動性能が大幅に低下することがある．本研究では，広域に分散した IoT デバイス群を対象としたネットワークブートの最適化手法として，P2P 形式による分散協調ソフトウェア配信機構を検討する．

**キーワード**：システム運用管理, ソフトウェア・プロビジョニング, IPFS

## 1. はじめに

近年，産業や農業などのさまざまな分野における Internet of Things（IoT）の導入が進んでいる．2020 年の時点で約 113 億台の IoT デバイスがインターネットに接続していることが確認されており，2025 年までに IoT デバイスの数は 270 億台を超えると予測されている [1]．また，Edge AI 技術などの登場により IoT 機器の高機能化が進み，比較的計算性能が高いハードウェア上で，PC と同規模の OS やライブラリなどの高度なソフトウェアコンポーネントが稼働する IoT デバイスも増えつつある．

IoT デバイスは，常にネットワークに繋がっているという性質上，サイバー攻撃の対象になる可能性が大きい．情報通信研究機構が運用するサイバー攻撃観測網（NICTER）が観測した，サイバー攻撃における攻撃対象の上位 10 位のうちの 6 つは，IoT 機器に関連した攻撃対象となっている [2]．実際，2016 年には Mirai と呼ばれるマルウェアが感染した IoT デバイスが，大規模な DDoS 攻撃に利用されるという事例も発生している [3]．

このような問題に対処するために，経済産業省と総務省のガイドラインでは，脆弱性に対処した対策版のソフトウェアを IoT デバイスへ配布・アップデートする手段が必要であると述べられている [4]．しかし，IoT デバイスの導入では，設置デバイス数の多さや，その設置範囲の広さが障害となることで，人海戦術による IoT デバイス毎のソフトウェア更新作業では，適切な頻度で対応できない場合がある．

多数のネットワーク接続デバイスを対象としたソフトウェア環境の導入・管理には，ネットワークを介したソフトウェアプロビジョニングの採用が適している．特に，OS を含むディスクイメージをネットワーク経由でサーバから取得しつつ OS 起動を行うネットワークブートは，ソフトウェア環境の集中管理が容易となることはもちろん，ストレージレスなハードウェア構成がもたらすデータ流出・デバイス流用対策としても有用性が高い．しかし，IoT デバイスを対象としたネットワークブートでは，デバイス数の膨大さによるサーバ負荷集中，広域配置された IoT デバイスとサーバ間のネットワークにおける帯域の不足や転送遅延の増加に対処する必要がある．

本研究では，広域に配置された IoT デバイス群を対象として，ネットワークブートにおける前述の課題に対処する手法として，Peer-to-peer（P2P）型のネットワークブー

[1] 公立はこだて未来大学
Future University Hakodate, Hakodate, Hokkaido, Japan
[a)] b1018043@fun.ac.jp
[b)] matsu@fun.ac.jp

ト機構を提案する．OS 起動に必要なディスクイメージをサーバからだけでなく，同じディスクイメージで起動した近隣の IoT デバイスからも取得できる機構を実現する．ディスクイメージ保持デバイスの把握と最適なイメージ取得先デバイスの選定のために，広域分散ファイルシステム実装のひとつである IPFS（Interplanetary Filesystem）[5] をディスクイメージ管理機構として採用する．さらに，起動した IoT デバイスの OS が管理するディスクキャッシュを起動元のディスクイメージの複製として再配布できるように，OS カーネル内に IPFS コンテンツ配信機能を組み込む．本提案機構は，多数の起動済み IoT デバイスがミラーサーバの機能を持つことでサーバ負荷集中に対処し，IPFS の仕組みにより最適な近隣デバイスからディスクイメージを取得することで，広域配置におけるネットワーク制限の課題にも対処できる．また，各デバイスが再配信に用いるディスクイメージ・データを OS がメモリ上に保持するディスクキャッシュから取得することで，ストレージレスなハードウェア構成を維持したデータ流出・デバイス流用対策も可能となる．

本稿では，提案する P2P 型ネットワークブート機構の有用性と実現手法についての検討を行う．また，OS ディスクキャッシュの再配信機能を含まない，IPFS をディスクイメージの配信元とするネットワークブート・クライアント機能のプロトタイプ実装について述べる．さらに，プロトタイプ実装を用いて，ネットワークブートにおける IPFS を介したディスクイメージ配信の影響を評価するための実験結果を示す．最後に，本提案のまとめと今後の課題について述べる．

## 2. IoT デバイス管理における課題

IoT デバイスは，PC とは異なる特性をもつため，ネットワークブートのように PC 管理で広く用いられる機構を採用する場合でも，さらに対処すべき課題が存在する．実際，十分な管理ができていない IoT デバイスが要因となって，社会的な問題に発展する可能性を指摘された事例もある [3]．

### 2.1 デバイス数

PC やタブレットに比べて，IoT デバイスは，ひとつのシステム内で導入・管理されるデバイスが複数であることが多く，数百や数千といった膨大な数のデバイスを管理すべきケースも存在する．これら膨大な数のデバイスの管理では，PC の集中管理で用いられる手法を採用した場合でも，データ管理やネットワークトラフィックなどにおいて PC を対象とする場合を超える負荷を想定する必要がある．

このような数の多い IoT デバイスに対しては管理が簡潔に行え，一括での管理が可能なネットワーク経由での集中管理が適している．ネットワーク経由でデバイスを集中管理する手法の中でも，遠隔のサーバからデバイスの起動に必要なディスクイメージなどのファイル群を取得し起動を行うネットワークブートでは，IoT デバイスの集中管理だけでなく，デバイスのストレージを利用しないためデバイスの高寿命化に貢献できる．一方で，IoT デバイスは台数が多く，起動時やソフトウェアアップデートにともなう再起動時にはディスクイメージを配布するサーバへの大規模なアクセスが発生する．その結果，サーバやその周辺のネットワークに負荷がかかり，起動にかかる時間が増加するブートストームと呼ばれる現象が発生するため，膨大な数の IoT デバイスを対象としたネットワークブートでは，ブートストームへの対処が必須となる．

### 2.2 設置範囲

IoT デバイスに対してネットワークブートを適用する場合には広域に分散している IoT デバイスへの対応が課題となる．様々な地域の気象情報をセンシングする気象センサーや遠隔地の監視などに利用されるネットワークカメラなどのデバイスは広域にわたって分散して配置される場合がある．ブートストームを緩和する手法として，ディスクイメージを配布するサーバを冗長化する既存手法が知られているが，広域にわたって管理対象デバイスが分散配置されている場合，多数のサーバを広範囲に分散させて設置することは，導入コストや管理オーバヘッドの観点から現実的でない．

これに加え，広域に分散したデバイスを対象とする場合にはデバイスによりサーバへのネットワーク的な距離が異なる場合がある．サーバとネットワーク的な距離が離れているデバイスでは起動時間の増加やディスクイメージ読み出しのレイテンシ増加といった問題が存在する．

### 2.3 セキュリティ

IoT デバイスは，小型で持ち運びも容易であり，常時監視が行われていない箇所へ設置されることもあるため，盗難や分解などの物理的な攻撃のリスクが大きい．特に，IoT デバイスがストレージを装備している場合には，デバイス内部に残っているデータの流出や，ストレージに保存されたソフトウェアを流用した IoT デバイスの不正利用の可能性を考える必要がある．これらのリスクに対処するため，IoT デバイスのストレージレス化や，ソフトウェアの脆弱性対策などの対策が求められる．

## 3. 提案システム

本研究では IoT デバイス群を対象としてネットワークブートを適用した際に生じるブートストームへの対策として P2P 通信を用いたディスクイメージの分散管理を行う．ディスクイメージをデバイス間で分散管理することでデバイスが一斉に起動した際にサーバだけでなく，同一のディ

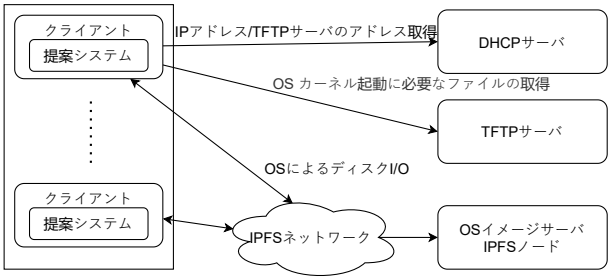


**図 1** 提案システムの全体図

スクイメージを利用する端末からもディスクイメージを取得することでサーバなどにかかる負荷を他の端末に分散させ，ブートストームを軽減する．

P2P の通信をネットワークブートに導入するために本研究では IPFS を利用する．IPFS は P2P 技術を用いた分散ファイルシステムであり，同一のコンテンツを複数の端末で保持できる．複数の端末から同一のコンテンツを取得できる場合には高速に応答する端末からコンテンツを取得する性質がある．その性質を利用することでサーバが遠隔地に存在しており，高速に応答する近隣の端末にディスクイメージが保持されている場合には，近隣の端末からディスクイメージを取得することができる．

IPFS ではコンテンツ取得後，ストレージにコンテンツが保持されリクエストがあった場合には再配布される．一方，2 章で述べたように端末のストレージにデータを保持する場合，データ流出や不正利用のリスクが存在する．そこで，本研究ではストレージではなく，OS が管理するメモリ上に存在するキャッシュを再配布することでストレージを利用せずに P2P によるディスクイメージの再配布を実現する．

提案システムの全体像を図 1 に示す．現状のネットワークブートでは TFTP サーバや HTTP サーバなどから OS 起動に必要なファイル群を取得したのちに NBD[6] サーバや NFS サーバなどからディスクイメージを取得する流れになっている．本研究では外部のサーバとの通信を IPFS ネットワークとの通信に置き換えることでディスクイメージを配布するサーバだけでなく，同一のディスクイメージで起動する他の端末からもディスクイメージを取得できるようにする．

提案システムの実現のために図 2 で示すように端末のストレージシステム層に対して IPFS ネットワークからのディスクイメージ取得機能，メモリ上のキャッシュの再配布機能を導入する．これによりネットワークブートを行う端末は IPFS からディスクイメージを取得し，メモリ上にディスクイメージがキャッシュされたあとは，他の端末に IPFS ネットワークを通じてディスクイメージを再配布することでディスクイメージを配布するサーバの負荷を削減できる．

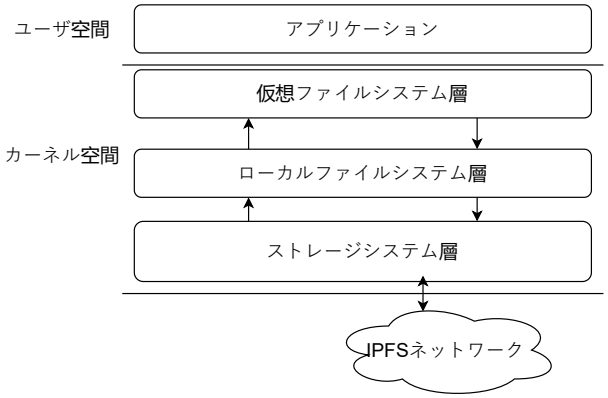


**図 2** 提案システムの改造箇所

IPFS を用いた提案システムと既存のネットワークブートのブートシーケンスの違いは OS 起動に必要なファイル取得後の流れである．提案システムのブートシーケンスを図 3 に示す．提案システムにおいても PXE を用いて DHCP サーバから IP アドレス，TFTP サーバのアドレスを取得し，TFTP サーバから OS カーネルの起動に必要なファイル群を取得するまでの流れは通常のネットワークブートと同様である．一方で，提案システムでは OS カーネル起動後に IPFS を利用するために，他の端末との接続情報を保持している Bootstrap ノードにアクセスする．その後，IPFS ネットワークに接続しているディスクイメージを配布するサーバを含む他の端末の情報を取得し，それらの端末からディスクイメージを取得し，それらの端末に対してキャッシュされたディスクイメージを再配布することでディスクイメージを分散管理する．

## 4. プロトタイプ実装

提案システムを実装する前に IPFS 上のディスクイメージをブロック単位で利用しネットワークブートが可能であるかを検証する必要がある．現状では IPFS 上のコンテンツを IPFS が管理するブロック単位で取得しネットワークブートを行った事例が存在しない．したがって，IPFS 上のディスクイメージをブロック単位で取得して，取得したブロック単位のディスクイメージでネットワークブートを行うプロトタイプを作成する．

実装したプロトタイプの概要は図 4 のようになる．プロトタイプの実装のためにネットワークブートにおいて OS によるディスク I/O で利用される NBD サーバに対して IPFS のコンテンツ取得機能を導入する．本研究ではプロトタイプ実装のために以下の 2 つを行った．

(1) NBD サーバへの IPFS 通信機能の導入
(2) ネットワークブート用ディスクイメージの作成

### 4.1 NBD サーバへの IPFS 通信機能の導入

本研究では改造の対象とする NBD サーバとして NBD

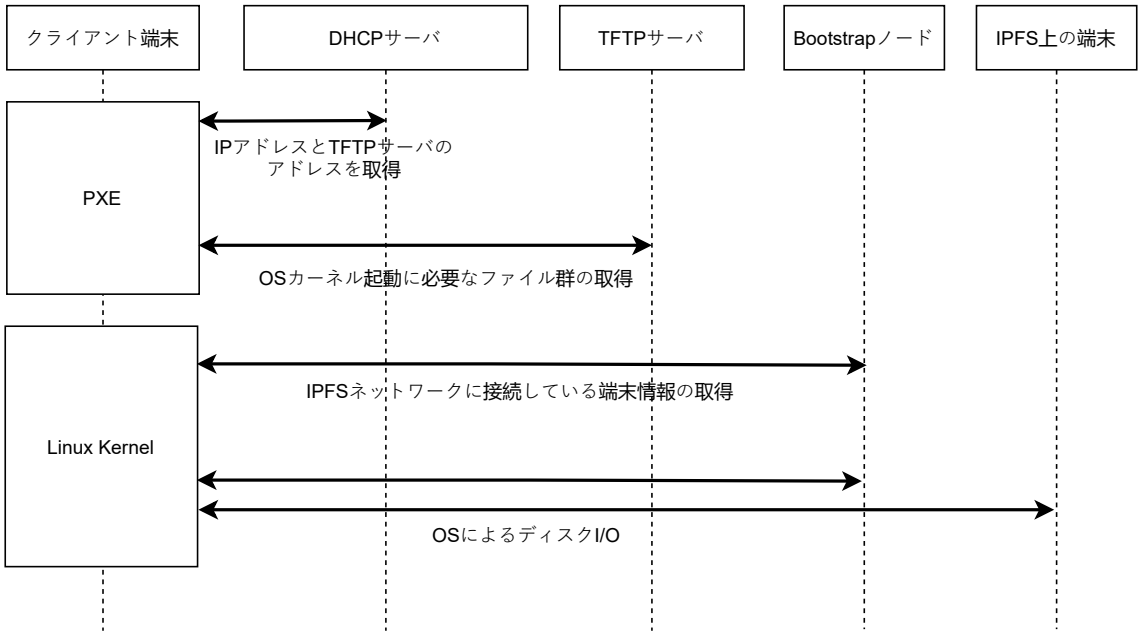


図 3　提案システムでのブートシーケンス

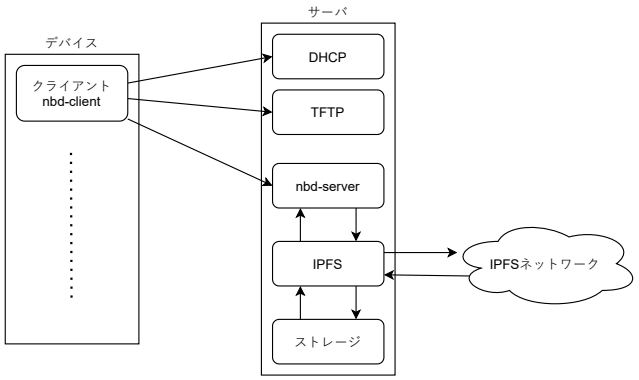


図 4　システム概要図

サーバの go 言語の実装である gonbdserver[7] を利用する．IPFS は go 言語と JavaScript での実装が存在しており，IPFS の処理を組み込むのが他の実装に比べて容易であるためである．

本実装では gonbdserver 内部にある nbd パッケージの Backend インターフェースに着目する．Backend インターフェースは gonbdserver におけるファイルの読み書きの処理を定義するためのインターフェースである．gonbdserver ではクライアントからリクエストがある場合には Backend インターフェースの実装の処理を呼び出してファイルの読み込み・書き込みを行っている．本実装では IPFS からコンテンツを取得できるように Backend インターフェースの実装を行っている．

### 4.2　ネットワークブート用ディスクイメージの作成

プロトタイプの実装では NBD サーバを利用しており，クライアントのディスクイメージでは NBD を用いたネットワークブートに対応させる必要がある．加えて単一のディスクイメージを複数の端末で同時利用できるようにするためにサーバが管理するディスクイメージへの直接的なファイル書き込みを回避する必要がある．それらの実現のためにディスクイメージの作成では以下の 2 つを実施した．

(1) initramFS の変更
(2) Overlayfs[8] の利用

起動時に NBD を利用するために initramFS を NBD を使ったブートに対応できるように変更した．

本実装では Overlayfs を利用しファイルへの書き込みを tmpfs[9] に行わせるために，overlayroot[10] と呼ばれるパッケージを利用した．これにより，ディスクイメージへの書き込みは tmpfs に対して実行され，サーバが管理するディスクイメージへの変更は行われない．

## 5.　実験

提案システム実現の前に IPFS の管理するブロック単位でのディスクイメージを用いたネットワークブートが可能であるか，どのようなオーバーヘッドが発生するかを評価する実験を行った．プロトタイプ実装を用いて IPFS 上のディスクイメージをブロック単位で取得しつつネットワークブートを行い，OS カーネル起動に必要なファイル群を取得し始めてから Ubuntu のログイン画面が表示されるまでの時間を計測した．実験は図 5 に示すような環境にて実施した．サーバとブート対象の PC のスペックを表 1 と表 2 に記載する．

表 1　実験環境におけるサーバのスペック

| | |
|---|---|
| CPU | Intel(R) Core(TM) i5-6260U CPU @ 1.80GHz |
| メモリ | 15.6GiB |
| OS | Ubuntu 18.04 LTS |

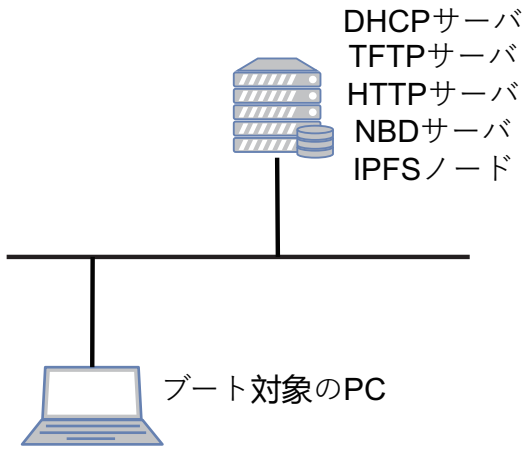


図 5　実験環境の概要図

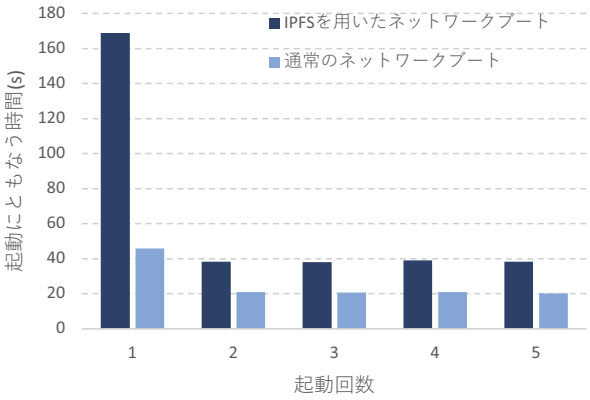


図 6　実験結果

表 2　実験環境における起動端末のスペック

| | |
|---|---|
| CPU | Intel(R) Core(TM) i3-4000M CPU @ 2.40GHz |
| メモリ | 7.7GiB |
| OS | Ubuntu 18.04.6 LTS |

これらの環境下でプロトタイプの実装と通常の gonbd-server を用いた状況でそれぞれ，5 回ネットワークブートを実施し，それにともなう起動時間を計測した．結果を図 6 に示す．初回起動時にはどちらの場合においても起動時間の遅延が見られた．2 回目以降の起動ではプロトタイプ実装を利用した場合，通常のネットワークブートに比べて 20 秒程度の遅延が発生した．

本実験では IPFS を通じてストレージからディスクイメージを取得し，ネットワークブートを行っている．2 回目以降の起動が初回の起動に比べて高速なのは OS が管理するキャッシュにディスクイメージが保持されるようになったことが考えられる．加えて，2 回目以降で 20 秒程度の遅延の原因は IPFS のブロックの取得処理が原因であると考えられる．

## 6. 関連研究

Reboot-oriented IoT[11] は使い捨ての利用を想定した IoT デバイスに対して，ネットワークブートを使用することでセキュアなソフトウェア環境を構築している．この研究ではネットワークブートを同時に行った際に生じるブートストームへの対処は考慮されていないため，本研究のシステムを活用することで大規模な IoT デバイス群が分散している状況下においても迅速なネットワークブートを行うことが可能になる．

深谷らの p3cache[12] は P2P 技術を用いてメモリにキャッシュされているディスクイメージを近隣の端末に配布することでネットワークブートによるサーバへの負荷を削減しブートに必要な時間の短縮を実現している．しかし，p3cache では実装に ATA over Ethernet (AoE)[13] を利用しており IP，UDP，TCP といったプロトコルを利用することができないため，IoT デバイスが LAN を跨いで分散している場合にはオーバレイネットワークの利用などの対策が必要となる．加えて，p3cache ではハイブリッド型の P2P 通信を利用しており，メタデータを管理するサーバがボトルネックとなる課題も存在する．本研究では，AoE の代わりに NBD を活用することで LAN を跨いだ通信を実現している．さらにピュア型の P2P 通信を利用している IPFS を活用することでデータ検索の速度は p3cache に劣るがメタデータを管理する端末を必要としないという利点も存在する．

八田ら [14] はブート対象の端末のローカルストレージにディスクイメージをキャッシュする ReadCache システム [15] を改良した．クライアント端末は初回起動時にサーバから取得したディスクイメージを自身のストレージにネットワークブートに適した形式でキャッシュし，次回以降のネットワークブートではキャッシュを利用することでサーバの負荷削減を行っている．この手法ではキャッシュを保持するためにクライアントの端末のストレージを利用している．本システムではクライアントのストレージにデータを保持しないため，IoT デバイスの盗難によるストレージからのデータ流出のリスクを削減することができる．

## 7. おわりに

本研究は広域に分散する大規模な IoT デバイス群を対象としたネットワークブートにおけるブートストームの対処を目的としている．提案システムの実現が可能か検証するために IPFS 上のコンテンツをブロック単位で取得してネットワークブートを行うプロトタイプを実装した．

プロトタイプを用いて実験した結果，IPFS 上のディスクイメージをブロック単位で取得してネットワークブート

を行うことは可能であることが明らかになった．一方で，プロトタイプの実装を用いたネットワークブートは通常のネットワークブートと比較して起動時間に遅延が生じることも明らかになった．

今後の課題として，IPFS 上からブロック単位でディスクイメージを取得した際に生じるオーバヘッドの原因について調査し，改善する必要がある．加えて，プロトタイプによる検証により IPFS 上からディスクイメージを取得してネットワークブートを行うことが可能であることが明らかになったため，提案システムの実装を行う．具体的には 2 のストレージシステム層に存在する NBD に対して IPFS ネットワークとの通信機能を導入する．加えて，起動対象の端末からディスクイメージのキャッシュを取得し再配布する機能を導入する必要がある．

**謝辞** 本研究は，JSPS 科研費 JP21K11832 の助成を受けたものです．

## 参考文献

[1] Satyajit Sinha: State of IoT 2021: Number of connected IoT devices growing 9% to 12.3 billion globally, cellular IoT now surpassing 2 billion. IoT Analytics. 2021. 入手先 ⟨https://iot-analytics.com/number-connected-iot-devices/⟩ （参照 2022-1-23）.

[2] 国立研究開発法人情報通信研究機構: NICTER 観測レポート 2020 の公開. 国立研究開発法人情報通信研究機構. 2021. 入手先 ⟨https://www.nict.go.jp/press/2021/02/16-1.html⟩ （参照 2022-1-23）.

[3] 森田 秀一: “野良 IoT”が“サイバーデブリ”と化し、ネット空間の環境汚染問題を引き起こす恐れ. INTERNET Watch. 2017. 入手先 ⟨https://internet.watch.impress.co.jp/docs/event/1089514.html⟩ （参照 2022-1-23）.

[4] IoT 推進コンソーシアム: IoT セキュリティガイドライン ver 1.0. 総務省，経済産業省. 2016. 入手先 ⟨https://www.soumu.go.jp/main_content/000428393.pdf⟩ （参照 2022-1-23）.

[5] Protocol Labs: IPFS powers the Distributed Web. Protocol Labs. 入手先 ⟨https://ipfs.io/⟩ （参照 2022-1-24）.

[6] Wouter Verhelst: Network Block Device. SourceForge. 入手先 ⟨https://nbd.sourceforge.io/⟩ （参照 2022-1-23）.

[7] abligh: gonbdserver. 2017. 入手先 ⟨https://github.com/abligh/gonbdserver⟩ （参照 2022-1-24）.

[8] Arch Linux: Overlayfs. Arch Linux. 2017. 入手先 ⟨https://wiki.archlinux.jp/index.php/Overlayfs⟩ （参照 2022-1-24）.

[9] Arch Linux: tmpfs. Arch Linux. 2020. 入手先 ⟨https://wiki.archlinux.jp/index.php/Tmpfs⟩（参照 2022-1-30）.

[10] Rhonda D’Vine: Ubuntu – focal の overlayroot パッケージに関する詳細. Canonical Ltd. 入手先 ⟨https://packages.ubuntu.com/focal/admin/overlayroot⟩ （参照 2022-1-30）.

[11] Kuniyasu Suzaki; Akira Tsukamoto; Andy Green; Mohammad Mannan. Reboot-Oriented IoT: Life Cycle Management in Trusted Execution Environment for Disposable IoT devices. Annual Computer Security Applications Conference. New York, NY, USA, Association for Computing Machinery, 2020, pp. 428-441, DOI:https://doi.org/10.1145/3427228.3427293

[12] 深谷健太, 松原克弥: p3cache：ネットブート型シンクライアント端末群を対象とした P2P 型 OS キャッシュ共有機構. コンピュータシステム・シンポジウム論文集. 2019, vol. 2019, pp.15-22.

[13] Arch Linux: ATA over Ethernet . Arch Linux. 2016. 入手先 ⟨https://wiki.archlinux.jp/index.php/ATA_over_Ethernet⟩ （参照 2022-1-24）.

[14] 八田 直樹, 丸山 伸, 松川 正義, 西村 浩二, 相原 玲二. ネットブート環境における読み込みキャッシュ機構の改善による起動時間短縮の試み 電子情報通信学会技術研究報告. 電子情報通信学会技術研究報告. SITE, 技術と社会・倫理 : IEICE technical report. 2012, vol. 111, no. 484, pp. 77-82.

[15] CO-CONV: ReadCache システム. CO-CONV. 入手先 ⟨https://www.co-conv.jp/product/readcache/⟩ （参照 2022-1-24）.